Contrast ドキュメント

January 17, 2024 EOP 3.10.11

本ドキュメントには、Contrast Security の製品の基本の使用方法、サポートされる使用方法、および推奨の使用方法を記載しています。

製作著作 © 2024 Contrast Security

目次

- Contrast へようこそ ... 13
  - カスタマイズ ... 13
  - 次の手順 ... 15
  - Contrast の仕組み ... 15
    - Contrast は開発サイクル全体で使用可能 ... 15
    - 解析手法とデータソース ... 15
    - Contrast エージェント ... 16
    - エージェントの設定 ... 16
    - 静的スキャン ... 16
    - クラウドネイティブアプリケーションの保護 ... 16
    - インテグレーション ... 17
  - Contrast を体験する ... 17
    - Contrast の環境をカスタマイズ ... 17
    - 手順 1：セキュリティテストのためのアプリケーション設定 ... 19
    - 手順 2：攻撃の防御のためのアプリケーション設定 ... 21
    - 手順 3：コードの修正とアプリケーションの再テスト ... 23
  - Contrast Assess ... 23
    - 機能 ... 24
    - カスタマイズ ... 24
  - Contrast SCA ... 24
    - 機能 ... 24
    - Contrast データ ... 25
  - Contrast Protect ... 25
    - Contrast Protect の仕組み ... 26
    - カスタマイズ ... 26
  - Contrast Scan ... 26
    - 機能 ... 27
    - 関連項目 ... 27
  - Contrast Serverless ... 27
    - 機能 ... 28
    - 利点 ... 28
    - 仕組み ... 28
    - セキュリティとプライバシー ... 29
    - 関連項目 ... 29
  - パフォーマンス ... 30
    - 関連項目 ... 30
  - Community Edition (CE) ... 30
    - Community Edition の機能 ... 30
    - Community Edition ポータルサイト ... 31
    - 次の手順 ... 31
- デベロッパーガイド ... 32
  - Contrast プラットフォーム ... 32
  - 開発中のコード解析 ... 32
  - オープンソースライブラリの解析 ... 32
  - 実行時のコード解析 ... 33
  - 本番ビルドの保護 ... 33
  - 次の手順 ... 33
  - Contrast の解析パス ... 33
    - 開発中の解析パス ... 34
    - オープンソースライブラリの解析パス ... 34
    - 実行時の解析パス ... 34
    - 本番ビルドの防御パス ... 34

GitHub Action 34
コードの解析 35
次の手順 35
CLI でオープンソースライブラリを解析 35
CLI を使用する静的スキャン 36
CLI を使用するサーバレス関数のスキャン 37
CLI を使用する脆弱性の検出 37
GitHub アプリでオープンソースライブラリを解析 38
GitHub Action を使用する静的スキャン 39
エージェントでオープンソースライブラリを解析 39
アプリケーションにエージェントを組み込んで脆弱性を検出 39
Contrast Web インターフェイスを使用してサーバレス関数のスキャンを設定 40
Contrast Web インターフェイスを使用する静的スキャン 40
結果の確認 41
CLI で結果を取得 41
IDE インテグレーションで結果を取得 41
SARIF ファイルで結果を取得 41
Contrast Web インターフェイスで結果を取得 42
攻撃の監視・ブロック 43
アプリケーションにエージェントを組み込んで Protect を使用 43
Contrast Web インタフェースで攻撃情報を確認 43
CI/CD でのオプション 44
エージェント 45
エージェントのインストール 46
Java 47
.NET Framework 47
.NET Core 47
Node.js 48
PHP 48
Python 48
Ruby 48
Go 49
エージェント設定ファイルのダウンロード 49
Java ワークフロー 49
.NET Framework ワークフロー 52
.NET Core ワークフロー 53
Node.js ワークフロー 56
PHP ワークフロー 57
Python ワークフロー 58
Ruby ワークフロー 59
Go ワークフロー 59
エージェントの設定 60
手順 60
エージェントキーの検索 61
優先順位 62
その他の設定 67
アプリケーションの疎通 72
CI/CD パイプラインへの組込み 72
手動テストでの組込み 73
Web アプリケーションテストツールでの組込み 73
API テストツールでの組込み 73
DAST ツールでの組込み 73

オープンソースのクローラーでの組込み 74
手動ペネトレーションテストでの組込み 74
Burp Suite ベースのペネトレーションテストでの組込み 74
Assess データを使用して curl コマンドを実行 74
Java (Kotlin、Scala) 75
サポート対象テクノロジ 75
インストール 77
設定 103
旧 Java エージェント 165
Java エージェントのテレメトリ 166
.NET Framework 166
サポート対象テクノロジ 167
システム要件 168
インストール 168
設定 181
IIS Express で使用 214
Azure で使用 215
Azure Service Fabric 215
プロファイラチェーン 217
.NET エージェントエクスプローラ 218
.NET Framework Contrast トレイ 220
アプリケーションプール 222
テレメトリ 223
.NET Core 224
サポート対象テクノロジ 225
システム要件 226
インストール 227
設定 247
.NET エージェントエクスプローラ 277
プロファイラチェーン 279
テレメトリ 281
Azure Functions 283
Node.js 284
Contrast サービス 285
サポート対象テクノロジ 285
システム要件 288
インストール 289
設定 304
コンテナ起動時間の短縮 338
ESM で Node.js エージェントを使用する 339
トランスパイラ、コンパイラ、ソースマップ 340
テレメトリ 341
PHP 341
サポート対象テクノロジ 342
システム要件 342
インストール 342
設定 345
Python 359
システム要件 359
サポート対象テクノロジ 360
インストール 360
設定 362
テレメトリ 414
Contrast ランナー 414
Ruby 415

システム要件 415
サポート対象テクノロジ 416
インストール 417
設定 419
テレメトリ 475
Go 476
サポート対象テクノロジ 476
Go エージェントのインストール 477
設定 480
Contrast サービス 494
インストール 495
設定 495
インストール 496
設定 497
エージェントオペレータ(Kubernetes オペレータ) 498
セキュリティポリシー 498
関連項目 498
サポート対象テクノロジ 498
ネットワーク要件 499
インストール 500
アンインストール 508
設定 509
.NET Core によるチェーンのサポート 514
テレメトリ 514
エージェントのパフォーマンス 515
関連項目 515
Protect 使用時のパフォーマンス 515
ユーザガイド 519
Contrast のサポート対象言語 520
Contrast Scan による追加のサポート対象 520
ユーザの設定 521
ログイン 521
パスワードの変更 521
2 段階認証 521
プロファイルの管理 522
API キーの確認 522
通知の管理 523
権限の表示 523
プロジェクト 524
関連項目 524
プロジェクトの表示 524
プロジェクトの情報をエクスポート 525
アプリケーション 526
表示 526
アプリケーションの設定の編集 530
タグ 531
マージ 532
アーカイブ 533
リセット 534
削除 535
セッションメタデータ 535
ルートカバレッジ 538
フローマップ 544
スキャン 546
スキャンの操作 546

関連項目 546
Contrast Scan リリース情報 546
スキャンのプロセス 553
Contrast Scan のサポート対象言語とテクノロジ 554
スキャンパッケージの準備 556
スキャンプロジェクトの表示 556
スキャンプロジェクトの作成 557
スキャンプロジェクトの削除 559
スキャンの開始 560
スキャンのキャンセル 561
スキャンの確認 561
SAST コンプライアンス対応レポート 562
Contrast Scan ローカルエンジン 563
結果の分析 569
スキャンポリシーの表示 582
スキャン設定の変更 582
スキャンプロジェクトのアーカイブ 583
スキャンプロジェクトのアーカイブ解除 583
ビルドパイプラインでのインテグレーション 583
サーバ 588
サーバの設定 588
設定ファイルの定義 588
エージェントの設定手順 589
オプションの設定 589
サーバの表示 589
設定 593
アプリケーションのサンプリング 594
自動診断データ収集 594
Syslog への出力 596
ライブラリ 599
関連項目 600
SCA リリース情報 600
ライブラリの表示 600
検出と削除 603
タグ 603
送信 605
実行時のライブラリの使用状況 605
エクスポート 607
オープンソースライセンス 609
依存関係ツリーの表示 609
ライブラリのスコアガイド 610
CVE 検索 611
サーバレス 611
関連項目 612
リリース情報 612
Contrast Serverless のサポート対象の言語 638
Contrast Serverless のサポート対象のプラットフォーム 638
マルチリージョンのサポート 638
インベントリ 640
スキャンの種類と監視について 640
AWS で使い始める 641
Azure で使い始める 647
オンデマンドで関数をスキャン 648
結果の表示 649
インベントリ基準の変更 654

サーバレスのスキャン設定の変更 655
関数とサービスの関係の表示 655
コンテキストのリスクスコア 659
Contrast Serverless のアップグレード 662
アカウントのブロック 665
オフボード 665
アンインストール 666
Contrast CLI 666
開始する前に 666
Contrast CLI について 666
Contrast CLI のサポート対象言語とパッケージマネージャ 666
Contrast CLI のインストール 667
認証 668
解析 668
Contrast CLI コマンド 674
旧 Contrast CLI 685
脆弱性 692
組織の表示 692
アプリケーションの脆弱性の表示 693
脆弱性の発生率の表示 695
脆弱性の削除 696
シンクで脆弱性をグループ化 697
脆弱性のマージ 697
タグ 697
追跡 699
イベント 700
修正 701
エクスポート 701
CVE に関連付けられている CWE の検索 703
ステータス 703
深刻度の変更 707
攻撃 707
イベントデータの保持 708
操作 708
攻撃の表示 708
攻撃の監視 710
攻撃の管理 711
攻撃にタグを付ける 713
攻撃スクリプトの実行 714
Contrast Security GitHub アプリ 715
使い方 715
Contrast Security GitHub アプリでサポートされる言語 716
インストールと認証 716
リポジトリの追加・削除 717
トラブルシューティング 718
レポート 718
コンプライアンス対応レポート 718
DISA STIG Viewer チェックリスト 720
SBOM(ソフトウェア部品表) 721
脆弱性の傾向レポート 722
組織の統計値 723
修復の概要 724
インテグレーション 726
クラウドでの連携 726
チャットツール 727

コードリポジトリとの連携 728
継続的インテグレーション(CI)とビルドツール 729
エンタープライズおよび拡張 731
IDE プラグイン 733
インシデント管理システム 734
SIEM ツール 735
作業管理プラットフォーム 736
Agile Central 737
認証情報の管理 738
AWS Security Hub 739
開始する前に 739
設定 739
Contrast Assess のアプリケーションを設定 739
再試行の仕組み 740
AWS Security Lake 740
開始する前に 740
AWS でカスタムソースを作成 740
AWS Security Lake に接続 740
Contrast Assess のアプリケーションを設定 741
再試行の仕組み 741
Azure Boards 741
関連項目 742
接続 742
チケットの自動作成 742
双方向インテグレーション 743
個人用アクセストークン 744
Azure Pipelines 744
インストールと設定 744
タスクの設定 745
リリースゲートの追加 745
Azure Service Fabric 746
Bamboo 748
インストール 748
しきい値の設定 748
Bugzilla 749
Eclipse 749
Generic Webhook 750
Generic Webhook 751
変数 752
イベントと変数 754
GitHub 755
Gradle 756
サンプルアプリケーション 756
プラグインの使用 757
IntelliJ プラグイン 757
IntelliJ プラグインをインストールして設定し、使用するには： 758
IntelliJ で Java エージェントを設定 758
Jenkins 758
Jenkins プラグインのインストールと使用 759
関連項目 759
接続の定義 759
システムレベルのセキュリティ制御 760
ジョブレベルのセキュリティ制御 761
パイプラインのセキュリティ制御 761
Jenkins でのセキュリティ制御 763

ジョブ結果ポリシーの定義 763
ビルドの実行 767
Jira 767
関連項目 767
Jira に接続する 767
Assess と Jira の設定 768
サーバレスと Jira の設定 769
認証情報の管理 771
Maven 772
関連項目 772
Microsoft Teams 773
PagerDuty 773
Solutions Business Manager 774
ServiceNow 774
ServiceNow に接続 774
Slack 775
VictorOps 776
Visual Studio 776
Visual Studio Code 777
Visual Studio for Mac 778
管理ガイド 780
ルールとポリシー 780
Assess ルール 782
セキュリティ制御 783
脆弱性ポリシー 786
Protect ルール 792
CVE シールド 796
仮想パッチ 801
ログエンハンサー 802
アプリケーションの例外 804
コンプライアンスポリシー 807
IP の管理 808
ライブラリポリシー 810
機密データのマスキング 811
通知 813
組織 813
Assess を有効にする 813
Protect を有効にする 815
設定 818
通知 831
スコア 834
アクセス制御 834
脆弱性の承認 868
監査ログの表示 869
なりすまし 872
システム管理(EOP) 874
はじめに 874
Contrast のインストール 875
次の手順 875
Contrast のシステム要件 875
サイジングの推奨 876
curl で Contrast をダウンロード 877
Contrast インストーラのダウンロード 878
インストール 879
WAR ファイルを使用して Contrast をデプロイ 882

分散型 MySQL 883
分散環境の構成 886
Contrast の実行 889
認証情報の管理 889
Contrast の再起動 890
アンインストール 890
インストール後の作業 891
インストール後の作業 891
Tomcat 891
JRE 892
HTTPS の設定 892
HTTP ヘッダの設定 896
MySQL のカスタマイズ 896
プロキシ構成の設定 897
レポート用ストレージの設定 898
Contrast のログ 899
Redis を共有キャッシュに使う(オンプレミス版) 901
システムの更新とアップグレード 902
更新およびアップグレード 902
Contrast のアップグレード 902
エージェントのアップグレード (オンプレミス版) 904
IP アドレスの更新 904
SCA ライブラリデータを手動で更新 904
SCA ライブラリデータを自動で更新 906
Contrast ライセンスの更新 906
運用管理 907
複数の組織の管理 908
組織の追加/編集 908
ユーザと権限 910
ユーザの追加 910
複数ユーザの追加 911
SuperAdmin の指定 913
アクセスグループの追加 913
Protect 権限の付与(オンプレミス版) 915
ユーザの自動追加 916
認証情報の管理 918
ユーザのなりすまし 919
認証 920
2 段階認証 921
Active Directory 922
LDAP 926
SSO 931
HTTPS プロキシ 934
パスワード 934
キー 936
システムの設定 936
手順 937
その他のシステム設定 937
全般的な設定 938
診断情報 938
ライセンス 939
スコア 942
ライブラリコンプライアンスポリシー 942
メール 943
メンテナンス 943

暗号化プロパティエディタ 943
MySQL のバックアップ 945
SSL の管理 947
リファレンス 949
用語集 949
ロールと権限 953
アプリケーションロール 953
組織ロール 955
システムロール 957
アプリケーションのスコアガイド 958
ライブラリのスコアガイド 959
ログレベル 960
イベントと変数 960
変数 961
正規表現 963
対応ブラウザ 963
ベータ版利用規約 964
プライバシーとデータの収集 964

Contrast へようこそ

Contrast は、ソフトウェア開発ライフサイクル(SDLC)の全てのフェーズで、リアルタイムのアプリケーションセキュリティを提供します。

Contrast の使用例として、Contrast を体験する (17ページ)を参照ください。

ご要望は...	Contrast が提供するのは...
SDLC の開発フェーズ・テスト(QA)フェーズでアプリケーションの脆弱性を分析したい： • **開発フェーズで：**アプリケーションや使用するライブラリの脆弱性に関する高精度なフィードバックを即座に得たい アプリケーションを実行することで、アプリケーション内の疎通ルートをシミュレートし、Contrast からの情報を使用して安全なコードをチェックイン • **テストフェーズで：**手動・自動化されたテストケースを適用する時や、CI/CD パイプラインにおいて、アプリケーションの脆弱性は検証済みであることを保証したい • **本番環境で：**SDLC の運用フェーズで、攻撃を完全に把握し、悪意のある不正利用からアプリケーションを防御したい	• エージェント (45ページ)：アプリケーションにセンサーを搭載し、解析します。さまざまなプログラミング言語、フレームワーク、コンテナテクノロジをサポートします。 • Contrast Assess (23ページ)：調整可能な検出ルールを使用して、脆弱性を高精度に検出します。問題が検出された経緯、再現方法、修正方法についての情報を提供します。 • Contrast Scan (26ページ)：高速で効率的な静的スキャンの実行により、アップロードされたバイナリパッケージの脆弱性を特定します。 • Contrast Protect (25ページ)：攻撃を自動的に特定し、本番環境での攻撃を監視したり悪用されるのを防ぎます。Protect は実行中のアプリケーション内から攻撃を検知してブロックしますが、WAF(Web アプリケーションファイアウォール)と統合することもできます。
アプリケーションが使用するライブラリを解析したい	Contrast SCA： (24ページ) アプリケーションの実行時に使用されるオープンソースライブラリがもたらすセキュリティリスクや法的な問題を可視化します。Contrast SCA は、オープンソースライブラリの脆弱性を検出します。また、使用中のライブラリが古く、更新する必要があるかどうかも明らかにします。
SDLC の早い段階でコードの脆弱性を見つけ、その修正方法について分かりやすい説明が知りたい	Contrast Asess、Scan、SCA で検出された脆弱性 (692ページ)の情報には、修正方法 (701ページ)が記載されます。
組織内や組織外でデータやリソースが共有されている箇所をインタラクティブに表示するアーキテクチャを図で見たい	フローマップ (544ページ)により、アプリケーション、アプリケーション内のテクノロジ層、アプリケーションが接続しているバックエンドシステムなどを表す詳細な構成図を参照できます。
CI/CD パイプラインに Contrast を組み込みたい	多種多様なインテグレーション (726ページ)機能により、Contrast の処理やデータを、開発者用の IDE やビルドシステム、コミュニケーションツールなどと連携することができます。

カスタマイズ

Contrast にはさまざまなオプションがあり、Contrast に登録したアプリケーションのデータアクセスやデータ表示、データ収集などをカスタマイズできます。カスタマイズすることにより、Contrast で提供されるデータをより分かりやすく効果的に参照できるようになります。

オプション	説明

ロールベースのアクセス制御	アクセスグループ (822ページ)を使用すると、特定のユーザに権限と機能を割り当てることができます。グループに関連付けたアプリケーションごとに、役割(ロール)に応じた異なる種類のアクセス権を割り当てることができます。
	
	Contrast にアプリケーションを登録する前に、グループ構成を検討しておくと便利です。
	
	最初にアプリケーションを Contrast に登録する際に、Contrast 設定ファイルにグループを指定しない場合は、Contrast Web インターフェイスからのみグループに追加できます。Contrast 設定ファイルを使用してアプリケーションを登録したい場合は、アクセスグループに関連付けるために、アプリケーションを一度削除してから再度登録する必要があります。
	
	まず、Contrast Web インターフェイスで、既存のグループにユーザやアプリケーション(またはその両方)を作成または追加します。
	
	次に、各アプリケーションの Contrast 設定ファイルを使用して、アプリケーションをアクセスグループに関連付けて Contrast に登録することができます。
	
	`# application:`
	
	`  # Add the name of the application group with which this`
	`  # application should be associated in the Contrast UI.`
	`  # group: NEEDS_TO_BE_SET`

カスタムフィルタ	Contrast では、カスタマイズしたフィルタを作成できるタグオプションがいくつかあります。カスタムフィルタを作成する利点は、デフォルトのフィルタを使用するだけでなく、特定のニーズに合わせてデータを表示できることです。 アプリケーションのメタデータ (830ページ)を使用して、カスタムフィルタを作成できます。 また、特定のアプリケーション (531ページ)データや脆弱性 (697ページ)情報にタグを適用することもできます。アプリケーションや脆弱性にタグを付けた後は、アプリケーションのページや脆弱性のページでフィルタとしてそれらのタグを使用できます。 **例：アプリケーションメタデータ** アプリケーションのメタデータを収集するために、Contrast Web インターフェイスでフィールドを作成します。以下は、フリーフォーマットのフィールドの例です。 **フィールド：** managersInfo **値：** `"John Doe"` **フィールド：** businessUnit **値：** `"NodeGoat Group"` **フィールド：** officeLocation **値：** `"New York City"` **例：アプリケーションのタグ** • **Appname：**特定のアプリケーション名 • **Groupname：**アクセスグループの名前 • **Environment：**アプリケーションをテストする環境(開発、QA、または本番環境) • **Server Name：**アプリケーションをホストするサーバの名前 **例：脆弱性のタグ** • **Build：**特定のビルド番号 • **Version：**特定のリリースバージョン
アプリケーションのカスタムデータ	セッションメタデータ (535ページ)により、アプリケーションの脆弱性のソースを特定できます。 セッションメタデータに必要なプロパティをエージェントの設定ファイルに追加すると、標準の脆弱性データと一緒に、セッションメタデータがエージェントにより報告されるようになります。そして、そのセッションメタデータを Contrast Web インターフェイスでフィルタとして使用することができます。 Contrast エージェントの設定ファイル内でセッションメタデータの値を変更すると、異なる値で脆弱性データをフィルタ処理できます。例えば、ブランチ名やバージョンの値を変更すると、異なるブランチやバージョンでデータをフィルタできます。 **例：** 以下の Java アプリケーションの例では、javaagent フラグを追加する行にエントリを追加しています。この例では、`contrast.application.session_metadata` プロパティにブランチ名(branchName)、コミットしたユーザ(committer)、リポジトリ(repository)のメタデータプロパティに対して、キーと値をペアにして指定しています。 `-Dcontrast.application.session_metadata="branchName=build22,committer=Jane,repository=Contrast-Java”`

オプション	説明
名前のカスタマイズ	アプリケーション (530ページ)やアプリケーションをホストするサーバ (593ページ)の名前を変更することができます。 デフォルトでは、Contrast エージェントがコード内で検出したデータに基づいて名前が付けられます。 カスタム名を指定するには、アプリケーションを追加 (46ページ)する際にエージェント設定ファイルで指定するか、アプリケーションを追加した後に Contrast Web インターフェイスで名前を変更します。

次の手順

- Contrast の仕組み (15ページ)について概要を把握する
- Contrast を無料で試す - Community Edition (CE) (30ページ)
- Contrast エージェントをインストール・ 設定する (46ページ)
- Contrast をインストールする(オンプレミス版) (874ページ)
- 特定のツールやテクノロジとインテグレーション (726ページ)する

Contrast の仕組み

Contrast Security は、アプリケーションポートフォリオにおいて、高精度で継続的なリアルタイムのアプリケーションセキュリティテストと攻撃のブロックを実現可能にします。Contrast が各アプリケーション内で動作し、ソフトウェア開発ライフサイクル(SDLC)全体を通して、アプリケーションを安全なものにします。

Contrast を使用して実施する機能テストがセキュリティテストになります。そのため、品質管理の作業としてアプリケーションを疎通するたびに、セキュリティフィードバックを得ることができます。Contrast の結果は継続的かつリアルタイムに得られるため、ソースコードから実行中のアプリケーションまで、開発パイプライン全体にセキュリティを統合することができます。

Contrast は開発サイクル全体で使用可能

解析手法とデータソース

Contrast では、以下のように様々なデータソースと解析手法が統合されています。

- ランタイムの制御フローやデータフロー(IAST)

- アプリケーションのコードや API(SAST)
- HTTP リクエストとレスポンス
- アプリケーション内の全てのライブラリおよびフレームワークとその使用状況(SCA)
- 設定情報
- バックエンドの接続状況
- ローカルファイルの静的スキャン(SAST)

Contrast エージェント

Contrast Assess と Contrast Protect は、エージェント (45ページ)を使用してデータフローを解析し、実行中のアプリケーションから脆弱性を検出します。Contrast Assess と Contrast Protect は、同じエージェントを使用して、データフローの解析や脆弱性の検出を行います。Assess 用のエージェントと Protect 用のエージェントの両方が必要になるわけではありません。

エージェントを追加・設定する (46ページ)と、カスタムコードやライブラリ全体にわたって、アプリケーションの既存のメソッドに Contrast のセンサーが組み込まれることになります。エージェントのセンサーは、データがアプリケーションに入って出る場所(ルート)を観測します。この動作により、アプリケーションのデータの流れがリアルタイムに可視化され、そのコードパスにあるセキュリティ上の欠陥や脆弱性が検出されて、Contrast に報告されます。また、エージェントによって Contrast は攻撃を検知して、攻撃のブロックが可能になります。

エージェントの設定

エージェントの設定は、YAML 設定ファイルの編集、コマンドラインでの環境変数の使用、または使用している言語やツールに固有のその他の方法で行います。

アプリケーションにエージェントを設定する際に、以下の情報を指定します。

- エージェントが Contrast と通信するための情報
- エージェント固有の設定
- Assess および Protect ルールの設定
- アプリケーション固有の設定
  - ここでの設定には、セッションメタデータやアプリケーションメタデータも含まれます。メタデータは、報告された各脆弱性の追加情報として、またはフィルタとして利用できます。
- アプリケーションとエージェントをホストするサーバ：
  - 統合開発環境(IDE)で実行している開発者のローカルアプリケーションサーバ
  - 自動テストプロセスで使用している CI(継続的インテグレーション)アプリケーションサーバ
  - アプリケーションのテストサーバ
  - アプリケーションのステージングサーバ
  - アプライアンスに組み込まれているサーバ
  - 仮想マシンで実行中のアプリケーションサーバ
  - クラウドで実行中のリモートアプリケーションサーバ
  - 本番環境用のアプリケーションサーバ

静的スキャン

Contrast Scan (546ページ)は、ソフトウェア開発ライフサイクル(SDLC)の開発フェーズにおいて、脆弱性の検出と修正を容易にする静的アプリケーションセキュリティテスト(SAST)ツールです。

アプリケーションをスキャンするには、ソースコードかバイトコードファイルをアップロード (557ページ)します。 Contrast のテクノロジにより、Contrast が定義済した一連のルールに基づいて脆弱性が特定されます。

クラウドネイティブアプリケーションの保護

Contrast Serverless (27ページ)は、サーバレスベースのアプリケーションを対象とした、次世代のアプリケーションセキュリティテストツールです。

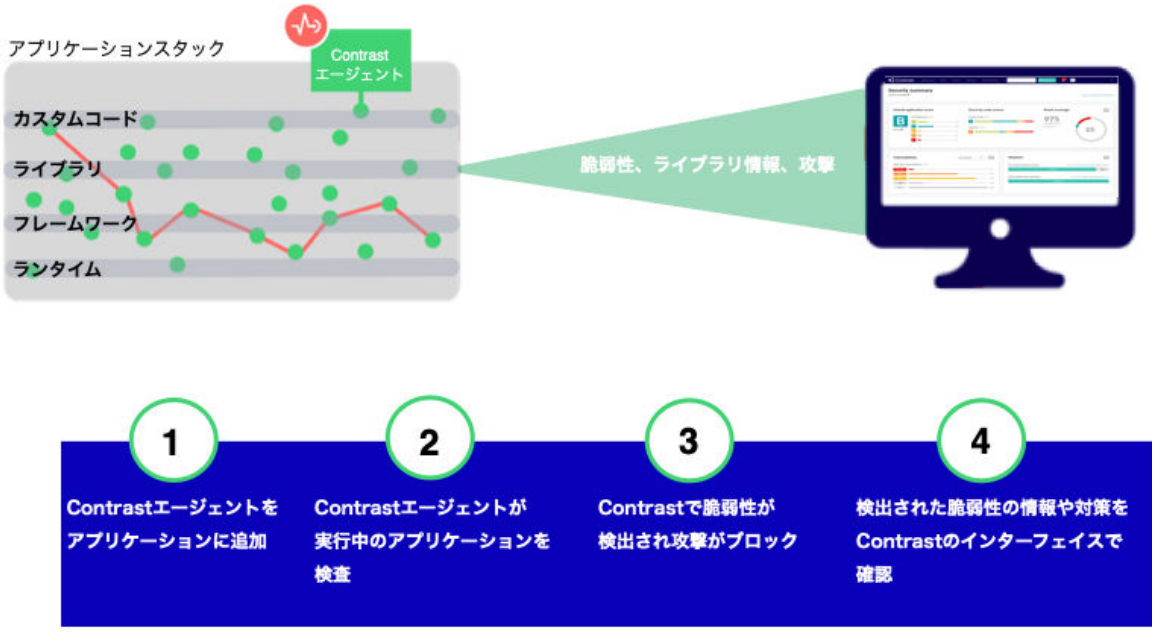
Contrast Serverless は、クラウドネイティブアーキテクチャを使用して環境内の全てのリソースをマッピングし、自動的にその結果を検証して優先順位付けを行い、誤検知や不用なアラートを解消します。AWS アカウントに読み取りモードで接続し、継続的に環境を監視し、関連情報を収集します。

インテグレーション

Contrast は、さまざまなツールやテクノロジとインテグレーション (726ページ)できるため、現在ご利用中のツールから脆弱性に関する高精度なフィードバックを得ることができます。このアプローチにより、セキュリティチームと開発チーム間の連携が効果的に促進され、ソフトウェアの開発プロセスが加速します。

Contrast を体験する

Contrast を使用して、重大な脆弱性がコードに導入されるのを防ぎ、アプリケーションを攻撃から守る方法について詳しく見ていきましょう。



本項では、Contrast Java エージェントを使用して、アプリケーションを疎通しながらセキュリティテストと攻撃の防御を行うために、アプリケーションを設定する例をご紹介します。

Contrast の環境をカスタマイズ

Contrast Web インターフェイスにアクセスできるようになったら、Contrast データへのアクセス制御や重大な検査結果の検索が簡単にできるように、環境をカスタマイズすることを検討しましょう。

アクセスグループ

例えば、お使いの財務アプリケーションに 3 つの担当が関与しているとします。 その場合、これら 3 つの担当用のアクセスグループを作成し、Contrast の設定ファイルに指定します。

- **Team1-Dev** は開発者向けで、次のような設定をします。

開発者には、検出結果の修正、タグの追加、脆弱性の管理、属性の編集、アプリケーションのマージ、アプリケーションの追加· 削除、サーバの作成などを可能にします。

- **Team2-Test** はテスト担当用で、次のような設定をします。

テスト担当者は、スコア、ライブラリ、脆弱性、コメントの参照が可能で、アプリケーションのトレースの編集はできません。

- **Team3-AppSec** はアプリケーションセキュリティ担当用で、次のような設定をします。

アプリケーションセキュリティ担当には、アプリケーションのルールやポリシーの編集、Protect の有効化、組織の通知やスコア付けの管理ができるようにします。

アプリケーション名とサーバ名

すべての担当が同じアプリケーションで作業しているため、Contrast の各設定ファイルには同じアプリケーション名を使用することにします。開発環境用に 1 つの設定ファイル、テスト環境用に 1 つの設定ファイルを使用する予定です。

2 つの設定ファイルを使用することになりますが、両方の設定ファイルでアプリケーションに同じ名前を指定するため、Contrast ではアプリケーションのインスタンスが 1 つしかないかのようにデータが表示されることになります。

サーバ名は指定せずに、Contrast で検出される名前をそのまま使用することにします。

セッションメタデータ

アプリケーションに関して、特定のブランチ、コミットしたユーザ、リポジトリなどに対する脆弱性やルート情報を表示することができます。このような情報を収集するには、Contrast の設定ファイルでセッションメタデータの値を定義します。

```
-Dcontrast.application.session_metadata="branchName=release24,committer=Jane
,repository=finapp-Java”
```

手順 1：セキュリティテストのためのアプリケーション設定

開発時には、開発者が安全なコードをチェックインしていることを確実にしたいと考えます。 また、テスト中には、本番環境での攻撃を可能にする脆弱性がアプリケーションにないことを検証したいと考えます。

Contrast をアプリケーションに追加して、製品を一般公開する前に必要なセキュリティテストを行うことにしました。 アプリケーションは Java を使用しているため、Contrast の Java エージェントを使用することになります。

1. ビルドプロセスに Maven を使用しているので、Maven Central リポジトリから Contrast エージェントをダウンロード (78ページ)します。また、Contrast の Web インターフェイスから YAML 設定ファイル (63ページ)をダウンロードします。
2. 開発環境では YAML 設定ファイルを編集して、次のような設定をします。

```
api:
    # ********************** REQUIRED **********************
```

```
    # Set the URL for the Contrast UI.
    url:https://mycontrast.mycompany.com:8080/Contrast/
    # ********************** REQUIRED **********************
    # Set the API key needed to communicate with the Contrast UI.
    api_key:A2xxxxxxxxxxxxxxxxxxxxxxxxxxxxG9N
    # ********************** REQUIRED **********************
    # Set the service key needed to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    service_key:service_key:88xxxxxxxxxxx5Z
    # ********************** REQUIRED **********************
    # Set the user name used to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    user_name:agent_xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx@mydevorg
.
.
.

# \
================================================================================
=====
# server
# Use the properties in this section to set
# metadata for the server hosting this agent.
# \
================================================================================
=====
# server:
  # Override the reported server environment.
  environment: development
.
.
.
```

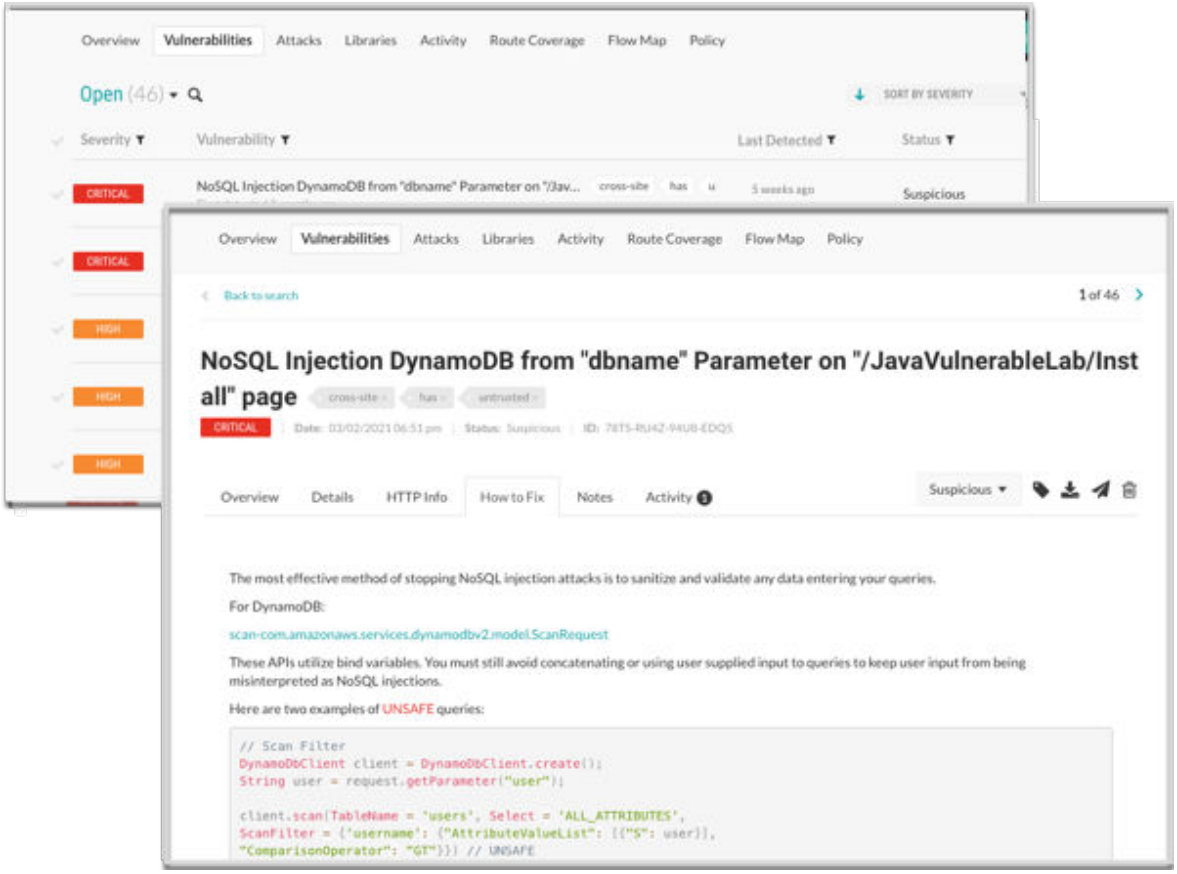
3. テスト環境では YAML 設定ファイルを編集して、次のような設定をします。

```
api:
    # ********************** REQUIRED **********************
    # Set the URL for the Contrast UI.
    url:https://mycontrast.mycompany.com:8080/Contrast/
    # ********************** REQUIRED **********************
    # Set the API key needed to communicate with the Contrast UI.
    api_key:A2xxxxxxxxxxxxxxxxxxxxxxxxxxxxG9N
    # ********************** REQUIRED **********************
    # Set the service key needed to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    service_key:service_key:88xxxxxxxxxxx5Z
    # ********************** REQUIRED **********************
    # Set the user name used to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    user_name:agent_xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx@mydevorg
.
.
.

# \
================================================================================
=====
```

```
# server
# Use the properties in this section to set
# metadata for the server hosting this agent.
# \
================================================================================
=====
# server:
  # Override the reported server environment.
  environment: QA
.
.
.
```

4. 次に、アプリケーションを起動して機能テストを実行し、アプリケーションやビジネスロジックで公開されるの全ルートとデータエンドポイントを疎通します。
5. まず、Contrast Web インターフェイスの「アプリケーション」のページにアクセスして、Contrast でアプリケーションが認識されていることを確認します。次に、脆弱性の有無を確認し、脆弱性に対しては修正方法を参照してコードを安全にするための修正や対策を決めます。



6. 最初のテストの後に、CI/CD プロセスで Contrast と連携するために Maven プラグイン (772ページ) を使用することにしました。この連携では、深刻度がまたはの脆弱性を Contrast が検出した場合にビルドが失敗するように設定します。

手順 2：攻撃の防御のためのアプリケーション設定

開発とテストの段階で Contrast を使用してきましたが、ユーザが製品を使用する際に悪意のある行為を受けないようにしたいと考えます。 アプリケーション、ユーザ、およびデータを保護するために、本番環境にあるアプリケーションに Contrast を追加することにしました。

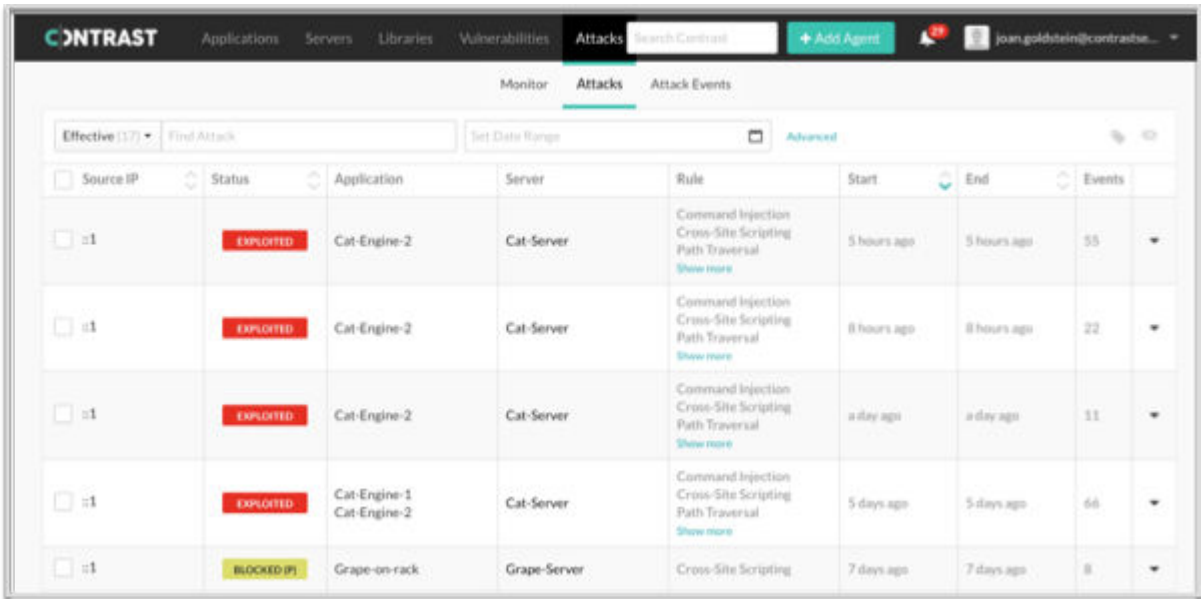
まず Protect ライセンスがあり、組織で Protect が有効になっていることを確認します。

開発時とテスト時にアプリケーションにエージェントをインストールして設定した方法と同様に、本番環境でも Contrast の Protect を有効にするための新しい設定ファイルが必要になります。新しい設定ファイルを作成した後にアプリケーションを実行し、Contrast Web インターフェイスに本番環境のアプリケーションが表示されることを確認します。

```
api:
    # ********************** REQUIRED **********************
    # Set the URL for the Contrast UI.
    url:https://mycontrast.mycompany.com:8080/Contrast/
    # ********************** REQUIRED **********************
    # Set the API key needed to communicate with the Contrast UI.
    api_key:A2xxxxxxxxxxxxxxxxxxxxxxxxxxxxG9N
    # ********************** REQUIRED **********************
    # Set the service key needed to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    service_key:service_key:88xxxxxxxxxxx5Z
    # ********************** REQUIRED **********************
    # Set the user name used to communicate with the Contrast
    # UI. It is used to calculate the Authorization header.
    user_name:agent_xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx@myprodorg
.
.
.
# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Use the properties in this section to determine if the
  # Protect feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: true
.
.
.
# \
================================================================================
==
# server
# Use the properties in this section to set
# metadata for the server hosting this agent.
# \
================================================================================
==
# server:
  # Override the reported server environment.
  environment: production
.
.
.
```

アプリケーションが本番運用になったら、Contrast Web インターフェイスの「攻撃」ページを参照して、攻撃が発生していないか監視します。

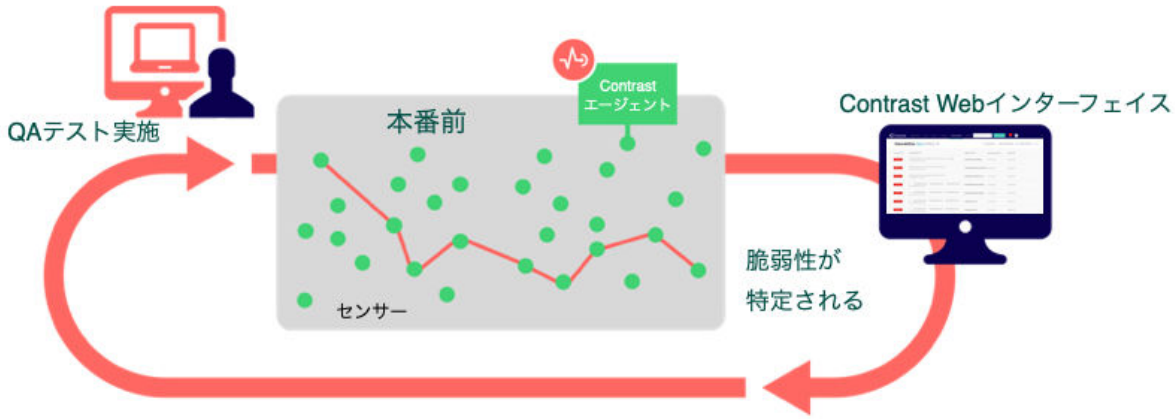


手順 3：コードの修正とアプリケーションの再テスト

Contrast を使用したテストの結果や検知された攻撃を調査したら、アプリケーションのコードを修正します。アプリケーションの最新バージョンが Contrast に表示されていることを確認します。アプリケーションの最新バージョンに、本番環境でブロックした脆弱性が含まれていないことを確認したら、アプリケーションを再デプロイします。

Contrast Assess

Contrast Assess は、静的アプリケーションセキュリティテスト(SAST)、動的アプリケーションセキュリティテスト(DAST)、インタラクティブアプリケーションセキュリティテスト(IAST)のアプローチを組み合わせた、アプリケーションセキュリティのテストツールです。アプリケーションの脆弱性に関して、極めて正確な情報を継続的に得ることができます。



Contrast Assess はエージェントを使用して、アプリケーションにセンサーを配置します。このセンサーがデータフローをリアルタイムで監視し、アプリケーションを内部から解析し、以下のような様々なデータソースから脆弱性を特定するために機能します。

- ライブラリ、フレームワーク、カスタムコード
- 設定情報

- ランタイムの制御フローやデータフロー
- HTTP リクエストとレスポンス
- バックエンドの接続状況

Contrast Assess は、テストサーバ、QA サーバ、ステージングサーバなど、開発環境や試験環境に適しています。また、開発者のワークステーションにも適用できます。Visual Studio などで Contrast とのインテグレーションを利用することで、開発者は利用中の統合開発環境(IDE)から脆弱性を検出し、修正に対応できます。

機能

エージェントをインストールして設定 (46ページ)し、Contrast Assess を有効 (813ページ)にしたら、次の機能を利用できるようになります。

- アプリケーションの脆弱性 (692ページ)の一覧と脆弱性を修復するためのガイダンス
- アプリケーションのセキュリティを一目で判定できるアプリケーションのスコア (958ページ)
- 脆弱性と元の Web リクエストを関連付けることで、可能な限りのルートを検出するルートカバレッジ (538ページ)
- 実行中のアプリケーションのアーキテクチャを把握できるフローマップ (544ページ)
- コンプライアンス対応などのレポート (718ページ)の作成

カスタマイズ

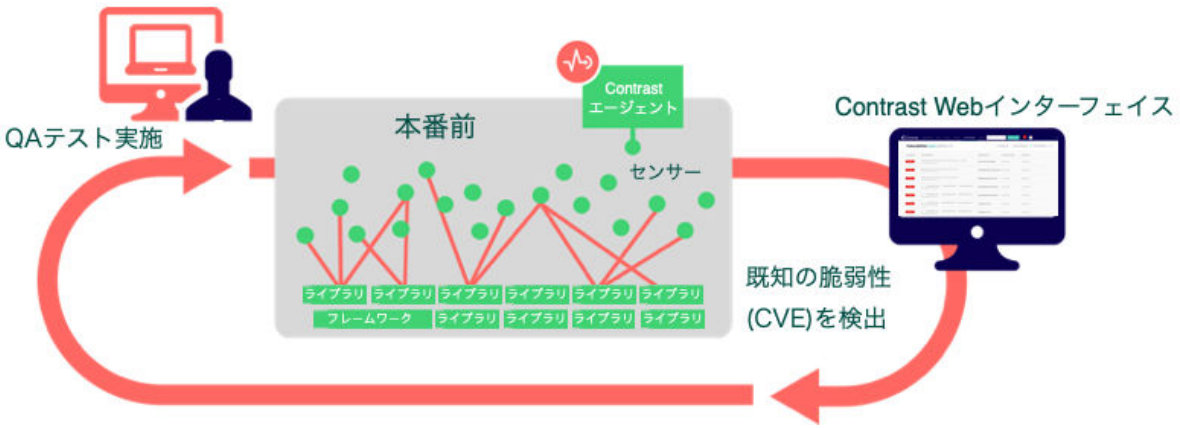
以下のポリシーの設定を変更する事で、特定の要件に合わせて Assess をカスタマイズできます。

- Assess ルール (782ページ): 特定の種類の脆弱性を検出するルールを有効または無効にでき、Contrast Assess の検出機能を調整できます。
- セキュリティ制御 (783ページ) : コード内でデータを安全に使用するためのメソッドを設定します。

Contrast SCA

Contrast SCA は、ランタイム分析、ファイルシステムのスキャン、依存関係の解析によって、オープンソースコンポーネントを特定します。Contrast SCA ではこれらの手法を活用し、正確なインベントリを報告します。

デフォルトで、Contrast Assess には優れた SCA 機能が備わっています。SCA ライセンスを取得すると、さらに高度な SCA 機能を利用できます。



機能

SCA 機能は、Contrast プラットフォームの一部として組み込まれています。プロセスを簡素化し、オープンソースの解析とカスタムコードの解析を統合するためです。Contrast SCA で実行できることは、以下のとおりです(一部の機能は無料ですが、その他には SCA ライセンスが必要です)。

- **オープンソースライセンスの管理** : Contrast SCA によって、オープンソースコンポーネントに関連付けられたライセンス情報 (609ページ)が表示されます。この情報により、知的財産のコンプライアンスを理解し、運用リスクを軽減できます。
  この機能には、SCA ライセンスが必要です。
- **オープンソースポリシーの設定** : ポリシーを設定して、オープンソースライセンスの使用を制限することができます。使用を制限したライセンスがアプリケーションでデプロイされた場合、警告を発します。ライブラリの使用を安全に保つには、組織のコンプライアンスポリシーを設定 (807ページ)します。特定のオープンソースライブラリやライセンスの使用を制限したり、バージョン要件を指定するには、ライブラリポリシーを設定 (810ページ)します。
  この機能には、SCA ライセンスが必要です。
- **脆弱性(CVE)の特定**:Contrast SCA は、アプリケーションが使用している各ライブラリの脆弱性(CVE)の情報を提供します。この情報には、ライブラリの各 CVE の説明と、そのライブラリを使用しているアプリケーションの数などが含まれます。
  この機能は、SCA ライセンスがなくても利用できます。
- **CLI および依存関係ツリー** : Contrast CLI は、アプリケーションに対して SCA(ソフトウェアコンポジション解析)を実行し、オープンソースライブラリ間の依存関係を、脆弱性が導入された場所を含めて表示します。
  Contrast CLI (685ページ)で収集されるデータは依存関係ツリー (609ページ)を表示するために使用され、 ライブラリの依存関係を認識することができます。
  この機能は、SCA ライセンスがなくても利用できます。
- **GitHub Action** : このインテグレーション機能を使用すると、プロジェクトの依存関係の脆弱性を解析できます。Contrast SCA Action を実行することによって、脆弱なライブラリが検出されます。詳細は、Contrast SCA Action を参照してください。

Contrast データ

Contrast にライブラリの情報が報告されると、以下を利用できるようになります。

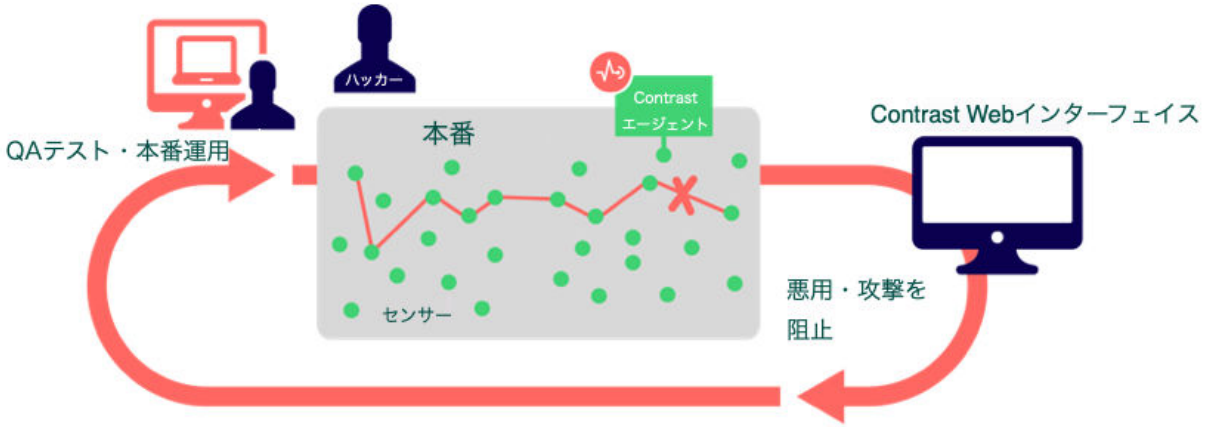
- 脆弱なコンポーネントが実際にアプリケーションで使用されているかを確認するためのライブラリの使用状況分析
- 使用されているライブラリのバージョンと最新バージョンの情報
- ライブラリで検出されている脆弱性の情報
- ポートフォリオ全体で、オープンソースコンポーネントをリアルタイムに報告

Contrast Protect

Contrast Protect は本番環境向けの防御制御で、特定の脆弱性(例えばコマンドインジェクションなど)に応じて、アプリケーションを積極的に防御します。

NIST 800-53、PCI-DSS、PCI-SSS、およびその他の業界標準に準拠する「ランタイムアプリケーション自己保護(RASP)」を提供します。Contrast Protect は、Java (75ページ)、.NET (166ページ)、.NET Core (224ページ)、Node.js (284ページ)、Ruby (415ページ)、Python (359ページ)のランタイム内部で直接動作し、手動で調整をすることなくアプリケーション内のインテリジェンスを活用します。

Contrast Protect は、Web アプリケーションや API を攻撃する自動化された脅威と高度な脅威の両方をブロックし、アプリケーションポートフォリオ全体に渡って、的確で価値ある脅威情報をタイムリーに提供します。



Contrast Protect の仕組み

Contrast Protect は、ネットワークトラフィックではなく、完全なデータフローを把握するために、アプリケーションソフトウェア内で動作します。受信データを解析するだけではなく、そのデータを見て、完全な SQL クエリやコマンド引数など、基になるアクションへの影響を観察します。

この解析は、検知精度を向上させ、誤検知の可能性がある多くの攻撃のノイズを分離し、意図したターゲットに合致する攻撃に焦点を当てることができます。この情報を SIEM などの外部システムと共有して、主要な攻撃イベントに焦点を絞ることができます。

Contrast Protect は、アプリケーションと同じ共有メモリで運用することでアプリケーションのパフォーマンスへの影響を抑え、オーバーヘッドの増加を防ぎます。コンテキストに関する防御に不要なアクションを回避することで、パフォーマンスは向上します。例えば、NoSQL アプリケーションでは、SQL API が呼び出される事がなければ、SQL インジェクションに対するチェックは必要ありません。

カスタマイズ

Contrast Protect が有効な場合、以下のポリシーやルールをカスタマイズできます。

- **Protect ルール (792ページ)**：攻撃を監視するアプリケーションを設定します。
- **CVE シールド** (796ページ)：脆弱性をブロックする CVE シールドを指定します。
- **仮想パッチ (801ページ)**：特定の脆弱性に対するカスタム防御を定義します。
- **ログエンハンサー (802ページ)**：Contrast エージェントによりアプリケーションのパラメータやデータを追加でログに記録できるよう指定します。
- **IP 管理 (809ページ)**：拒否リストと許可リスト(信頼できるホスト)を管理します。

関連項目

Protect 使用時のエージェントのパフォーマンス (515ページ)

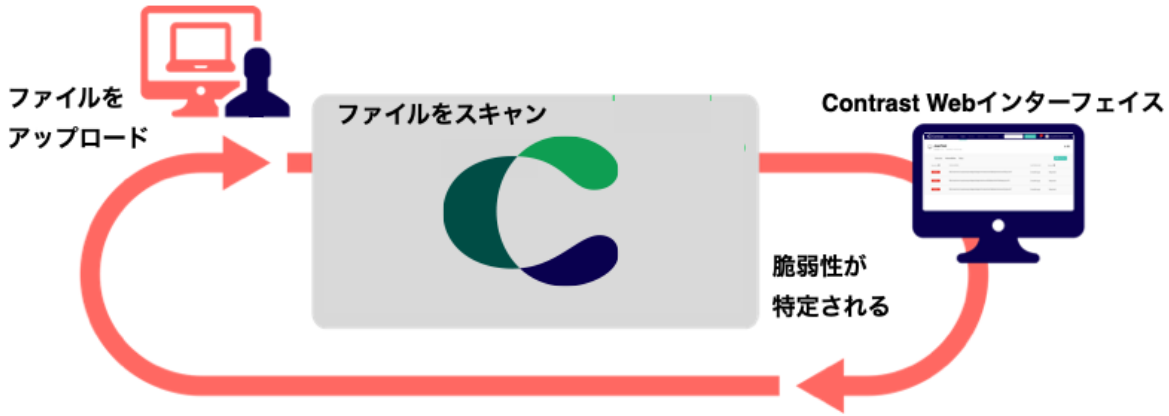
Contrast Scan

Contrast Scan (546ページ)は、脆弱性の検出と修復を容易にする静的アプリケーションセキュリティテスト(SAST)ツールです。 これは、アプリケーションの開発フェーズで使用できる便利なツールです。ライセンスをお持ちで SaaS 版をご利用のお客様は、この機能をご利用いただけます。

アプリケーションをスキャンするには、アプリケーションのバイナリパッケージを Contrast の環境にアップロードします。アプリケーションコードをアップロードしたら、スキャンが開始します。スキャンは、ソースコード内のデータフローを確認し、悪意のある攻撃を可能にする脆弱性を特定します。悪意のある攻撃の例としては、SQL インジェクション、コマンドインジェクション、サーバサイドインジェクションなどがあります。

スキャンの結果、カスタムコード内の脆弱性が特定されます。これらの問題を修正して、スキャンを再度実行すると、コードの変更によって脆弱性が無くなったことを確認できます。

オープンソースのコードとライブラリはスキャンの対象に含まれません。



機能

- 複数のスキャン結果を追跡できるスキャングループの作成
- スキャンの設定(スキャン名の変更)
- スキャンの開始と停止
- 検出された脆弱性の情報の表示
- スキャンの進行状況と履歴の照会
- 脆弱性情報へのステータスの設定
- CI/CD パイプラインへのスキャンの組み込み
- 各脆弱性のリスクと修正方法に関する説明

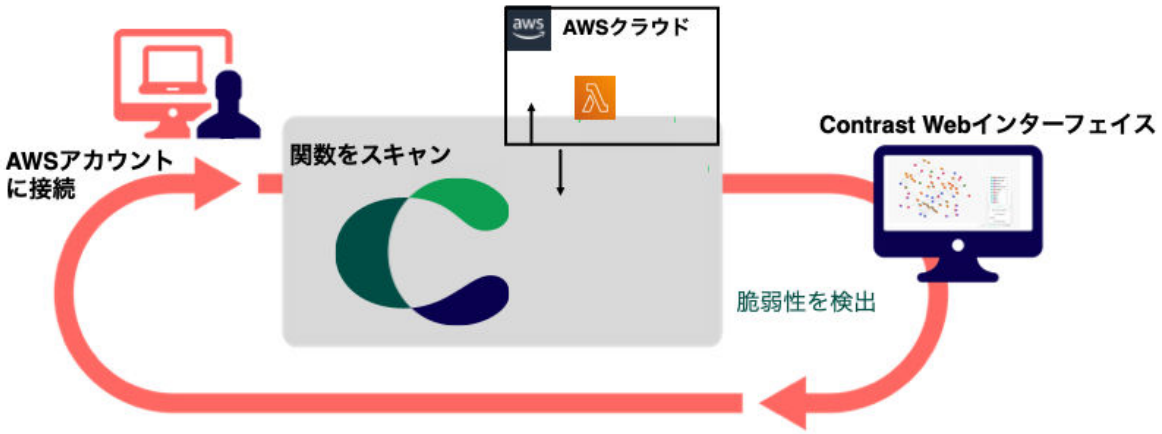
関連項目

Contrast Scan のサポート対象テクノロジ (554ページ)

Contrast Serverless

Contrast Serverless は、サーバレスベースのアプリケーションを対象とした、次世代のアプリケーションセキュリティテストツールです。

Contrast Serverless は、クラウドネイティブアーキテクチャを使用して環境内の全てのリソースをマッピングし、自動的にその結果を検証して優先順位付けを行い、誤検知や不用なアラートを解消します。カスタムコードの脆弱性(インジェクション攻撃など)、依存関係(ライブラリの CVE など)、および設定のリスク(過剰に権限が設定された関数ポリシー)をスキャンします。



機能

- **AWS への容易な接続**
  2 回のクリックと数分で、Contrast の Web インタフェースから AWS アカウントに接続できます。
- **インベントリの検出**
  AWS アカウントに接続すると、AWS 環境にある関数、リソース、ポリシー、サービスのインベントリが作成されます。
- **脆弱性の解析**
  動的スキャンと静的スキャンによってコードが解析され、アプリケーションの弱点やデータフロー、攻撃対象、および脆弱性にさらされている箇所が検出されます。
- **継続的な監視**
  Contrast は常に Lambda 関数を監視し、コードが変更されると、注意が必要な脆弱性を特定します。
- **攻撃のシミュレーション**
  動的スキャンは、コードに変更を加えることなく、リソースとデータフローに対して情報収集のための攻撃を生成して実行します。
- **関数とサービスの関連性を可視化**
  アカウント内の関数とサービスの関連性を表示するだけでなく、各要素に含まれるリスクの詳細を表示できます。
- **レポート作成**
  スキャン結果には、コードにある CVE、権限違反、脆弱性、およびその他のリスクが一覧表示されます。

利点

- 導入がスピーディで簡単
  Contrast と連携するために、多数の専門家やコンサルタント、また多くの時間は必要ありません。
- 連携に煩わされることがない
  開発プロセスに大幅な変更を加えることなく、サーバレスセキュリティを強化できます。開発フェーズの早い段階で悪用可能な脆弱性を簡単に見つけて修正できるため、アプリケーションを本番環境にデプロイする際に、アプリケーションをより安全な状態にすることができます。

仕組み

Contrast は、AWS アカウントに読み取りモードで接続します。この接続を使用して継続的に環境を監視し、関連情報を収集します。

そして、監視対象の環境内に 1 つの Lambda 関数(Cloud Agent)を配置して、コード解析やスキャンしたリソースに関するメタデータを Contrast へ送信するなどの処理を実行します。

これらの情報は全て、Contrast のコンテキストエンジンによって使用され、この環境の全てのリソースと変更に合わせた攻撃プロファイルを生成します。攻撃のシミュレーションは、お客様のアカウント内でクラウドエージェント(Cloud Agent)によって実行されます。

全ての検出結果は、内部検証メカニズムによって検証することにより、誤検知をなくし、現実的な優先順位を付けた結果として提供されます。

セキュリティとプライバシー

- Contrast は、監視対象のアカウントからコードやコードスニペットを収集することはありません。
  Contrast は、次のようなメタデータ情報のみを利用します。
  - 確認された脆弱性
  - 関数名とメタデータ(ポリシーハンドラなど)
  - 使用しているライブラリ
  - 使用している AWS の API(例：Boto3、asw-sdk)
  - サービスの構成(バケット通知、API Gateway のパスやメソッドなど)
- Contrast は、お客様のコードに変更を加えることはありません。ただし、スキャン中に(例えば、関数のデプロイ時や変更時)、スキャンされた関数にレイヤーを一時的に組み込んだり、いくつかの設定変更(例えば、タイムアウトやハンドラ)を行うことがあります。
  スキャンが完了すると、レイヤーや設定は元の状態に戻されます。この処理は完全に透過的で影響を与えず、自動的に実行されます。
  スキャン中でも、引き続き独自のテスト(関数呼び出し)を実行できます。
- 動的スキャンでは、Contrast は悪意のあるデータを使用してスキャンする関数を実行します。この処理は、コードには影響を与えません。ただし、関数が行うアクションはトリガーされる可能性があります。
  この機能はデフォルトで無効になっています。これは、設定タブを使用して、いつでも有効にできます。
- 監視対象の AWS アカウントから Contrast が受信する全てのデータは、転送中および保存時に暗号化されます。
  Contrast は、共有シークレットを用いた Amazon EventBridge を使用して、全てのデータを送受信します。Contrast が AWS アカウントとの通信に使用する Web API や REST API はありません。

必要となる権限

Contrast を使用して AWS アカウントに接続する場合、次のアクセス許可に同意するものとします。

- Contrast の AWS アカウントから監視対象の AWS アカウントへの読み取り専用アクセス権。
  このポリシーは、ベータ版の処理を行う際だけ使用されます。
- Contrast の AWS アカウントから監視対象の AWS アカウントにデプロイされた Lambda 関数への Lambda 読み取り/書き込みアクセス。
- Contrast が監視対象の AWS アカウントにインストールする Lambda 関数には、次のアクセスポリシーが必要です。
  - CloudWatch ログの読み取り
  - Lambda Layer バージョンの読み取り
  - 関数の呼び出し
  - 関数設定の変更
  - イベントバスへのメッセージの書き込み
  - KMS キーの読み取り
  - 特定の S3 バケットへのオブジェクトの読み取り / 書き込み(Contrast はこれらのバケットを作成します)

関連項目

- はじめに(Contrast Serverless) (641ページ)

Contrast のパフォーマンスとリソース消費について

適切な設定を使用することでに、本番サーバで Contrast の影響を最小限に抑えることができます。

- **開発環境**：Contrast Assess はオンにして、Contrast Protect はオフにします。この設定により、開発時にアプリケーションのセキュリティの状況を的確に把握することができます。パフォーマンスよりも、開発者がセキュリティ上の欠陥を特定することに焦点を当てて、詳細な状況を把握する必要があります。
- **テスト環境**：プロジェクトの必要性に応じて、Contrast Assess または Contrast Protect を有効にします。プロジェクト全体の目標に合わせて、バランスを取ります。
  - 開発時にテストがほとんど実施されていない場合、Contrast Assess を活用して、アプリケーションを使用しながら脆弱性を見つける必要があります。
  - パフォーマンスを評価する場合は、Contrast Assess をオフにして、Protect のみを有効にしてください。パフォーマンスを優先しながらも、修正処置が必要な場合にはコードレベルの情報を取得するという修正対応が可能になります。
- **本番環境**：Contrast Protect のみをオンにします。この設定により、パフォーマンスを優先しながらコンテキストに応じた防御が可能になります。

関連項目

Protect 使用時のエージェントのパフォーマンス (515ページ)

.NET Framework エージェントのシステム要件 (168ページ)

.NET Core エージェントのシステム要件 (226ページ)

Node.js エージェントのシステム要件 (288ページ)

PHP エージェントのシステム要件 (342ページ)

Python エージェントのシステム要件 (359ページ)

Ruby エージェントのシステム要件 (415ページ)

Community Edition (CE)

Community Edition では、Contrast の製品(Assess、SCA、Protect)をほぼ完全に利用することができます。インタラクティブアプリケーションセキュリティテスト(IAST)、ソフトウェアコンポジション解析(SCA)、ランタイムアプリケーション自己保護(RASP)を無料でお試し頂けます。

利用を開始するには、Community Edition のアカウントに登録して、Contrast エージェントをインストールします。詳細については、Community Edition に関するブログを参照ください。

注記

Community Edition で利用できるアプリケーションは **1 つ**で、Java、.NET Core、Node.js アプリケーションのいずれかになります。

Community Edition の機能

Community Edition で提供する機能：

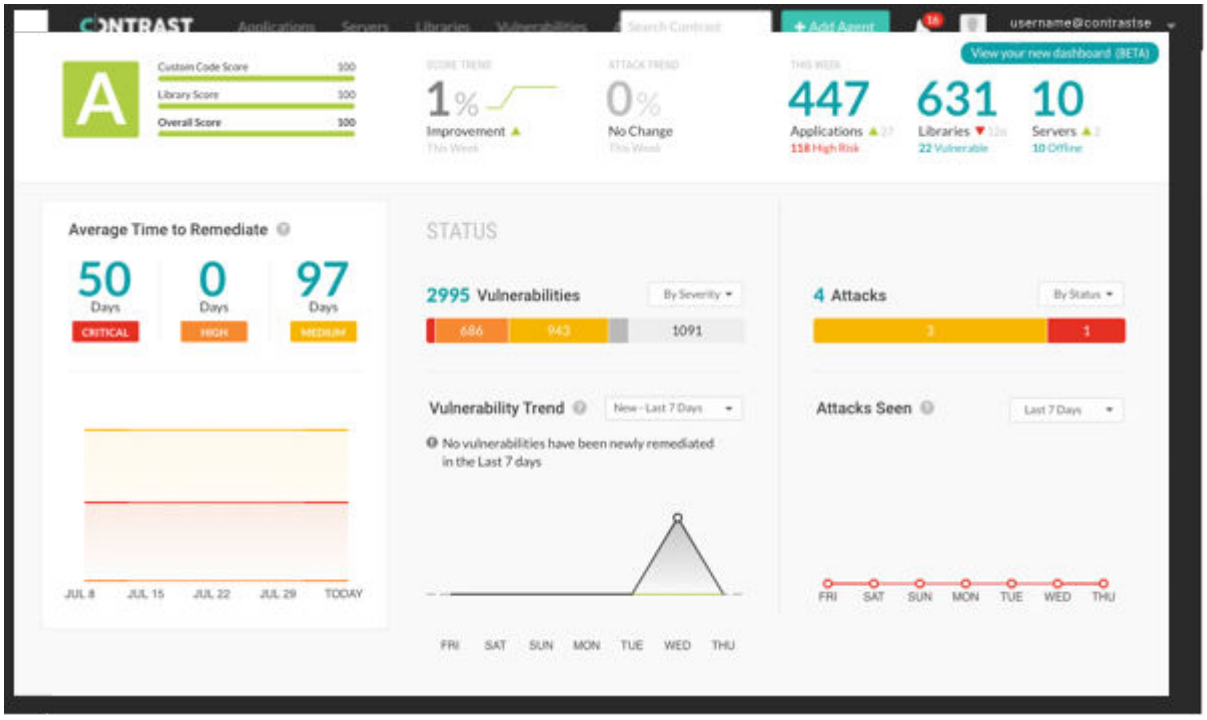
- Contrast Assess：実際に問題となるカスタムコードに起因する脆弱性を検出し、開発者は脆弱性の修正に焦点を当てることができます。
- Contrast SCA：オープンソースおよびサードパーティのライブラリがもたらす脆弱性を高度に可視化し、セキュリティリスクの管理が可能になります。

Contrast SCA は、オープンソースセキュリティまたはソフトウェアコンポジション解析(SCA)のソリューションです。

- Contrast Protect : 自作のコードやオープンソースのライブラリに脆弱性が残っている場合でも、アプリケーション内に組み込んだエージェントのセンサーが、アプリケーションへの攻撃を監視し、自動的にブロックします。

Community Edition ポータルサイト

Contrast を使用する際の Community Edition ポータルサイトは、以下のような画面になります。



次の手順

- .NET Core エージェントのインストール (227ページ)
- Java エージェントのインストール (77ページ)
- Node.js エージェントのインストール (289ページ)

デベロッパーガイド

Contrast プラットフォーム

最初のステップは、Contrast プラットフォームのテクノロジを理解し、適切な解析戦略を選択することです。

開発中のコード解析

- **Contrast Scan**
  Contrast Scan は、静的アプリケーションセキュリティテスト(SAST)ツールで、コードをすぐにスキャンして、開発の初期段階で脆弱性を特定できます。
  - **Contrast Scan を使う理由は？**
    Contrast Scan は、精度を落とすことなく、非常に早いスピードでスキャンができます。わずか数分で、数クリックするだけでスキャンを開始できます。さらに、ローカルスキャンエンジンを使用すれば、コンテンツを Contrast プラットフォームに直接アップロードしないようにすることもできます。Contrast Scan は、Angular、React、Vue.js ベースのアプリケーションなどのクライアントサイドのコードに適しています。
- **Contrast Serverless**
  FaaS(Function as a Service)サービスであるサーバレスの場合は、Contrast Serverless を使用して次のような方法で保護ができます。
  - 静的解析や動的解析を行い、脆弱性を検出します。
  - オープンソースライブラリに対して SCA 解析を行います。
  - サーバレス関数を解析して、関数の機能に必要な最小権限の設定を判定し、攻撃者の侵入経路を遮断します。

  AWS Fargate などのサーバレスで提供されるサービスには、Contrast Assess を使用できます。Azure Functions では、Contrast Serverless と Contrast Assess の両方を使用できます。Contrast CLI を使用すれば、機能の一部を利用することができ、パイプラインでのインテグレーションも可能です。ただし、Contrast Serverless の主な使用形態としては、数回のクリックと数分程度の作業だけで、クラウドプロバイダのアカウント内の全ての関数が認識されて保護されることです。
  - **Contrast Serverless を使う理由は？**
    - 関数を包括的に一覧で参照することができ、それらの関数を攻撃から守ることができます。
    - サーバレス関数のスキャンによるメリットを得るために、追加の時間をかけたり DevOps パイプラインを再構築する必要がありません。
    - 無効なデータを調べるための追加のオーバーヘッドは必要ありません。
    - Contrast Serverless によって、適切なポリシーを選択できるようになり、コードの安全性をより高めることができます。

オープンソースライブラリの解析

Contrast SCA

以前より Contrast Assess の機能と一緒にソフトウェアコンポジション解析(SCA)機能を提供してきましたが、Contrast SCA を使用することによって、コマンドラインインターフェイス(CLI)や GitHub との連携により、サードパーティの依存関係(主にオープンソース)の脆弱性の静的検出や、プロジェクト単位でのアプリの一括登録もできるようになりました。

他の優れた(多くの場合が無料か安価な)SCA ツールではなく、SCA に Contrast を使う理由は？

Contrast SCA を使用すれば、重要な問題だけに集中できます。Contrast のランタイム SCA は、アプリケーションの依存関係のマニフェストに脆弱なライブラリの脆弱なバージョンが指定されているかどうかを判別するだけでなく、実際に実行時にどのライブラリがどの程度呼び出されているかを判別できる独自の機能があります。この機能により、実行時に呼び出されていない 70%の優先度を下げることができます。また、Contrast SCA では、マニフェストには記載されていないが環境によって実行時に組み

込まれるライブラリも検出できます。これは、ほとんどの無償の SCA ソリューションなどの、単独の静的 SCA ソリューションでカバーされていない機能です。

実行時のコード解析

Contrast Assess

Contrast Assess は、Contrast プラットフォームにおける *インタラクティブアプリケーションセキュリティテスト*(IAST)の部分です。IAST ツールの核となるのは、アプリケーションコードに組み込むソフトウェアライブラリである、センサーモジュールです。このセンサーモジュールが、アプリケーションでインタラクティブなテストを実行中に、アプリケーションの動きを追跡します。IAST は、 *静的アプリケーションセキュリティテスト*(SAST)ツールがコンパイルやアプリケーション実行前に解析を行うように、アプリケーションの実行時にコードを解析して脆弱性を検出します。そして、*動的アプリケーションセキュリティテスト*(DAST)ツールのように、実行時の動きも解析します。また、実行時の *ソフトウェアコンポジション解析*(SCA)機能のコレクターとしての役割も果たします。つまり、Contrast Assess は 4 つのツールが 1 つになったものと言うことができます。

Contrast Assess を使う理由は？

Contrast Assess を使用することで、他の SAST ツールと比較して過検知がわずかになります。その一方で、SAST と DAST の両ツールでのスキャン待ちの時間を増やすことなく、最大で 2 倍の脆弱性(真陽性)が検出されるようになります。これは、Contrast Assess を使えば、ユーザや自動化された QA テストによるアプリケーションの各操作によって、実行中のコードのセキュリティに関する貴重なテレメトリが生成されるからです。このような情報が得られるため、IAST は開発サイクルの早い段階に追加できる、最もシンプルで手間のかからないセキュリティプロセスといえます。IAST なら、現在のプロセスを変更する必要はなく、リリーススケジュールを遅らせることもないからです。

本番ビルドの保護

Contrast Protect

Contrast Protect とは、*ランタイムアプリケーション自己保護*(RASP)ツールです。攻撃が成功する可能性のあるトラフィックを、Contrast Assess と同じテクノロジを使用してブロックします。

Contrast Protect を本番にリリースするビルドに使う理由は？

Contrast Protect によって、ゼロデイ(zero-day)攻撃に対する、マイナスデイ(negative-day)の保護が可能になります。Contrast Protect は、3 年前の悪名高い Log4Shell のリモートコード実行の攻撃をブロックすることができました。活発に開発が行われるアプリケーションの場合、新たなゼロデイ攻撃発生時の深夜の急対応などを回避して、アップグレードする時間を確保することができます。

メンテナンス状態のアプリケーションの場合は、長期的な保護として機能し、新しいアプリケーションに集中することができます。また、開発者にとって脅威に関する実用的な情報も提供されます。これは、攻撃が発生していることを知らせる(セキュリティ担当から得られるかどうかわからない情報)だけでなく、ブロックされた攻撃に使用されたコードパスも表示されるため、攻撃者を簡単に排除することができます。

次の手順

解析パスを確認する (33ページ)

Contrast の解析パス

Contrast には、開発ワークフローにセキュアコード解析を組み込むための方法が複数あります。

本項では、Contrast の解析を開発ワークフローに組み込むための推奨パスを紹介します。環境によっては、異なるワークフローが必要になる場合があります。

開発中の解析パス

Contrast Scan による静的スキャン	➡	以下のいずれかを使用： • Contrast CLI (36ページ) • Contrast GitHub Action (39ページ) • Contrast Web インターフェイス (40ページ)	➡	以下から結果を取得： • SARIF ファイル (41ページ) • Contrast CLI (41ページ) • Contrast Web インターフェイス (42ページ)	➡	CI/CD システムに統合 (44ページ)
Contrast Serverless で関数を静的/リアルタイムにスキャン	➡	以下のいずれかを使用： • Contrast CLI (37ページ) • Contrast Web インターフェイス (40ページ) (スキャン設定のみ)	➡	以下から結果を取得： • Contrast CLI (41ページ) • Contrast Web インターフェイス (42ページ)	➡	CI/CD システムに統合 (44ページ)

オープンソースライブラリの解析パス

Contrast SCA による静的スキャン	➡	以下のいずれかを使用： • Contrast CLI (35ページ) • GitHub アプリ	➡	以下から結果を取得： • Contrast CLI (41ページ) • Contrast Web インターフェイス (42ページ) • ワークフローでのインテグレーション (736ページ)	➡	CI/CD システムに統合 (44ページ)
Contrast SCA によるランタイムスキャン	➡	以下を使用： Contrast エージェントの組込み (39ページ)	➡	以下から結果を取得： • Contrast Web インターフェイス (42ページ) • ワークフローでのインテグレーション (736ページ)	➡	CI/CD システムに統合 (44ページ)

実行時の解析パス

Contrast Assess でアプリケーションの脆弱性を検出	➡	以下のいずれかを使用： • Contrast エージェントの組込み (39ページ) • Contrast CLI (37ページ)	➡	以下から結果を取得： • Contrast CLI (41ページ) • Contrast Web インターフェイス (42ページ) • IDE 統合 (41ページ) • ワークフローの統合 (736ページ)	➡	CI/CD システムに統合 (44ページ)

本番ビルドの防御パス

Contrast Protect で本番ビルドを保護	➡	以下を使用： Contrast エージェントの組込み (43ページ) *[en] Contrast GitHub action (39ページ)*	➡	以下から結果を取得： • Contrast Web インターフェイス (42ページ) • SIEM ツール (735ページ)		

GitHub Action

Contrast では、様々な解析タスクを簡素化する GitHub Action を提供しています。例えば、以下のように便利な GitHub Action があります。

GitHub Action	説明
Contrast Verify	この GitHub Action は、Contrast を使用するアプリケーションがジョブ結果ポリシーや脆弱性のしきい値に違反しているかどうかを判定することによって、アプリケーションを検証します。
Contrast Security SCA	この GitHub Action は、Contrast を使用して、コード内の脆弱な依存関係を検出します。
Contrast Security EKS Build Deploy	この GitHub Action は、Contrast Java エージェントを使用して、Java アプリケーションを Amazon Elastic Kubernetes Service(EKS)にビルドしてデプロイするものです。
Contrast Scan Analyze	この GitHub Action は、Contrast Scan を使用して、コードの脆弱性を検出します。
Contrast Security AKS Build Deploy	この GitHub Action は、Contrast Java エージェントを使用して、Java アプリケーションを Azure Kubernetes Service(AKS)にビルドしてデプロイするものです。
Contrast Security Azure Spring Cloud Deploy	この GitHub Action は、Contrast Java エージェントを使用して、Java アプリケーションを Azure Spring Cloud の PaaS 環境にデプロイするものです。

GitHub Marketplace に、最新の Contrast の GitHub Action があります。

コードの解析

コードを解析する方法を決めたら、オープンソースとソースコードのテストを始めましょう。

解析対象	解析方法	参照先
オープンソースライブラリ	Contrast CLI	CLI を使用してオープンソースライブラリを解析 (35ページ)(静的解析)
	GitHub アプリ	GitHub アプリを使用してオープンソースライブラリを解析 (38ページ)(静的解析)
	Contrast エージェント組み込み	アプリケーションにエージェントを組み込んでオープンソースライブラリを解析 (39ページ) (ランタイム解析)
ソースコード	Contrast CLI	• CLI を使用する静的スキャン (36ページ) • CLI を使用するサーバレス関数のスキャン (37ページ)
	GitHub Action	GitHub Action を使用する静的スキャン (39ページ)
	Contrast エージェント組み込み	アプリケーションにエージェントを組み込んで脆弱性を検出 (39ページ)
	Contrast Web インターフェイス	• Contrast Web インターフェイスを使用する静的スキャン (40ページ) • Contrast Web インターフェイスを使用してサーバレス関数のスキャンを設定 (40ページ)

次の手順

- CLI で結果を取得 (41ページ)
- IDE インテグレーションで結果を取得 (41ページ)
- SARIF ファイルで結果を確認 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

CLI を使用するオープンソースライブラリの解析

Contrast CLI を使用すると、オープンソースライブラリの脆弱性が解析され、その結果を確認できます。

デフォルトでは、Contrast CLI の結果はローカルに保存されません。永続的なデータを保持するには、Contrast CLI の `--track` オプションを使用して、結果を Contrast Web インターフェイスに送信してください。

開始する前に

- Contrast CLI (666ページ)を学んでおくこと。
- Contrast CLI をインストール (667ページ)しておくこと。

手順

1. ターミナルウィンドウで以下のコマンドを実行し、Contrast の認証情報をローカルに保存します。

```
contrast auth
--api-key <ContrastAPIKey>
--authorization <ContrastAuthorizationHeader>
--host <YourHosDomain>
--organization id <ContrastOrganizationID>
```

   Contrast Web インターフェイスにログインして、**ユーザメニュー > ユーザの設定**にアクセスして、Contrast API キー、認証ヘッダー、組織 ID を取得します。
2. ターミナルウィンドウで以下のコマンドを使用して、脆弱なライブラリを検索します。

```
contrast audit [option]
```

   - `--track` オプションを使用すると、Contrast Web インターフェイスの ライブラリ (600ページ) ページの静的タブに結果が送信されます。
   - `--file` オプションを使用すると、検査するディレクトリまたはファイルを指定できます。

   Contrast CLI コマンド (674ページ)にて、`audit` コマンドで有効な全てのオプションについて説明しています。

次の手順

- CLI で結果を取得 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

CLI を使用する静的スキャン

Contrast Web インターフェイスを使用する代わりに、Contrast CLI を使用してコードをスキャンすることができます。

開始する前に

- Contrast CLI (666ページ)を学んでおくこと。
- Contrast CLI をインストール (667ページ)しておくこと。

手順

1. ターミナルウィンドウで以下のコマンドを実行し、Contrast の認証情報をローカルに保存します。

```
contrast auth
--api-key <ContrastAPIKey>
--authorization <ContrastAuthorizationHeader>
--host <YourHosDomain>
--organization id <ContrastOrganizationID>
```

   Contrast Web インターフェイスにログインして、**ユーザメニュー > ユーザの設定**にアクセスして、Contrast API キー、認証ヘッダー、組織 ID を取得します。
2. ターミナルウィンドウで以下のコマンドを実行すると、パッケージがアップロードされてスキャンされます。

```
contrast scan --file <FileName>
```

Contrast CLI コマンド (679ページ)にて、`scan` コマンドで有効な全てのオプションについて説明しています。

次の手順

- CLI で結果を取得 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

CLI を使用するサーバレス関数のスキャン

Contrast Web インターフェイスを使用する代わりに、Contrast CLI を使用してサーバレスの関数をスキャンすることができます。

開始する前に

- Contrast CLI (666ページ)を学んでおくこと。
- Contrast CLI をインストール (667ページ)しておくこと。

手順

1. ターミナルウィンドウで以下のコマンドを実行し、Contrast の認証情報をローカルに保存します。

   ```
   contrast auth
   --api-key <ContrastAPIKey>
   --authorization <ContrastAuthorizationHeader>
   --host <YourHosDomain>
   --organization id <ContrastOrganizationID>
   ```

   Contrast Web インターフェイスにログインして、**ユーザメニュー > ユーザの設定**にアクセスして、Contrast API キー、認証ヘッダー、組織 ID を取得します。
2. ターミナルウィンドウで以下のコマンドを使用すれば、脆弱性が検出されます。

   ```
   contrast lambda --function-name <function> [options]
   ```

   - `--json` を指定すると、レスポンスが JSON 形式で返ります。
   - `--verbose` を指定すると、ターミナルウィンドウに詳細情報が返ります。
   - Contrast CLI コマンド (681ページ)にて、`lambda` コマンドで有効な全てのオプションについて説明しています。

次の手順

- CLI で結果を取得 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

CLI を使用する脆弱性の検出

Assess CLI によって、Contrast Assess を使用してリアルタイムに脆弱性を検出できます。

Assess CLI は、以下の Contrast エージェントでサポートされます。

- Java
- Node.js
- .NET
- Python
- Ruby
- Go

開始する前に

- Contrast CLI (666ページ)を学んでおくこと。
- Contrast CLI をインストール (667ページ)しておくこと。
- エージェントでサポートされているテクノロジを確認して、お使いのアプリケーションで Assess CLI が使用できることを確認してください。

手順

1. Contrast エージェントをインストールまたはアップデートします。
2. ターミナルウィンドウで以下のコマンドを実行します。

   ```
   conrast assess
   ```

   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。ファイルのデフォルトの場所は次のとおりです。
   - **MacOS および Linux**： `/etc/contrast/contrast_security.yaml`
   - **Windows**： `%ProgramData%\Contrast\contrast_security.yaml`

   別のファイルの場所を指定するには、`config-path` オプションを使用します。`assess` コマンドで有効な全てのオプションについては、Contrast CLI コマンド (674ページ)にて説明しています。
3. IDE または 2 つ目のターミナルウィンドウでアプリケーションを実行します。
4. アプリケーションを疎通します。
5. Assess CLI コマンドを入力したターミナルウィンドウに検出結果が表示されます。

次の手順

CLI で結果を取得 (41ページ)

関連項目

Java エージェントで Assess CLI を使用する (669ページ)

Node.js エージェントで Assess CLI を使用する (671ページ)

.NET エージェントで Assess CLI を使用する (670ページ)

Python エージェントで Assess CLI を使用する (671ページ)

Ruby エージェントで Assess CLI を使用する (672ページ)

Go エージェントで Assess CLI を使用する (673ページ)

GitHub アプリを使用するオープンソースライブラリの解析

Contrast GitHub アプリを使用すると、GitHub リポジトリを Contrast に接続することができます。Contrast と接続したら、選択した GitHub リポジトリのオープンソースライブラリが Contrast でスキャンされて、脆弱性が特定されます。

開始する前に

- GitHub アプリを接続するために、Contrast アカウントのサブドメインとホストが必要です(例、`app.contrastsecurity.com`)。

手順

1. Contrast Web インターフェイスにログインして、ナビゲーションバーで**新規登録**を選択します。
2. 「リポジトリ」カードタブを選択したら、**GitHub に接続**を選択します。
3. 画面が表示されたら、GitHub でアプリをインストールする場所を指定します。
4. Contrast Web インターフェイスで最終的な承認が完了するまで、表示された手順に従います。プロジェクトの一覧に、GitHub のリポジトリが表示され始めます。

5. **リポジトリを追加**を選択することで、いつでもリポジトリを追加できます。

次の手順

Contrast Web インタフェースで結果を確認 (42ページ)

GitHub Action を使用する静的スキャン

GitHub Action「Contrast Scan Analyze」は、プルリクエスト(PR)のコードスキャン解析を、宛先ブランチでの前回のコードスキャン解析と比較するものです。

開始する前に

- Contrast Web インターフェイスの**ユーザメニュー > ユーザの設定**から、以下の情報が必要です。
  - あなたの API キー
  - 認証ヘッダー
  - 組織 ID

手順

1. Contrast リポジトリの GitHub Action にアクセスします。
2. アクションを設定します。

次の手順

- SARIF ファイルで結果を取得 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

アプリケーションにエージェントを組み込んでオープンソースライブラリを解析

アプリケーションに Contrast エージェントをインストールすると、アプリケーションに含まれているオープンソースライブラリが識別されます。ライブラリの脆弱性が特定され、そのライブラリがランタイムに使用されているかどうかも確認できます。

手順

1. アプリケーションで使用している言語に合わせて、その言語の Contrast エージェントをインストールして設定 (46ページ)します。
   Contrast エージェントは、パッケージマネージャやリポジトリからダウンロードできます。
   Contrast エージェントは、直接インストールすることもできますし、インテグレーション (726ページ)を使用して Contrast を他のツールに組み込むこともできます。
2. アプリケーションを実行して、Contrast が機能していることを確認します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API や CLI コマンドをいくつか送信してみてください。

次の手順

- Contrast Web インターフェイスで結果を確認 (42ページ)
- IDE で結果を確認 (41ページ)

アプリケーションにエージェントを組み込んで脆弱性を検出

アプリケーションの脆弱性を見つけるには、Contrast エージェントを使用してアプリケーションを検査します。IDE で Contrast の拡張機能やプラグインを使用して、直接 IDE で検出結果を確認して脆弱性を修正することもできます。

基本手順

1. アプリケーションがあるローカルディレクトリに、Contrast エージェントをインストール (46ページ)します。
2. YAML ファイルか環境変数を使用し、Contrast の接続データなどを指定してエージェントを設定 (60ページ)します。
   Contrast エージェント設定エディタ (64ページ)を使用すると、エージェントを簡単に設定できます。
3. アプリケーションを起動して、ルートを疎通します。

Contrast IDE プラグインを使用する場合の手順

1. アプリケーションがあるローカルディレクトリに、Contrast エージェントをインストールします。
2. YAML ファイルか環境変数を使用し、Contrast の接続データなどを指定してエージェントを設定します。
   Contrast エージェント設定エディタ (64ページ)を使用すると、エージェントを簡単に設定できます。
3. アプリケーションを起動します。
4. 必要な接続情報を指定して、Contrast IDE プラグイン (726ページ)を設定します。
   この場合、Contrast Web インターフェイスの**ユーザメニュー > ユーザの設定 > プロファイル**より、個人のキーや API 情報が必要です。
5. アプリケーションでルートを疎通します。

次の手順

IDE で脆弱性を確認 (41ページ)

Contrast Web インターフェイスを使用してサーバレス関数のスキャンを設定

Contrast Web インターフェイスを使用して、サーバレス関数のスキャンを設定できます。

手順

1. Contrast Web インターフェイスにログインして、アカウント(AWS (641ページ)か Azure (647ページ))に接続します。
2. スキャンを設定します。
   a. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
   b. スキャンを行いたい関数が含まれるアカウントを選択します。
   c. スキャンする関数を選択します。 (648ページ)

次の手順

Contrast Web インターフェイスで結果を確認 (42ページ)

Contrast Web インターフェイスを使用する静的スキャン

Contrast Web インターフェイスで、コードを簡単にスキャンすることができます。

開始する前に

- Contrast Scan のサポート対象テクノロジ (554ページ)を確認すること。
- スキャンするパッケージの準備 (556ページ)について理解すること。
- スキャンしたいアーティファクトを確認すること。
- スキャンプロジェクトを作成 (557ページ)すること。

手順

1. Contrast Web インターフェイスにログインします。
2. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
3. スキャンするファイルを追加するスキャンプロジェクトを選択します。
4. スキャンを開始します。
   a. **新規スキャン**を選択します。
   b. ファイルをアップロードします。

次の手順

Contrast Web インターフェイスで結果を確認 (42ページ)

結果の確認

オープンソースライブラリやコードを解析した後に結果を確認するには、複数の方法があります。

- Contrast CLI を使用して結果を取得 (41ページ)
- IDE で結果を取得 (41ページ)
- SARIF ファイルで結果を取得 (41ページ)
- Contrast Web インターフェイスで結果を確認 (42ページ)

CLI で結果を取得

Contrast CLI コマンドを実行すると、ターミナルウィンドウに結果が返されます。これらの結果は戻り値として表示されるだけで、保存されません。使用するコマンドによっては、結果を Contrast Web インターフェイスに送信したり、結果を SARIF ファイルでダウンロードしたりできます。

手順

1. `audit` コマンドを使用する場合、`--track` オプションを指定すれば、Contrast Web インターフェイスのライブラリページの静的タブに結果を送信できます。
2. `scan` コマンドを使用する場合、`--save` オプションを指定すれば、結果を SARIF ファイルにして、現在の作業ディレクトリに、ダウンロードできます。

IDE インテグレーションで結果を取得

Contrast の IDE プラグインを使用すれば、ご利用の IDE 環境にて脆弱性の情報を直接表示できます。

Contrast がサポートしている IDE プラグイン：

- Eclipse
- IntelliJ
- Visual Studio
- Visual Studio Code
- Visual Studio for Mac

開始する前に

Contrast エージェントをインストールして設定 (46ページ)し、アプリケーションにエージェントを組み込みます。

手順

1. Contrast IDE プラグイン (733ページ)を検索します。
2. そのプラグインの設定手順に従います。

SARIF ファイルで結果を取得

静的スキャンの結果をターミナルウィンドウではなく、SARIF ファイルで取得するよう指定できます(Contrast CLI を使用している場合)。また、Contrast Web インターフェイスから SARIF ファイルをダウンロードすることもできます。

手順

1. Contrast CLI を使用して静的スキャンを行う場合、以下のコマンドオプションを指定すると、結果を SARIF に保存できます。

   ```
   contrast scan --save
   ```

   このコマンドによって、`results.sarif` このファイルは、任意のテキストエディタで表示できます。
2. Contrast Web インターフェイスを使用する場合は、結果を SARIF(または CSV)ファイルにダウンロードできます。
   - Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
   - スキャンプロジェクトの一覧から、プロジェクトを選択します。
   - スキャンの行の最後に表示されているダウンロードアイコン(⬇)を選択します。結果は、スキャン完了後の 5 日間ダウンロード可能です。
3. 静的スキャンに GitHub Action を使用しており、リポジトリの Security タブで結果を表示するためには、以下の GitHub Action を設定に含めてください。

   ```
       - name: Upload SARIF file
         uses: github/codeql-action/upload-sarif@v2
         with:
           sarif_file: results.sarif
   ```

   SARIF ファイル名は、`results.sarif` にする必要があります。

Contrast Web インターフェイスで結果を取得

どのような方法で開発ワークフローに Contrast を組み込んでも、ほとんどの場合、Contrast Web インターフェイスでコード解析の結果を確認できます。

開始する前に

- Contrast Web インターフェイスにログインします。

手順

1. オープンソースライブラリの解析の結果を表示するには、Contrast Web インターフェイスのナビゲーションバーで、**ライブラリ**を選択します。
   このビューには、すべてのプロジェクト(静的)とアプリケーション(実行時)における全ライブラリが表示されます。また、特定のアプリケーションのライブラリは、そのアプリケーションのライブラリタブで表示できます。
   - ライブラリの一覧で、特定の脆弱性に関する情報を表示するには、脆弱性バーの名前かセクションを選択します。
2. Contrast CLI を使用したマニフェストファイルの解析後か、GitHub との接続後に、オープンソースライブラリの結果を表示するには、Contrast Web インターフェイスのナビゲーションバーで**プロジェクト**を選択します。
   - プロジェクトの一覧で、特定の脆弱性に関する情報を表示するには、脆弱性バーの名前かセクションを選択します。
3. アプリケーションの脆弱性情報を表示するには、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   - アプリケーションの一覧で、特定の脆弱性に関する情報を表示するには、脆弱性バーのセクションを選択します。

4. 静的スキャンの情報を表示するには、Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
   a. スキャンの一覧から、スキャンプロジェクトを選択します。
   b. 脆弱性に関する詳細を表示するには、**脆弱性**タブを選択します。
5. サーバレス関数のスキャン情報を表示するには、Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
   a. 脆弱性に関する詳細を表示するには、**結果**タブを選択します。
   b. 特定の関数の詳細を表示するには、**結果**の一覧で関数を選択します。

攻撃の監視・ブロック

Contrast Protect では、Contrast エージェントを設定して、本番環境のアプリケーションに対する悪意のある攻撃を検知し、管理することができます。

開発者としては、Contrast Protect のルールを使用して攻撃を監視・ブロックするように Contrast エージェントを設定 (43ページ)することができます。監視・ブロックなどの結果は、Contrast エージェントによって Contrast Web インターフェイス (43ページ)に報告されます。

アプリケーションにエージェントを組み込んで Protect を使用

Contrast Protect をオンにすると、Contrast エージェントを使用して、本番環境のアプリケーションに対する攻撃を検知、監視、ブロックすることができます。

開始する前に

- スーパー管理者(SuperAdmin)に問い合わせて、組織で Contrast Protect が有効になっていることを確認すること。
- 組織管理者(Admin)に問い合わせて、Protect の情報を表示できる権限があることを確認すること。

手順

1. アプリケーションで使用している言語に合わせて、その言語の Contrast エージェントをインストールして設定 (46ページ)します。
   Contrast エージェントは、パッケージマネージャやリポジトリからダウンロードできます。
   Contrast エージェントは、直接インストールすることもできますし、インテグレーション (726ページ)を使用して Contrast を他のツールに組み込むこともできます。
   Contrast エージェントの設定ファイル(YAML ファイル)で：
   a. Contrast Protect をオンにします。
   b. Protect ルールのモード(監視、ブロック、オフ)を指定します。
2. アプリケーションを実行して、Contrast が機能していることを確認します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API や CLI コマンドをいくつか送信してみてください。

次の手順

Contrast Web インタフェースで攻撃情報を確認 (43ページ)

Contrast Web インタフェースで攻撃情報を確認

本番環境で発生した攻撃の情報は、Contrast Web インターフェイスに表示されます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで、**攻撃**を選択します。
2. 各タブを確認すると、アプリケーションに影響を与えた攻撃に関する情報が参照できます。

継続的インテグレーション/継続的デリバリーのためのオプション

Contrast には、継続的インテグレーション/継続的デリバリー(CI/CD)パイプラインに Contrast を組み込むためのオプションがあります。CI/CD の自動化を担当していない場合は、これらのオプションについて DevOps 担当と相談してください。

オプション	説明
Azure Pipelines の拡張機能 (744ページ)	Azure Pipelines の拡張機能を使用すれば、タスクとリリースゲートを設定して、Contrast から報告される脆弱性情報に基づいてタスクや検証を失敗させることができます。
Bamboo (748ページ)	Contrast Bamboo プラグインを使用して、Contrast に接続するためのプロファイルを設定すれば、脆弱性のしきい値に対してビルドを検証することができます。
Circle CI	Contrast の Circle CI Orb によって、Contrast API にクエリを発行し、アプリケーションで脆弱性が検出されたかを確認できます。設定したしきい値を超える脆弱性が検出された場合に、ビルドを失敗させることができます。
GitLab	GitLab のパイプライン内にステージを作成して、Contrast から報告される検出結果に基づいて、セキュリティゲートとして機能させることができます。GitLab の変数を設定して、ステージを失敗させるトリガーとなる脆弱性を指定できます。
Gradle (756ページ)	Contrast の Gradle プラグインによって、ビルドに `Contrast.jar` を組み込むことができます。Contrast への認証、最新の Java エージェントのダウンロード、ビルドの検証を行うことができます。
Jenkins (758ページ)	Contrast の Jenkins プラグインを使用すると、Jenkins のパイプラインにアプリケーションセキュリティゲートを追加できます。セキュリティゲートとして、脆弱なアプリケーションの Jenkins ジョブを `FAILURE()` や `UNSTABLE()` などのビルド結果で失敗させる基準を指定できます。
Maven (772ページ)	Contrast Maven プラグインにより、Contrast Assess や Contrast Scan をプロジェクトの Maven ビルドに組み込むことができます。

エージェント

Contrast エージェントは、アプリケーションからセキュリティに関連するデータを収集し、そのデータを解析して、必要に応じて検出結果を Contrast に報告します。特定の条件で、Contrast エージェントは悪用を防止したり、セキュリティ防御を有効にしたりするために、アプリケーション内で処理を実行することもあります。

Contrast エージェントをインストール (46ページ)することで、コードスキャン、ライブラリスキャン、アプリケーションの解析、設定ファイルのスキャンなど、さまざまなセキュリティ計測技術を使用してセキュリティに関連する情報を収集できます。このさまざまなセキュリティ計測技術となって情報を収集するのが、センサーです。

センサーによって、アプリケーション内からスナップショットで直接情報が取得されイベントが生成されます。例えば、センサーは、受信した HTTP パラメータやデータベースに対して実行される SQL クエリの詳細を取得します。センサーによっては、必要に応じて、防御機能を強化したり悪意のある攻撃をブロックしたりするために、脆弱性を迂回させるセキュリティ例外をスローする処理などを行う場合もあります。

センサーによって生成されたイベントは全て、エージェントによる追跡と解析対象として Contrast サーバに報告されます。解析エンジンはアプリケーションのあらゆるコードからイベントを受け取り (700ページ)、それらのイベントによって次第にトレースが構築されていきます。解析エンジンは、これらのトレースを監視して、Contrast ルールに違反する動作パターンを探します。

例えば、解析エンジンは以下のようなデータフローを確認します。

- 受信した HTTP リクエストのパラメータに関するイベント
- 次に、そのパラメータが SQL クエリに追加されていることを示す別のイベント
- 最後に、そのクエリがデータベースに送信されていることを示す別のイベント

適切な防御策(エスケープ処理やパラメータ化)がないデータフローを解析エンジンが検出した場合は、そのトレースが SQL インジェクションの Contrast ルールに一致すると認識され、Contrast サーバに報告されます。解析の大部分はエージェント内でローカルに行われるため、Contrast の拡張性とパフォーマンスが向上します。

検査対象のアプリケーションの言語と一致するエージェントを使用してください。

- Java (75ページ)エージェントは、コンテナで実行される Java の Web アプリケーションおよび Web API を計測します。
- .NET Framework (166ページ)エージェントは、IIS で実行される.NET Framework の Web アプリケーションおよび API を計測します。
- .NET Core (224ページ)エージェントは、.NET Core ランタイムで実行されるアプリケーションおよび API を計測します。
- Node.js (284ページ)エージェントは、Node.js の Web アプリケーションおよび API を計測します。
- PHP (341ページ)エージェントは、PHP の Web アプリケーションを実行時に解析し、ライブラリの使用状況や脆弱性の検出を行います。
- Python (359ページ)エージェントは、Django、Flask および Pyramid の Web アプリケーションを計測します。
- Ruby (415ページ)エージェントは、Ruby on Rails の Web アプリケーションを計測します。
- Go (476ページ)エージェントは、ライブラリのサポートと脆弱性報告のために、Go の Web アプリケーションを計測します。

注記

Contrast エージェントは、リリース後 1 年間サポートされます。古いエージェントは、引き続き機能し互換性を維持できる可能性もありますが、完全にサポート対象外になります。

Contrast では、バグの修正の適用や新機能の開発は、エージェントの最新バージョンが対象となります。コードの修正は、古いバージョンにはバックポートされません。バグに対して回避策が提供されている場合でも、問題を解決するには、その問題が修正されたリリースに更新してください。

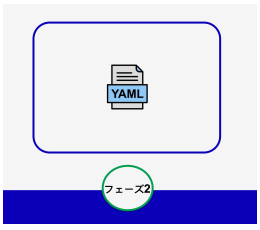
エージェントのインストール

Contrast ではエージェントを使用して、コードの脆弱性をモニターするセンサーを配置します。エージェントによって、開発環境の脆弱性を検査し、実行時の本番環境では攻撃を検知することができます。

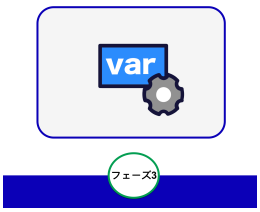
アプリケーションを実行すると、エージェントは情報(HTTP リクエスト、データフロー、バックエンド接続、ライブラリの依存関係など)を解析し、脆弱性や攻撃を Contrast サーバに送信します。これらの情報は Contrast Web インターフェイスで確認することができ、優先順位付けを行い、すぐに対策を取ることができます。

本項は、アプリケーションで Contrast をすぐに使用できるようになり、Contrast がどのように機能するかを確認するための参考にしてください。

下記のオプションからエージェントのインストール方法を決定します。(47ページ)



Contrast サーバへの認証をエージェントに設定します。(60ページ)



必要に応じてその他の設定を指定します。(67ページ)

通常通りにアプリケーションを使用します。Contrast Web インタフェースでアプリケーション (526ページ)を選択すると、アプリケーションの名前が表示されているはずです。

Contrast エージェントをアプリケーションに組み込んで検査を行うには、いくつかのフェーズに分けることができますが、本項のガイドを参考にすると、すぐにアプリケーション上で Contrast を起動して、動作させることができます。

- アプリケーションサーバや Web サーバ上
- ビルドパイプラインやコンテナ内
- 開発、テスト(QA)、本番などの環境

一度、仕組みを理解したら、後は自分のニーズに合わせて調整することができます。そのための方法がたくさんあります。Contrast ドキュメント (45ページ)を参照して、利用する状況に合わせて Contrast を調整してください。

ヒント
今後の導入にあたっては、利用するビルドツールやデプロイパイプライン、セキュリティ目標、Contrast を使用する環境などの検討が必要な場合があります。ご利用状況に合わせて、インテグレーションによって Contrast をインストールする方法 (726ページ)も参照ください。

Java

エージェントのインストールと設定のワークフロー (49ページ)を参照ください。

実行可能な JAR のインストール	アプリサーバへインストール	ビルド自動化ツールと連携してインストール	コンテナにインストール	クラウドオーケストレーションサービスでインストール
JAR ファイルを使って 1 つのアプリケーションにエージェントをインストールします。 Maven Central (78ページ)、Debian (79ページ)、RPM (80ページ)の各リポジトリ使用してインストールします。	アプリケーションサーバにエージェントをインストールし、テスト/QA 環境や本番環境で動作するアプリケーションのセキュリティ検査を行います。 JBoss/Wildfly (96ページ)の場合。 Jetty (98ページ)の場合。 Tomcat (98ページ)の場合。 Weblogic (99ページ)の場合。 Websphere (100ページ)の場合。 Axis2 の場合。 Glassfish の場合。	Contrast プラグインを使用してエージェントをインストールして、インストールを自動化します。 Maven (772ページ)の場合。 Gradle (756ページ)の場合。 Bamboo (748ページ)の場合。 Azure Pipelines (744ページ)の場合。 VMware Tanzu (86ページ)の場合。	コンテナにエージェントをインストール、または Kubernetes オペレータでエージェントをインストールします。 エージェントを Docker(ベースイメージまたはアプリケーションイメージ)に追加 (81ページ)。 OpenShift の場合。 エージェントを Contrast K8s オペレータで Kubernetes Pod に追加 (498ページ)。	エージェントを Google App Engine でインストールします。 エージェントを AWS Elastic Beanstalk でインストール (91ページ)します。

また、Contrast Java エージェントを使用して、Contrast Assess や Contrast SCA で、Scala ベース (95ページ)のアプリケーションや Kotlin ベース (96ページ)のアプリケーションを解析することができます。

.NET Framework

エージェントのインストールと設定のワークフロー (52ページ)を参照ください。

インストーラによるインストール	Azure でインストール	コンテナにインストール
セルフホストのアプリケーションや IIS 上のアプリケーションに、エージェントのインストーラ (169ページ)を使用してインストールします。	Azure App Service でエージェントをインストールします。	エージェントを Azure App Service でインストール (172ページ)します。

.NET Core

エージェントのインストールと設定のワークフロー (53ページ)を参照ください。

Windows

基本のインストール	インストーラによるインストール	Azure でインストール	コンテナにインストール
基本のインストール (227ページ)で、.NET Core エージェントをインストールします。	セルフホストのアプリケーションや IIS 上のアプリケーションに、エージェントのインストーラ (234ページ)を使用してインストールします。	エージェントを Azure App Service でインストール (232ページ)します。 エージェントを Terraform でインストール (239ページ)します。	コンテナにエージェントをインストール、または Kubernetes オペレータでエージェントをインストールします。 エージェントを Docker(ベースイメージまたはアプリケーションイメージ)に追加 (236ページ)。 エージェントを Contrast K8s オペレータで Kubernetes Pod に追加 (498ページ)。

Linux

基本のインストール	コンテナにインストール
基本のインストール (227ページ)で、.NET Core エージェントをインストールします。	エージェントをコンテナイメージにインストール (236ページ)します。

Node.js

エージェントのインストールと設定のワークフロー (56ページ)を参照ください。

基本のインストール	コンテナにインストール	クラウド環境のデプロイにインテグレーションしてインストール
基本のインストール (290ページ)で、Node エージェントをインストールします。	エージェントをコンテナイメージにインストール (290ページ)します。	IBM Cloud でインストール (296ページ)します。 VMWare Tanzu でインストールします。

PHP

エージェントのインストールと設定のワークフロー (57ページ)を参照ください。

リポジトリによるインストール
Debian (342ページ)または RPM (344ページ)リポジトリを使用して、PHP エージェントをインストールします。

Python

エージェントのインストールと設定のワークフロー (58ページ)を参照ください。

Contrast ランナーによるインストール
Contrast ランナー (414ページ)を使用して Python エージェントをインストールしてアプリケーションに組み込みます。

ミドルウェアによるインストール
ミドルウェア (360ページ)(AIOHTTP、Bottle、Django、Falco、Fast API、Flask、Pyramid、Quart、WSGI)で、Python エージェントをインストールします。

Ruby

エージェントのインストールと設定のワークフロー (59ページ)を参照ください。

ミドルウェアによるインストール
ミドルウェア (417ページ)(Rails、Sinatra、または Grape)で、Ruby エージェントをインストールします。

Go

エージェントのインストールと設定のワークフロー (59ページ)を参照ください。

インストーラによるインストール
Go エージェントを Contrast インストーラでインストール (477ページ)します。

エージェント設定ファイルのダウンロード

必要な設定があらかじめ入力された YAM L 設定ファイルをダウンロードできます。

手順

1. Contrast Web インターフェイスで、**新規登録**を選択します。

2. 「アプリケーション」タイルを選択します。
3. エージェントの言語を選択します。
4. YAML ファイルをダウンロードします。

Java エージェントのインストールと設定のワークフロー

以下の各ワークフローを記載したページより、Java エージェントのインストールと設定の手順を確認できます。ご利用の環境に合わせて参照してください。

- JAR ファイルによる Java エージェント (49ページ)
- アプリケーションサーバへの Java エージェント (50ページ)
- ビルド自動化ツールで連携する Java エージェント (50ページ)
- コンテナでの Java エージェント (51ページ)

JAR ファイルを使用する Java エージェントのインストールと設定

以下のワークフローにより、実行可能な JAR ファイルで Java エージェントをインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Web アプリケーションが JAR ファイルとしてパッケージ化されていること。
- Contrast でサポートされているバージョン、フレームワーク、ツール (75ページ)を使用していること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **ダウンロード**：以下のいずれかのリポジトリから、Java エージェントをダウンロードして、インストールします。
   - Maven (78ページ)
   - Debian (79ページ)
   - RPM (80ページ)
2. **変数の設定**： エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定

- 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)

3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

アプリケーションサーバへの Java エージェントのインストールと設定

以下のワークフローにより、アプリケーションサーバに Java エージェントをインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Web アプリケーションが JAR ファイルとしてパッケージ化されていること。
- Contrast でサポートされているバージョン、フレームワーク、ツール (75ページ)を使用していること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **ダウンロード**：お使いのリポジトリから、Java エージェントの JAR ファイルをダウンロードします。
   - Maven (78ページ)
   - Debian (79ページ)
   - RPM (80ページ)
2. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
3. **アプリケーションサーバを設定**：ご利用のサーバタイプに合わせて、アプリケーションサーバを設定します。
   - Jboss/Wildfly (96ページ)
   - Jetty (98ページ)
   - Tomcat (98ページ)
   - Weblogic (99ページ)
   - WebSphere (100ページ)
   - Axis2
   - Glassfish
4. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

ビルド自動化ツールを使用する Java エージェントのインストールと設定

以下のワークフローにより、ビルド自動化ツールを使用して Java エージェントをインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Web アプリケーションが JAR ファイルとしてパッケージ化されていること。
- Contrast でサポートされているバージョン、フレームワーク、ツール (75ページ)を使用していること。
- 設定した値が有効になる優先順位 (62ページ)を理解する。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **プラグイン別に設定**：利用するプラグインタイプに合わせて、インストールおよび設定を行います。
   - Maven (772ページ)(ゴールの設定を忘れずに)
   - Gradle (756ページ)
   - Bamboo (748ページ)(脆弱性のしきい値の設定 (748ページ)を忘れずに)
   - Azure Pipelines (744ページ)(パイプラインにリリースゲートを追加するのを (745ページ)忘れずに)
   - VMware Tanzu (86ページ)
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

コンテナでの Java エージェントのインストールと設定

以下のワークフローにより、コンテナで Java エージェントをインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Web アプリケーションが JAR ファイルとしてパッケージ化されていること。
- Contrast でサポートされているバージョン、フレームワーク、ツール (75ページ)を使用していること。また、エージェントオペレータを使用する場合は、サポート対象のテクノロジ (498ページ)を使用していること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **コンテナ別の設定**：利用するコンテナタイプに合わせて、インストールおよび設定を行います。

- コンテナ(Docker など) (81ページ)
- OpenShift
- Kubernetes (498ページ)

3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

.NET Framework エージェントのインストールと設定のワークフロー

以下の各ワークフローを記載したページより、.NET Framework エージェントのインストールと設定の手順を確認できます。ご利用の環境に合わせて参照してください。

- インストーラを使用する.NET Framework エージェント (52ページ)
- Azure App Service での.NET Framework エージェント (52ページ)
- コンテナでの.NET Framework エージェント (53ページ)

インストーラを使用する.NET Framework エージェントのインストールと設定

以下のワークフローにより、インストーラを使用して.NET Framework エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (167ページ)を使用しシステム要件 (168ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **ダウンロード**：インストーラ (169ページ)をダウンロードして、エージェントをインストール (170ページ)します。
3. **設定**：エージェントを設定 (181ページ)します。
4. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Azure App Service での.NET Framework エージェントのインストールと設定

以下のワークフローにより、Azure App Service で.NET Framework エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (167ページ)を使用しシステム要件 (168ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (172ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

コンテナでの.NET Framework エージェントのインストールと設定

以下のワークフローにより、コンテナで.NET Framework エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (167ページ)を使用しシステム要件 (168ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (174ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

.NET Core エージェントのインストールと設定のワークフロー

以下の各ワークフローを記載したページより、.NET Core エージェントのインストールと設定の手順を確認できます。ご利用の環境に合わせて参照してください。

- .NET Core エージェントの基本インストール (54ページ)
- インストーラを使用する.NET Core エージェント (54ページ)

- Azure App Service で.NET Core エージェント (55ページ)
- コンテナでの.NET Core エージェント (55ページ)

.NET Core エージェントの基本のインストールと設定

以下のワークフローにより、.NET Core エージェントの基本的なインストール手順および設定手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (225ページ)を使用しシステム要件 (226ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (227ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

インストーラを使用する.NET Core エージェントのインストールと設定

以下のワークフローにより、インストーラを使用して.NET Core エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (225ページ)を使用しシステム要件 (226ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (234ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。

また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Azure App Service での.NET Core エージェントのインストールと設定

以下のワークフローにより、Azure App Service で.NET Core エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (225ページ)を使用しシステム要件 (226ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーション、セッションメタデータ)
2. **インストールと設定**：エージェントをインストール (232ページ)して設定 (247ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

コンテナでの.NET Core エージェントのインストールと設定

以下のワークフローにより、コンテナで.NET Core エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (225ページ)を使用しシステム要件 (226ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **コンテナ別の設定**：利用するコンテナタイプに合わせて、エージェントをインストールおよび設定を行います。
   - Docker(ベースイメージまたはアプリケーションイメージ)にエージェントを追加 (236ページ)
   - Kubernetes (498ページ)

3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Node.js エージェントのインストールと設定のワークフロー

以下の各ワークフローを記載したページより、Node.js エージェントのインストールと設定の手順を確認できます。ご利用の環境に合わせて参照してください。

- Node.js エージェントの基本インストール (56ページ)
- コンテナでの Node.js エージェント (56ページ)
- クラウド環境のデプロイにインテグレーションする Node.js エージェント (57ページ)

Node.js エージェントの基本のインストールと設定

以下のワークフローにより、Node.js エージェントの基本的なインストール手順および設定手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (285ページ)を使用しシステム要件 (288ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストール (290ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

コンテナでの Node.js エージェントのインストールと設定

以下のワークフローにより、コンテナで Node.js エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (285ページ)を使用しシステム要件 (288ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。

- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (290ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

クラウド環境のデプロイにインテグレーションする Node.js エージェントのインストールと設定

以下のワークフローにより、クラウド環境のデプロイにインテグレーションして、Node.js エージェントをインストールおよび設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (285ページ)を使用しシステム要件 (288ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **デプロイタイプ別の設定**：利用するクラウドデプロイタイプに合わせて、インストールします。
   - IBM Cloud (296ページ)利用の場合
   - VMware Tanzu 利用の場合
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

PHP エージェントのインストールと設定のワークフロー

以下のワークフローを記載したページより、PHP エージェントのインストールと設定の手順を確認できます。

- リポジトリによる PHP (58ページ)

リポジトリによる PHP エージェントのインストールと設定

以下のワークフローにより、リポジトリの種類によって PHP をインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (342ページ)を使用しシステム要件 (342ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **リポジトリ別の設定**：利用するリポジトリの種類に合わせて、エージェントをインストールして設定します。
   - RPM (344ページ)利用の場合
   - Debian (342ページ)利用の場合
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Python エージェントのインストールと設定のワークフロー

以下のワークフローを記載したページより、Python エージェントのインストールと設定の手順を確認できます。

- Contrast ランナーによる Python エージェント (414ページ)
- ミドルウェアによる Python エージェント (58ページ)

ミドルウェアによる Python エージェントのインストールと設定

以下のワークフローにより、ミドルウェアの種類別に Python をインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (360ページ)を使用しシステム要件 (359ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。

- こちら (60ページ)に記載されている基本の設定を指定
- 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)

2. **ミドルウェア別の設定**：利用するミドルウェア (384ページ)の種類に合わせて、エージェントをインストール (360ページ)して設定します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Ruby エージェントのインストールと設定のワークフロー

以下のワークフローを記載したページより、Ruby エージェントのインストールと設定の手順を確認できます。

- ミドルウェアによる Ruby エージェント (59ページ)

ミドルウェアによる Ruby エージェントのインストールと設定

以下のワークフローにより、ミドルウェアの種類別に Ruby のインストールして、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (416ページ)を使用しシステム要件 (415ページ)を満たしていること。
- 設定した値が有効になる優先順位 (62ページ)を理解していること。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **ミドルウェア別の設定**：利用するミドルウェア (441ページ)の種類に合わせて、エージェントをインストール (417ページ)して設定します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Go エージェントのインストールと設定のワークフロー

以下のワークフローを記載したページより、Go エージェントのインストールと設定の手順を確認できます。

- インストーラを使用する Go エージェント (59ページ)

インストーラを使用する Go エージェントのインストールと設定

以下のワークフローにより、インストーラを使用して Go エージェントをインストールし、設定する手順を確認できます。

開始する前に

開始する前に、必要なものが揃っているか事前に確認してください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Contrast のサポート対象テクノロジ (476ページ)。
- また、コマンドラインインターフェイス(エージェントをダウンロードするために選択したディレクトリを含む)と、ご利用になる組織の Contrast サーバへのアクセスも必要です。

手順

1. **変数の設定**：エージェントの認証情報を設定する変数を指定します。
   - こちら (60ページ)に記載されている基本の設定を指定
   - 必要に応じて追加の変数を設定(アプリケーションメタデータ、セッションメタデータ)
2. **インストールと設定**：エージェントをインストールして設定 (477ページ)します。
3. **確認**：Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

エージェントの設定

Contrast エージェントをインストールしたら、エージェントがアプリケーションを認識して、Contrast サーバに情報を通信できるように設定しなければなりません。

設定した値が有効になる優先順位 (62ページ)があります。

注記
ライセンスの有効期限が切れていたり、ライセンス数を超えていたりすると、設定に関係なく全てのエージェントの動作が無効になります。

手順

1. 最低限必要な認証情報の変数(Contrast Web インターフェイスで確認できます) (61ページ)を指定します。

```
api:
  url: https://app.contrastsecurity.com
  user_name: contrast_user
  api_key: demo
  service_key: demo
```

   指定する値：
   - **url**： Contrast エージェントが報告する Contrast サーバのアドレス。デフォルトは、https://app.contrastsecurity.com です。
   - **user_name**：Contrast のユーザアカウント(通常、自分のログイン ID)
   - **api_key**：所属する組織の API キー
   - **service_key**：Contrast のユーザアカウントのサービスキー

   これらの認証変数は、以下のいずれかで設定できます。

1. 環境変数
2. YAML 設定ファイル
   - 組織のキーがあらかじめ入力されている YAML 設定ファイル (63ページ)をダウンロードすることができます。Contrast Web インターフェイスで**新規登録**を選択し、**アプリケーション**カードを選択したら、アプリケーションの言語を選択すると、YAML 設定ファイルをダウンロードするリンクが表示されます。

   - また、「YAML 設定エディタを開く」のリンクから Contrast エージェント設定エディタ (64ページ)を開いて、ファイルを編集することもできます。
3. システムプロパティやコマンドラインフラグなど、使用している言語およびツールにネィティブなその他の方法については、各ドキュメントのページを参照してください。

> **注記**
> 設定オプションの一覧およびデフォルト値は、Contrast エージェント設定エディタで参照できます。

2. その他の変数を必要に応じて設定します。
   - セッションメタデータ (535ページ)を使用すると、特定のブランチ、ビルド、コミットしたユーザ、リポジトリなどによって、脆弱性やルート情報をフィルタリングできます。
   - アプリケーションメタデータ (830ページ)を使用すると、カスタム値でアプリケーションをフィルタリングすることができます。

   エージェントの設定ファイルにメタデータとして必要な設定情報を追加すれば、標準の脆弱性データとともに追加の情報がエージェントによって Contrast に報告されるようになります。設定値の一覧や、上記で説明した必要な値以外に何があるかは、こちら (67ページ)をご覧ください。

エージェントキーの検索

> **重要**
> Contrast Web インターフェイスの**新規登録**を選択して、YAML 設定ファイルをダウンロードすると、YAML 設定ファイルにはあらかじめエージェントキーが入力されます。
>
> 独自に YAML ファイルを作成する場合は、自分でエージェントキーを追加する必要があります。

エージェントキーは、組織内の全てのエージェントに共通です。エージェントキーの情報は、アクセスする組織とエージェントを表し、識別するための値です。

エージェントをインストールする場合には、次の全てのキーが必要です。

- エージェントユーザ名
- エージェントサービスキー
- API キー
  こちらの API キーは、全てのエージェント用の API キーとなります。カスタムスクリプトで使用する API キーは、「ユーザの設定」にある API キーを使用してください。
- Contrast URL

手順

1. 右上隅にある**ユーザ名 > 組織の設定**を選択します。
2. **エージェント**を選択すると、 エージェントキーの値が表示されます。

Contrast URL は、*https://app.contrastsecurity.com/Contrast*、もしくはオンプレミス版かプライベートクラウドのインスタンスの URL になります。

> **注記**
> このページにエージェントユーザ名やエージェントサービスキーが表示されない場合は、所属する組織にライセンスが適用されていない可能性があります。その場合はサポートにお問い合わせください。

3. 認証情報が侵害された場合は、エージェントキーを**ローテーション**して新しいキーを生成できます。

> **重要**
> エージェントサービスキーをローテーションすると、全てのエージェントがオフラインになります。アプリケーションは引き続き機能しますが、データは Contrast に送信されなくなります。新しい認証情報の使用を開始するには、エージェントを再設定し、アプリケーションを再起動します。認証情報管理システムなどを使用して、システム間でこの変更を調整してください。

優先順位

設定値が有効になる順序は、以下の優先順位を使用します(1 が最も高い)。

1. ライセンスの有効期限が切れていたり、ライセンス数を超えていたりすると、設定に関係なく全てのエージェントの動作が無効になります。
2. コマンドラインまたはシステムのプロパティ値(使用している言語に合わせて適切なプロパティが指定されている場合)
   例：`-Dcontrast.enable`
3. 環境変数 (66ページ)
   例：CONTRAST__ENABLE
4. アプリケーション固有の設定ファイル(.NET Framework のみ)
   例：*web.config (182ページ)*

5. YAML ファイル (63ページ)の設定値。設定値は以下の順で最初に見つかったファイルから取得されます。
   a. ユーザが指定した YAML ファイル
      例：
      - Java：`contrast.config.path` システムプロパティ (104ページ)
      - Node.js：`--configFile` コマンドラインフラグ
        Node.js：`--configFile` コマンドラインフラグ
      - すべてのエージェント：`CONTRAST_CONFIG_PATH` 環境変数
   b. 現在の作業ディレクトリにある *contrast_security.yaml* ファイル(Java を除く全てのエージェント)
      例：*./contrast_security.yaml*
   c. アプリケーションの構成ディレクトリにある contrast_security.yaml ファイル(Ruby および Python のみ)
      例：
      - Ruby on Rails：*./config/contrast_security.yaml*
      - Django：*./settings/contrast_security.yaml*
   d. エージェント固有の設定ディレクトリにある *contrast_security.yaml* ファイル。エージェントがサービスを使用しており、エージェントとサービスに個別の YAML ファイルを使用する必要がある場合に、このディレクトリを使用します。
      例：
      - */etc/contrast/agentname/contrast_security.yaml* (*agentname* は次のいずれか：*dotnet*、*go*、*java*、*node*、*python*、*ruby*、 *webserver*)
      - *%ProgramData%\Contrast\agentname\contrast_security.yaml*(*agentname* は次のいずれか：*dotnet*、*dotnet-core*、*java*、*node*、*python*、*ruby*、 *webserver*)
   e. サーバの*/etc/contrast* ディレクトリ内にある *contrast_security.yaml* ファイル(.NET Framework と.NET Core 以外の全てのエージェント)。エージェントがサービスを使用しており、エージェントとサービス間で共有する YAML ファイルを使用する必要がある場合に、このディレクトリを使用します。
      例：
      - */etc/contrast/contrast_security.yaml*
      - *%ProgramData%\Contrast\contrast_security.yaml*
6. Contrast Web インターフェイスで設定されている値
   例：Contrast Web インターフェイスでの Assess や Protect モードのトグルボタンでの切り替えは、`assess.enable` や `protect.enable` に対応付けられます。
7. Contrast Security が設定したデフォルト値

関連項目

その他の設定 (67ページ)

YAML 設定

YAML 設定ファイルを使用して、Contrast エージェントの設定プロパティを指定できます。YAML 設定ファイルの値は、環境変数またはコマンドラインの引数で上書きできます。

すべての Contrast エージェントで、YAML 設定ファイルのプロパティは同じフォーマットを使用できますが、各エージェントに適用される固有のプロパティがあるため、各エージェントで固有の指定ファイルを使用する必要があります。

設定ファイルは *contrast_security.yaml* という名前で、ロードパス (62ページ)に正しく置かれている必要があります。

Contrast Web インターフェイスからエージェントの設定ファイルをダウンロードすると、設定ファイルには Contrast サーバと通信するために最低限必要な基本のプロパティが入ります。独自の設定ファイルを作成する場合は、基本のキー (61ページ)を自分で追加してください。

全てのエージェントで最低限必要な *contrast_security.yaml* の内容は、以下のようになります。

```
api:
    url: https://app.contrastsecurity.com
    user_name: contrast_user
    api_key: demo
    service_key: demo
```

ヒント

- Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。
- YAML は JSON のスーパーセットとなっているため、YAML ファイルで JSON を使用して Contrast エージェントを設定することもできます。

以下より参照できる YAML テンプレートを使用して、各エージェントの *contrast_security.yaml* を作成できます。

- Java (104ページ)
- .NET Framework (183ページ)
- .NET Core (248ページ)
- Node.js (305ページ)
- PHP (345ページ)
- Python (362ページ)
- Ruby (420ページ)
- Go (481ページ)

注意

YAML テンプレートでは空白が認識されます。スペースは使用できますがタブは使用できないため、編集時には注意が必要です。各プロパティの説明は、テンプレート内のコメントとして記載されています(スペースの後にシャープ記号「#」で続く行がコメントになります)。

Contrast エージェント設定エディタを使う

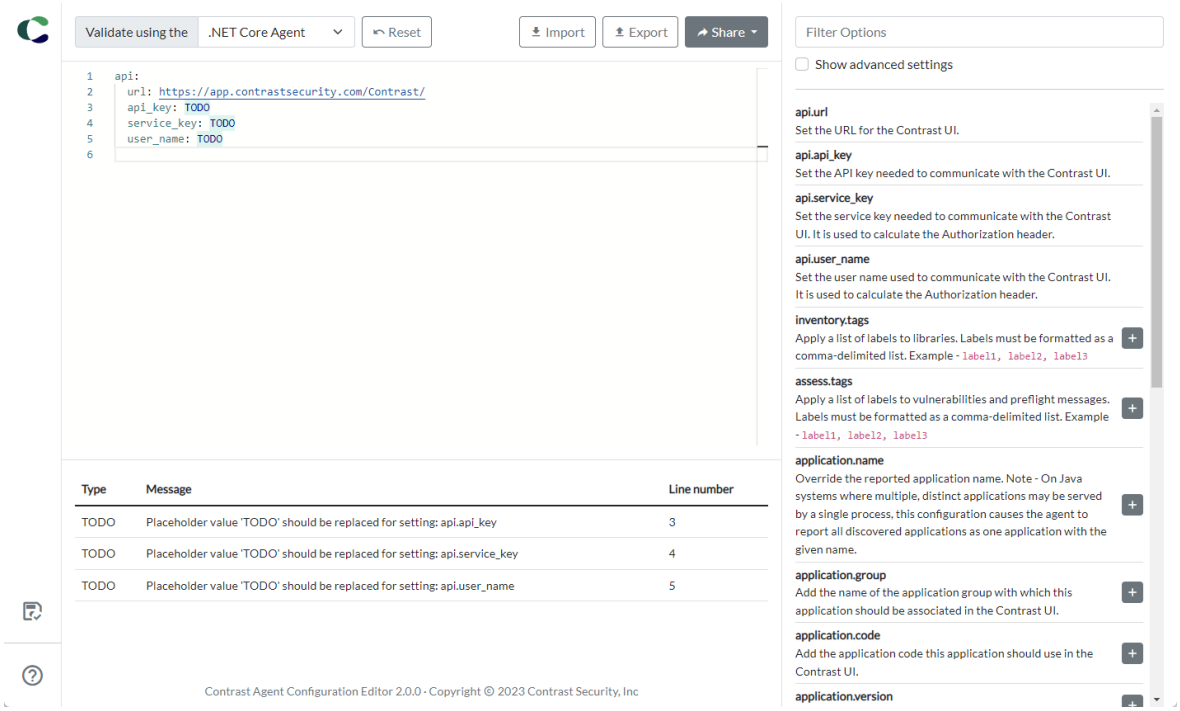
Contrast エージェント設定エディタは、Contrast エージェントの設定を検証および生成するための Web アプリケーションです。

開始する前に

- このエディタは、Contrast エージェントの設定を編集、検証、生成するために使用してください。
- 既に YAML 設定ファイルがある場合は、**Import** を選択するとエディタにファイルを取り込むことができます。
- エディタは完全にブラウザ内だけで実行され、API キーなどの機密情報がローカルマシンから出ることはありません。

手順

1. ブラウザで Contrast エージェント設定エディタを開きます。



2. **Import** を選択して既存の YAML ファイルをインポートするか、YAML ファイルの内容を貼り付けます。
3. テキストを編集する際に、無効な YAML を入力すると、エラー警告が表示されます。エージェント固有の YAML 検証をするために、**Validate using the** の一覧からエージェントのプログラム言語を選択します。
   ここでは次のような検証が行われます。
   - **YAML 構文解析** により、テキストが YAML であることを確認します。YAML が無効な場合、**Error** が表示され、それ以降の検証は行われません。
   - **設定キーの検証** により、YAML ノードが選択したエージェントでサポートされているキーであることを確認します。認識できないキーには **Warning** が表示されます。
   - **設定値の検証**により、YAML 値がブール値、数値、列挙型(例：ログレベル)などの型の期待値と一致するかどうかを確認します。値が無効な場合、**Warning** が表示されます。
   - **設定の整合性の検証**により、相容れない設定が同時に存在していないことを確認します。この検証は現在のところ、`application.session_id` と `application.session_metadata` の設定のみに限定されています。整合性のない設定には、**Warning** が表示されます。
   - **未定値の検証**により、設定に未定値の `TODO` がある場合に通知されます。未定値には、**TODO** が注として表示されます。

   Error、**Warning** や **TODO** が表示されている行をクリックすると、その設定行の先頭にカーソルが置かれます。
4. 右側のパネルを使用して、利用可能な設定(説明あり)を検索できます。プラスアイコン( )をクリックすると、その設定がエディタ画面の YAML に追加されます。
   この機能を使用して新しい設定を追加すると、YAML がフォーマットされ、ノードが並べ替えられ、YAML 内の余分な空白が削除されます。
   Reset をクリックすると、元のファイル設定に戻ります。

> **注記**
> テキストエディタの YAML に構文エラーがあるか、有効な YAML でない場合、生成した YAML は無効となります。

5. 編集が終了したら、エージェント設定ファイルを YAML または環境変数として エクスポートします。 他の共同作業者とファイルを共有することもできます。

注記
この時点で、Contrast エージェント設定エディタは完全にオフラインで実行されます (つまり、最初のアクセス後、インターネット接続がなくてもページを操作できます)。 エディタに更新がある場合は、自動的にバックグラウンドでダウンロードされます。新しいバージョンへ更新するには、エージェント設定エディタの全てのブラウザのタブを閉じる必要があります。 ページのリフレッシュだけでは、新しいバージョンには更新されません。

環境変数

環境変数を使用して、サポート対象のプロパティをエージェントに設定することができます。

環境変数は、エージェントの起動前に指定し、エージェントがアクセスできる場所に設定する必要があります。環境変数は、同じプロセス内でもシステム全体でも設定できます。

重要
システム全体の環境変数を設定する場合、同じサーバで実行中の他の Contrast エージェントに影響を与える可能性があります。

Contrast のプロパティは、コマンドライン、YAML 設定、環境変数に変換できます。

コマンドライン形式の変数を環境変数に変換するには、パスセグメントの区切り記号(.)をアンダースコア 2 つ(__)に置き換えます。

YAML 形式の変数を環境変数に変換するには、最上位のプロパティから始めて、ネストされたすべてのプロパティをアンダースコア 2 つ(__)で区切ります。

そして、「contrast」というネームスペース(`contrast.` か `CONTRAST__`)を先頭に付けます。

環境変数はすべて大文字で、スペースを入れないでください。

例：

コマンドライン	YAML プロパティ	環境変数
contrast.server.name	`server:` `  name:`	CONTRAST__SERVER__NAME
contrast.api.api_key	`api:` `  api_key:`	CONTRAST__API__API_KEY

各エージェントでサポートされているすべてのプロパティの一覧は、それぞれの YAML テンプレートで確認できます。

- Java (104ページ)
- .NET Framework (183ページ)
- .NET Core (248ページ)

- Node.js (305ページ)
- PHP (345ページ)
- Python (362ページ)
- Ruby (420ページ)
- Go (480ページ)

関連項目

設定に使用できる環境変数の詳細については、その他の設定 (67ページ)を参照してください。

その他の設定

以下の共通変数は、システムプロパティ、環境変数、YAML 設定ファイル、またはデフォルト値のいずれかで設定できます。

環境変数で設定するその他の設定値

以下の共通変数を使用して、システムを設定できます。

エージェント言語固有の設定や詳細な設定で更新もできますので、全ての設定の一覧については、Contrast エージェント設定エディタを参照してください。

環境変数	説明	言語
CONTRAST__API__URL	Contrast の URL を設定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__API__API_KEY	Contrast との通信に必要な API キーを設定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__API__SERVICE_KEY	Contrast との通信に必要なサービスキーを設定します。これは認証ヘッダを計算するために使用されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__API__USER_NAME	Contrast との通信に使用するユーザ名を設定します。これは認証ヘッダを計算するために使用されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__INVENTORY__TAGS	ライブラリにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：`label1, label2, label3`	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP
CONTRAST__ASSESS__TAGS	脆弱性やプリフライトメッセージにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：`label1, label2, label3`	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__APPLICATION__NAME	Contrast に報告されるアプリケーション名を上書きします。注：Java システムで、複数の異なるアプリケーションが 1 つのプロセスで処理される可能性がある場合、この設定によって、検出された全てのアプリケーションが指定された名前で 1 つのアプリケーションとして報告されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__APPLICATION__GROUP	Contrast でこのアプリケーションに関連付けるアプリケーショングループの名前を指定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__APPLICATION__CODE	Contrast でこのアプリケーションに使用するアプリケーションのコード名称を指定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__APPLICATION__VERSION	Contrast に報告されるアプリケーションのバージョンを上書きします。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__APPLICATION__TAGS	アプリケーションにタグを適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：`label1, label2, label3`	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__SERVER__NAME	Contrast に報告されるサーバの名前を上書きします。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go

環境変数	説明	言語
CONTRAST__SERVER__ENVIRONMENT	Contrast に報告されるサーバの環境を上書きします。有効な値：`QA`、`PRODUCTION`、`DEVELOPMENT`	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
CONTRAST__SERVER__TAGS	サーバにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：`label1, label2, label3`	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go

web.config で設定するその他の設定値

.NET Framework または.NET Core を次のいずれかで使用する場合、これらの変数も設定する必要があります。

プラットフォーム	変数セット

Web.config
 	
(.NET Core での IIS モジュール)
	`<environmentVariable name="CONTRAST__API__URL" value="https://app.contrastsecurity.com/Contrast/ " />`
	`<environmentVariable name="CONTRAST__API__API_KEY" value="" />`
	`<environmentVariable name="CONTRAST__API__SERVICE_KEY" value="" />`
	`<environmentVariable name="CONTRAST__API__USER_NAME" value="" />`
	`<environmentVariable name="CONTRAST__INVENTORY__TAGS" value="" />`
	`<environmentVariable name="CONTRAST__ASSESS__TAGS" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__NAME" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__GROUP" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__CODE" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__VERSION" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__TAGS" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__METADATA" value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__SESSION_ID" \ value="" />`
	`<environmentVariable name="CONTRAST__APPLICATION__SESSION_METADATA" \ value="" />`
	`<environmentVariable name="CONTRAST__SERVER__NAME" value="localhost" />`
	`<environmentVariable name="CONTRAST__SERVER__ENVIRONMENT" \ value="development" />`
	`<environmentVariable name="CONTRAST__SERVER__TAGS" value="" />`

プラットフォーム	変数セット

Azure App Service	[
	{
	"name": "CONTRAST__API__URL",
	"value": "https://app.contrastsecurity.com/Contrast/ "
	},
	{
	"name": "CONTRAST__API__API_KEY",
	"value": ""
	},
	{
	"name": "CONTRAST__API__SERVICE_KEY",
	"value": ""
	},
	{
	"name": "CONTRAST__API__USER_NAME",
	"value": ""
	},
	{
	"name": "CONTRAST__INVENTORY__TAGS",
	"value": ""
	},
	{
	"name": "CONTRAST__ASSESS__TAGS",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__NAME",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__GROUP",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__CODE",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__VERSION",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__TAGS",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__METADATA",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__SESSION_ID",
	"value": ""
	},
	{
	"name": "CONTRAST__APPLICATION__SESSION_METADATA",
	"value": ""
	},
	{
	"name": "CONTRAST__SERVER__NAME",
	"value": "localhost"
	},
	{
	"name": "CONTRAST__SERVER__ENVIRONMENT",
	"value": "development"
	},
	{
	"name": "CONTRAST__SERVER__TAGS",
	"value": ""
	}
	]

システムプロパティで設定するその他の設定値

設定値	言語
-Dcontrast.api.url	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.api.api_key	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.api.service_key	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.api.user_name	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.inventory.tags	Java、.NET Framework、.NET Core、Node.js、PHP、 Python、Ruby
-Dcontrast.assess.tags	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.application.name	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.application.group	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.application.code	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.application.version	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.application.tags	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.server.name	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.server.environment	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go
-Dcontrast.server.tags	Java、.NET Framework、.NET Core、Node.js、PHP、Python、Ruby、Go

YAML 設定ファイルで指定するその他の設定値

YAML 設定ファイルを使用して、以下の追加の設定値を指定できます。

プロパティ	説明	言語
contrast.api.url	Contrast の URL を設定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.api.api_key	Contrast との通信に必要な API キーを設定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.api.service_key	Contrast との通信に必要なサービスキーを設定します。これは認証ヘッダを計算するために使用されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.api.user_name	Contrast との通信に使用するユーザ名を設定します。これは認証ヘッダを計算するために使用されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.inventory.tags	ライブラリにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：label1, label2, label3	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP
contrast.assess.tags	脆弱性やプリフライトメッセージにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：label1, label2, label3	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.name	Contrast に報告されるアプリケーション名を上書きします。注：Java システムで、複数の異なるアプリケーションが 1 つのプロセスで処理される可能性がある場合、この設定によって、検出された全てのアプリケーションが指定された名前で 1 つのアプリケーションとして報告されます。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.group	Contrast でこのアプリケーションに関連付けるアプリケーショングループの名前を指定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go

プロパティ	説明	言語
contrast.application.code	Contrast でこのアプリケーションに使用するアプリケーションのコード名称を指定します。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.metadata	アプリケーションに関連付けるユーザ定義のメタデータを指定するための、key=value(キーと値)のペアの設定(RFC 2253 に準拠)を定義します。設定は、key=value をペアにしたカンマ区切りのリストとして書式設定する必要があります。例：business-unit=accounting、office=Baltimore	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.session_id	Contrast に既に存在するセッションの ID を指定します。エージェントによって検出された脆弱性が、このセッションに関連付けられます。無効な ID が指定された場合、エージェントは無効になります。このオプションと application.session_metadata は排他関係にあり、両方が設定されている場合エージェントは無効になります。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.session_metadata	Contrast でセッションの新規作成時に使用されるメタデータを指定します。エージェントによって検出された脆弱性は、この新しいセッションに関連付けられます。この値は、key=value をペアにして書式設定(RFC2253 に準拠)する必要があります。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.version	Contrast に報告されるアプリケーションのバージョンを上書きします。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.application.tags	アプリケーションにタグを適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：label1, label2, label3	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.server.name	Contrast に報告されるサーバの名前を上書きします。	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.server.environment	Contrast に報告されるサーバの環境を上書きします。有効な値：QA、PRODUCTION、DEVELOPMENT	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go
contrast.server.tags	サーバにタグの一覧を適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。例：label1, label2, label3	Java、.NET Framework、.NET Core、Node.js、Python、Ruby、PHP、Go

タグおよびデータ

タグ

ユーザ定義の基準に基づいて、アプリケーションやサーバのフィルタリングが必要な場合があります。この場合、メタデータの代わり、またはメタデータに加えてタグを使用することができます。タグは、アプリケーション (531ページ)、サーバ (589ページ)、ライブラリ (603ページ)、脆弱性 (697ページ)に適用できます。このようなタグを使用することで、Contrast で項目を整理して、効率的に検索ができます。

メタデータ

エージェントの設定時にカスタムフィールドを作成して、アプリケーションのメタデータを収集できます。特定のアプリケーションの所有者や事業部門、場所、またはアプリケーションに関するその他の重要な情報を識別するためのフィールドを設定できます。

セッションメタデータ

特定のフィールドの情報に対する脆弱性データをフィルタリングする必要がある場合があります。セッションメタデータをエージェントの設定に指定するとアプリケーションの脆弱性 (693ページ)タブで、フィルタとして使用できます。

指定できるフィールドは以下の通り：

名前	値
コミットハッシュ	`commitHash`
コミットしたユーザ	`committer`
ブランチ名	`branchName`
Git タグ	`gitTag`
Repository	
テストラン	`testRun`
バージョン	`version`
ビルド番号	`buildNumber`

詳細は、セッションメタデータ (535ページ)を参照ください。

アプリケーションを疎通するためのオプション

Contrast エージェントをインストールして設定した後、アプリケーションを十分に疎通することで、Contrast で脆弱性に関する最も正確な情報を参照できるようになります。

アプリケーションに Contrast エージェントを組み込む方法として、以下のようなオプションがあります。お使いのツールや環境に応じて、アプリケーションで可能な限り多くのルートを疎通するのに利用できます。

最適な結果を得るために：	テストの要件
CI/CD パイプラインで統合テストやスモークテストからのリクエストを受け取るサーバにエージェントを組み込む (72ページ)	既存の自動テスト
手動テストからのリクエストを受けるサーバにエージェントを組み込む (73ページ)	手動でテストを行うユーザ
Web アプリケーションテスト自動化ツールからのリクエストを受信するサーバにエージェントを組み込む (73ページ)	自動テスト
API テストツールからのリクエストを受け取るサーバにエージェントを組み込む (73ページ)	Postman などの API テストツール
DAST テストツールを使用したサーバにエージェントを組み込む (73ページ)	Rapid7 などの DAST ツール
オープンソースのクローラーを実行する (74ページ)	Zap などの無料ツール
手動によるペネトレーションテスト環境で使用するサーバにエージェントを組み込む (74ページ)	社内または社外のペネトレーションテスト担当者によるアプリケーションの実行
BurpSuite ベースのペネトレーションテストに使用するサーバにエージェントを組み込む (74ページ)	Burp(BurpTrast のインテグレーション)
Assess データを使用して curl コマンドを実行する (74ページ)	API 認証できるユーザ

CI/CD パイプラインへの組込み

アプリケーションを疎通するためのこのオプションでは、CI/CD パイプラインで統合テストやスモークテストからのリクエストを受信するサーバにエージェントを組み込みます。

詳細

必要条件	対象ユーザ	環境
• 自動化テスト(リグレッションテスト、統合テスト、スモークテスト、パフォーマンステスト)、 • 自動化を管理しオーケストレーション機能を持つ CI ツール、 • CI/CD のサーバ	テストの作成、CI の管理、CI サーバを操作する担当者 これには通常、QA、開発、DevOps の担当者が含まれます。	すべての環境

[en]

手動テストでの組込み

アプリケーションを疎通するためのこのオプションでは、手動テストによってリクエストを受信するサーバにエージェントを組み込みます。

詳細

必要条件	対象ユーザ	環境
手動テストに使用できるサーバ	手動テストを実施する QA(テスト環境)テスト担当者	自動テスト環境ではなく、手動によるリソースが利用可能な環境

[en]

Web アプリケーションテストツールでの組込み

アプリケーションを疎通するためのこのオプションでは、Web アプリケーションのテスト自動化ツール(例えば、Cypress や Selenium)からのリクエストを受信するサーバにエージェントを組み込みます。

詳細

必要条件	対象ユーザ	環境
• 自動化テスト(リグレッションテスト、統合テスト、スモークテスト、パフォーマンステスト)、 • テストのオーケストレーション機能を持つ Cypress や Selenium などのツール • アプリケーションを計測するためのサーバ	QA(テスト環境)、自動化エンジニア、DevOps 開発者	すべての環境

[en]

API テストツールでの組込み

アプリケーションを疎通するためのこのオプションでは、API テストツール(例えば、Postman)からのリクエストを受信するサーバにエージェントを組み込みます。

詳細

必要条件	対象ユーザ	環境
開発環境で Postman にアクセスできる開発担当者、または QA(テスト)環境のサーバと Postman の自動化	ローカルマシンで操作する開発担当者、あるいは QA(テスト)用に Postman スクリプトを作成する担当者	すべての環境

DAST ツールでの組込み

アプリケーションを疎通するためのこのオプションでは、Rapid7、Veracode、Netsparker などの既存の動的アプリケーションセキュリティテスト(DAST)と共にエージェントをサーバに組込みます。

詳細

必要条件	対象ユーザ	環境
必要な DAST ツールの導入 ほとんどの場合、購入が必要なツールとなります。	DAST 用サーバに対象アプリケーションをデプロイするセキュリティおよび DevOps 担当者	DAST ツールを使用するテスト環境 **注記** この方法は、他のテスト方法を補完するものと考えてください。

[en]

オープンソースのクローラーでの組込み

このオプションでは、オープンソースのクローラー(例えば、Zap)を使用してアプリケーションを疎通します。

詳細

必要条件	対象ユーザ	環境
• 無料のクローリングツール • QA(テスト)環境での実装、または開発環境のローカルで使用	• オープンソースのクローラーを CI/CD パイプラインに統合する DevOps 担当者 • 手動でツールを実行する開発担当者	自動化デプロイとテスト環境 **注記** オープンソースのクローラーツールは、ペイロードの性質上、アプリケーションを完全に疎通することができない可能性があります。

[en]

手動ペネトレーションテストでの組込み

アプリケーションを疎通するためのこのオプションでは、手動のペネトレーションテストを行う環境(自社または NetSPI などのサードパーティ)に使用するサーバにエージェントを組み込みます。

詳細

必要条件	対象ユーザ	環境
• ペネトレーションテストサービスの購入 • テストのスケジューリング	社内または社外のテスト担当者	社内にペネトレーションテスト担当者がいる環境。 社外のペネトレーションテスト担当者の場合は、ペネトレーションテストのコストを削減する方法として、Contrast Assess の使用を検討してください。

[en]

Burp Suite ベースのペネトレーションテストでの組込み

アプリケーションを疎通するためのこのオプションでは、Burp Suite ベースのペネトレーションテストに使用するサーバにエージェントを組み込みます。

詳細

必要条件	ユーザ	環境
Burp Suite で BurpTrast のインテグレーションを有効にします。 無料版はセキュリティ担当者にとっても使い易く便利です。 既に購入している場合は、有料版を使用してください。	社内のペネトレーションテスト担当者またはセキュリティ担当者	Burp Suite の無料版または有料版を使用する環境 ほとんどの場合、Burp Suite の無料版で十分で、セキュリティ担当者が直接使用することができます。 すでに購入されている場合は、有料版を使用してください。

Assess データを使用して curl コマンドを実行

アプリケーションを疎通するためのこのオプションでは、Assess データを使用して curl コマンドを実行します。

詳細

必要条件	対象ユーザ	環境
• コマンドラインインターフェイスの使用 • API コールを認証する機能	サーバへのアクセス権限と認証トークンがあれば、誰でも特定の API を呼び出すことができます。	サーバとトークンの情報を持っているユーザ。 この方法は、他のテスト方法を補完するものと考えてください。

[en]

Java エージェント

Contrast の Java エージェントによって、Java ベースのアプリケーションで、Contrast Assess や Contrast Protect による解析が可能になります。エージェントは、従来のアプリケーションサーバで構築される Java の Web アプリケーションや、Netty、Play、Spring Boot などで構築される新しい Java の Web アプリケーションを解析します。JVM があれば、Java エージェントは詳細なセキュリティ情報を提供できます。

アプリケーションが実行されると、Java エージェントのセンサーがアプリケーションのセキュリティ、アーキテクチャ、ライブラリに関する情報を収集します。エージェントによる解析の結果は、Contrast Web インターフェイスで確認できます。

アプリケーションの解析を開始するには、Java エージェントをインストール (77ページ)します。

Java(Kotlin、Scala)エージェントのサポート対象テクノロジ

Java エージェント

テクノロジ	サポート対象バージョン	備考
Java ランタイム	• IBM 8 • Oracle 8　*バージョン 11 以降は OpenJDK のサポートに従う • OpenJDK 8、11、12、13、14、15、16、17 、18、19 、20 、21	OpenJDK のサポートは、ここに示す現在のサポート対象バージョンの範囲内で、全ての公開中のビルドで動作するように設計されています。一般的によく利用される Azul や Amazon Corretto などもサポート対象の JDK のカテゴリに入ります。 **サポート対象外：** JDK プレビュー機能
旧 Java エージェント (165ページ)の Java ランタイム Java エージェント 3.x のみでの使用	• IBM 6、7 • Oracle 6、7 • OpenJDK 6、7	**参考** • サポート速報 : Java6 および Java7 のサポート終了について • FAQ : Java6 および Java7 のサポート終了について
アプリケーションサーバ	• GlassFish 4 • Grizzly 2.3.20 以降 • JBoss EAP 6.x および 7.x • Jetty 7、8、9、10、11 • Karaf 3.0.x • Netty 4.x • Play 2.4 • Resin 4 • Tomcat 5、6、7、8、9 、10 • Vert.x 3.1.0 、4.x • WebLogic 10、11g、12c、14 • WebSphere* 8.5、9.0 • WebSphere Liberty 22 • WildFly 10、11、14、18、23-27	* zSeries および AIX の環境ではサポートが限定されます。SPARC Solaris で WebSphere を使用する場合、バージョン 8.5.5.11 が必須となります。 **ルートカバレッジのサポート：** • GlassFish 4 • Jetty 11.0、10.0、9.4、8.1、7.6 • Resin 4.0 • Tomcat 5、6、7、8、9、10 • WebLogic 12、14 • WebSphere 8.5、9.0 • WildFly 10、11、14、18、23-27

テクノロジ	サポート対象バージョン	備考
オプティマイザ	Proguard	Proguard にある Java バイトコードの最適化機能は、Contrast のようなランタイムエージェントが基本前提とする動きに反します。Proguard ユーザが、Contrast を利用してアプリケーションを保護する場合は、Proguard の-dontoptimize 設定オプションを使用して最適化を行わないようにする必要があります。
データベース	• DB2 • DynamoDB • MySQL • Oracle • PostgreSQL • SQL Server • SQLite JDBC ドライバ	
メッセージングサービス	• JMS 2.0 • IBM MQ 9.x • Spring JMS 2.x	• エージェントのバージョン：Java 4.7.0 以降 • Contrast のバージョン：3.9.9 以降
	• Kafka メッセージキュー・ストリーミング	• エージェントのバージョン： Java 5.0.0 以降
その他の Java テクノロジ	• ADF JSF • Apache POI、Fileupload、HttpComponents • Axis (RPC)、XMLRPC、RMI、Apache CXF、JMS (javax.jms) • Direct Web Remoting (DWR) • DropWizard • Freemarker • Glowroot* • GSON、Kryo、minidev、org.json • Google Web Toolkit (GWT) • Hibernate • http4k(4.6.0.0 と 4.17、Contrast Assess 対象) • J2SE • JDBC、JDBI、MongoDB • JSF (MyFaces、RichFaces、Sun) • java.nio、java.beans • Java EE/J2EE、Servlet/JSP • Jersey • MyBatis • OWASP ESAPI、AntiSamy、Coverity • Quarkus RESTEasy • Seam • Spring、Spring Boot、Spring AOP • Spring WebFlux 5 および 6 • Struts、Struts 2 • Wicket • XStream、Jackson (JSON/XML) • Xerces、JAXB、nu.xom	* Glowroot を使用する場合、glowroot.jar よりも前に、Contrast エージェントの jar を指定して、ロードする必要があります。 **ルートカバレッジのサポート：** • http4k-core 4.17 • http4k-core 4.6 • Jersey 2.25、2.28、2.36、2.6 • Quarkus RESTeasy 2.15 • Spring Web MVC 4.2、5.3、6.0 • Spring WebFlux 5 および 6 • Struts 2

Kotlin

テクノロジ	サポート対象バージョン
Contrast エージェント	3.9.1.25108 以降
Java ランタイム	JDK 8 以上
Kotlin のバージョン	1.5.x

Scala

テクノロジ	サポート対象バージョン
Contrast エージェント	3.8.11.23624 以降
Java ランタイム	JDK 8 以上
Scala のバージョン	2.12, 2.13
Play のバージョン	2.6, 2.7, 2.8

テクノロジ	サポート対象バージョン
Akka HTTP	10.2.4

WebSphere の設定

WebSphere をアプリケーションサーバとして使用している場合は、エージェントをデプロイする前に、Websphere で Java エージェントを設定する (100ページ)に記載されている情報もご確認ください。

Java エージェントのインストール

Java エージェントのインストール方法はいくつかありますので、ご利用の環境に合わせて選択してください。 Contrast を使用する場所(例えば、開発環境で Assess、本番環境では Protect など)、既存のビルドツール、アプリケーションのデプロイ方法などを考慮する必要があります。

ヒント

エージェントベースのテクノロジを複数使用していて、Contrast Java エージェントを並行して使用する場合は、起動時にロードされる最初のエージェントとして、Contrast Java エージェントを必ず指定してください。例：

```
java -javaagent:contrast.jar -javaagent:newrelic.jar
```

Contrast Java エージェントを最初にロードすることで、パフォーマンスへの影響を抑えることができます。

Contrast とホットデプロイ

ホットデプロイとは、アプリケーションサーバのプロセスを停止して再起動することなく、稼働中のサーバに新しいコンポーネント(WAR ファイル、サーブレット、JSP ファイルなど)を追加することを指します。

Contrast エージェントは、ホットデプロイ中もホットリロード中も動作し続けますが、以下の点を考慮してください。

- ホットデプロイ中に動的に追加または削除されたライブラリは、Contrast で検出されない場合があります。
- ホットデプロイ中にセッションメタデータは更新されません。
- 一部の WebSphere ユーザで問題が発生する可能性があります。

ホットデプロイでの問題が発生した場合は、アプリケーションサーバを再起動してください。

クイックスタート

Java エージェントがどのように動作するか試してみたい場合は、こちらの Java クイックスタートガイド (102ページ)をご覧ください。

基本のインストール

Java エージェントのインストールは、ほとんどの場合(Tomcat のなどのアプリケーションサーバ、Docker などのコンテナを使用する場合)、Java エージェントを取得するリポジトリを選択し、各リポジトリの手順に従ってエージェントをダウンロードしてインストールしてください。

- Maven Central (78ページ)
- Debian (79ページ)
- RPM (80ページ)

ビルドに組み込むインストール

開発環境で Assess を使用しており、脆弱性検出時に既存のソフトウェアプロジェクトでビルド結果を設定したい場合は、以下を参照してエージェントをインストールしてください。

- Maven プラグイン (772ページ)
- Gradle プラグイン (756ページ)
- Jenkins プラグイン (758ページ)

Maven Central を使用して Java エージェントをインストール

Contrast Java エージェントは、グループ IDcom.contrastsecurity とアーティファクト IDcontrast-agent を使用して、Maven Central から入手できます。 Java エージェントをインストールするには：

1. Maven Central から *contrast-agent.jar* を入手します(Maven リポジトリからダウンロードする方法は例を参照)。
2. Java エージェントを設定 (103ページ)します。 YAML 設定ファイル (63ページ)を作成またはダウンロードします。設定ファイルには、エージェントキー (61ページ)を使用して、Contrast の接続パラメータを指定する必要があります。
3. エージェントがファイルを認識できるように YAML 設定ファイル(contrast.yaml)の場所を指定します。以下の例では、 <YourContrastJarPath>を Contrast JAR ファイルのパス(これは内部のファイル構造やダウンロードした方法によって異なります)に、<ApplicationJar>をアプリケーションの JAR ファイル名に置き換えます。

   ```
   java -javaagent:<YourContrastJarPath> -
   Dcontrast.config.path=contrast.yaml -jar <ApplicationJar>.jar
   ```

   注記

   YAML の代わりにシステムプロパティや環境変数を使って設定する、またはエージェントが自動的に検索できる標準の場所 (62ページ)に YAML 設定ファイルを置いている場合は、Java エージェントを含めるよう JVM パラメータを設定してください。

   ```
   java -javaagent:<YourContrastJarPath> -jar <AppName>.jar
   ```

4. 通常通りにアプリケーションを使用します(例えば、アプリケーションの Web インターフェイスをクリックして、API コマンドを送信するなど)。Contrast でアプリケーションが認識されていることを確認します(例えば、Contrast の Web インターフェイスでアプリケーションを参照したり、ログを表示するなど)。

Maven Central にデプロイされている Contrast アーティファクトは、https://keyserver.ubuntu.com がホストする GPG キーで署名されています。Contrast の公開署名キーの ID は 1AAD9AFB3FC5CCA6940D021534D84B137E8F1053 で、次のコマンドでローカル鍵にインストールできます。

```
gpg --keyserver keyserver.ubuntu.com --recv-keys \
1AAD9AFB3FC5CCA6940D021534D84B137E8F1053
```

また、以下のようなアプリケーションサーバと一緒に Contrast エージェントをインストールすることで、テスト環境や本番環境で動作するアプリケーションのセキュリティ分析を行うことができます。

- Jetty (98ページ)
- JBoss/Wildfly (96ページ)
- Tomcat (98ページ)
- WebLogic (99ページ)

- WebSphere (100ページ)

Docker などのコンテナを使用してインストール (81ページ)することもできます。

ヒント
エージェントのインストールで互換性があるその他の方法については、Contrast サポートポータルを参照してください。VMware Tanzu を使用している場合は、VMware Tanzu で Java をインストール (86ページ)をご覧ください。

Debian リポジトリを使用して Java エージェントをインストール

Contrast Debian リポジトリから Java エージェントを取得してインストールするようシステムを設定できます。これを行うには、以下の手順を実行します。

1. 以下のコマンドを使用して、リポジトリからパッケージを受け取るようシステムを設定します。

```
curl https://pkg.contrastsecurity.com/api/gpg/key/public | sudo apt-key \
add -
echo "deb https://pkg.contrastsecurity.com/debian-public/ all contrast" |
 sudo tee /etc/apt/sources.list.d/contrast-all.list
```

2. Contrast Java エージェントをインストールします。

```
sudo apt-get update && sudo apt-get install contrast-java-agent
```

3. */opt/contrast/contrast-agent.jar* として、Java 用の Contrast エージェントの JAR ファイルがインストールされます。
4. Java エージェントを設定 (103ページ)します。YAML 設定ファイル (63ページ)を作成またはダウンロードします。エージェントキー (61ページ)を使用して、Contrast の接続パラメータを指定する必要があります。
5. エージェントがファイルを認識できるように YAML 設定ファイル(`contrast.yaml`)の場所を指定します。以下の例では、 `<YourContrastJarPath>`を Contrast JAR ファイルのパス(これは内部のファイル構造やダウンロードした方法によって異なります)に、`<ApplicationJar>`をアプリケーションの JAR ファイル名に置き換えます。

```
java -javaagent:<YourContrastJarPath> -
Dcontrast.config.path=contrast.yaml -jar <ApplicationJar>.jar
```

注記
YAML の代わりにシステムプロパティや環境変数を使って設定する、またはエージェントが自動的に検索できる標準の場所 (62ページ)に YAML 設定ファイルを置いている場合は、Java エージェントを含めるよう JVM パラメータを設定してください。

```
java -javaagent:<YourContrastJarPath> -jar <AppName>.jar
```

6. 通常通りにアプリケーションを使用します(例えば、アプリケーションの Web インターフェイスをクリックして、API コマンドを送信するなど)。Contrast でアプリケーションが認識されていることを確認します(例えば、Contrast の Web インターフェイスでアプリケーションを参照したり、ログを表示するなど)。

また、以下のようなアプリケーションサーバと一緒に Contrast エージェントをインストールすることで、テスト環境や本番環境で動作するアプリケーションのセキュリティ分析を行うことができます。

- Glassfish
- Jetty (98ページ)
- JBoss/Wildfly (96ページ)
- Tomcat (98ページ)
- WebLogic (99ページ)
- WebSphere (100ページ)

Docker などのコンテナを使用してインストール (81ページ)することもできます。

ヒント
エージェントのインストールで互換性があるその他の方法については、Contrast サポートポータルを参照してください。VMware Tanzu を使用している場合は、VMware Tanzu で Java をインストール (86ページ)をご覧ください。

RPM リポジトリを使用して Java エージェントをインストール

RPM リポジトリを使用して Java エージェントをインストールするには：

1. 以下のコマンドを使用して、Contrast RPM リポジトリからパッケージを取得するようシステムを設定します。

```
OSREL=$(rpm -E "%{rhel}")
sudo -E tee /etc/yum.repos.d/contrast.repo << EOF
[contrast]
name=contrast repo
baseurl=https://pkg.contrastsecurity.com/rpm-public/centos-$OSREL/
gpgcheck=0
enabled=1
EOF
```

2. 設定をしたら、Contrast Java エージェントをインストールします。

```
sudo yum install contrast-java-agent
```

3. Contrast Java エージェントの JAR ファイルが、`/opt/contrast/contrast-agent.jar` として、インストールされます。
4. Java エージェントを設定 (103ページ)します。YAML 設定ファイル (63ページ)を作成またはダウンロードします。設定ファイルには、エージェントキー (61ページ)を使用して、Contrast の接続パラメータを指定する必要があります。
5. エージェントがファイルを認識できるように YAML 設定ファイル(`contrast.yaml`)の場所を指定します。 以下の例では、`<YourContrastJarPath>`を Contrast JAR ファイルのパス(これは組織内のファイル構造やダウンロードした方法によって異なります)に、`<ApplicationJar>`をアプリケーションの JAR ファイル名に置き換えます。

```
java -javaagent:<YourContrastJarPath> -
Dcontrast.config.path=contrast.yaml -jar <ApplicationJar>.jar
```

注記
YAML の代わりにシステムプロパティや環境変数を使って設定する、またはエージェントが自動的に検索できる標準の場所 (62ページ)に YAML 設定ファイルを置いている場合は、Java エージェントを含めるよう JVM パラメータを設定してください。

```
java -javaagent:<YourContrastJarPath> -jar <AppName>.jar
```

6. 通常通りにアプリケーションを使用します(例えば、アプリケーションの Web インターフェイスをクリックして、API コマンドを送信するなど)。Contrast でアプリケーションが認識されていることを確認します(例えば、Contrast の Web インターフェイスでアプリケーションを参照したり、ログを表示するなど)。

また、以下のようなアプリケーションサーバと一緒に Contrast エージェントをインストールすることで、テスト環境や本番環境で動作するアプリケーションのセキュリティ分析を行うことができます。

- Glassfish
- Jetty (98ページ)
- JBoss/Wildfly (96ページ)
- Tomcat (98ページ)
- WebLogic (99ページ)
- WebSphere (100ページ)

Docker などのコンテナを使用してインストール (81ページ)することもできます。

ヒント
エージェントのインストールで互換性があるその他の方法については、Contrast サポートポータルを参照してください。VMware Tanzu を使用している場合は、VMware Tanzu で Java をインストール (86ページ)をご覧ください。

コンテナを使用して Java エージェントをインストール

本項では、Docker を例として、コンテナ化されたアプリケーションに Contrast Java エージェントをインストールするための一般的な手順について説明します。

注記
エージェントの起動に時間がかかる場合は、Java Agent Effects on Startup Performance(Java エージェントが起動パフォーマンスに及ぼす影響)および Java agent with Docker(Docker 環境での Java エージェント)に、この問題の解決の参考になる情報があります。

インストールを行う前に

コンテナや関連ソフトウェアの仕組みを基本的に理解している必要があります。実際の手順は、ご利用の環境に合わせて調整してください。

ECS のサポート

Amazon ECS(Amazon Elastic Container Service)環境で Docker コンテナを使用する場合、この手順を使用して Contrast Java エージェントをインストールすることができます。

手順 1：エージェントをインストール

Contrast エージェントは、アプリケーションをコンテナイメージに追加する前または後のどちらでも追加できます。推奨される方法は、名前を付けたマルチステージビルドを使用することです。例：

```
FROM eclipse-temurin:17

# Hidden for brevity...

# Copy the required agent files from the official Contrast agent image.
COPY --from=contrast/agent-java:latest /contrast/contrast-agent.jar /opt/
contrast/contrast.jar
```

この例では、最新の Java エージェントが使用されます。使用可能なタグは、DockerHub を確認して下さい。

手順 2：エージェントを設定

Java エージェントをコンテナにインストールする場合は：

- 共通の設定には **YAML 設定ファイル**を使用して、ベースイメージで指定できるようにします。例えば、共通の設定には、ログをコンソール出力にリダイレクトすることや、プロキシの設定、パフォーマンスのチューニングなどがあります。
  Contrast エージェント設定エディタ (64ページ)を使用すると、エージェントを正しく設定することができます。
  YAML 設定ファイルを作成し、ベースイメージにコピーします。次の行を追加して、この YAML 設定ファイルをベースイメージの Dockerfile にコピーします。

  ```
  COPY WORKSPACE/contrast_security.yaml /opt/contrast/contrast_security.yaml
  ```
- アプリケーション固有の設定値には、**Java のシステムプロパティ**や**環境変数**を使用して、各アプリケーションのオプションをそれぞれ指定します。

Contrast の設定	機能	Java システムプロパティ	
アプリケーションのメタデータ	アプリケーションに関連付けるメタデータを指定 指定する前にアプリケーションメタデータを作成 (830ページ)してください。	`-Dcontrast.application.metadata`	CONTRAST__AP
アプリケーションのセッションメタデータ	ビルド番号、バージョン、ハッシュなどの情報でセッションの新規作成時に使用されるセッションメタデータ (535ページ)を指定	`-Dcontrast.application.session_metadata`	CONTRAST__AP
アプリケーショングループ	このアプリケーションを関連付けるアクセスグループをオンボード時に指定(アプリケーションのアクセスグループ (822ページ)は先に Contrast で作成しておく必要あり)	`-Dcontrast.application.group`	CONTRAST__AP

Contrast の設定	機能	Java システムプロパティ	
サーバの環境	アプリケーションを実行する環境を指定、このオプションで有効な値：Development、QA、Production	`-Dcontrast.server.environment`	CONTRAST__SE

手順 3：JVM パラメータを更新

Java アプリケーションにプロファイラをアタッチするために、`JAVA_TOOL_OPTIONS` 環境変数を設定して、アプリケーションに`-javaagent` オプションを指定する必要があります。

Contrast 共通の JVM パラメータをベースイメージ内の別の環境変数にあらかじめ設定しておき、アプリケーションチームがそれを `JAVA_TOOL_OPTIONS` で利用できるようにします。 例：

- ベースイメージの Dockerfile：

```
ENV CONTRAST_OPTS "-javaagent:/opt/contrast/contrast.jar \
-Dcontrast.config.path=/opt/contrast/contrast_security.yaml"
```

- アプリケーションイメージの Dockerfile：

```
ENV JAVA_TOOL_OPTIONS $CONTRAST_OPTS \
-Dcontrast.application.metadata=bU=<value>,contactEmail=<value>,contactNam
e=<value> \
-Dcontrast.application.group=APP_GROUP
```

手順 4：アプリケーションイメージを実行

Docker イメージにエージェントを追加 (82ページ)して設定 (82ページ)したら、イメージを実行します。

エージェントが Contrast サーバにデータを送信するには、エージェントの認証情報 (61ページ)が必要です。 エージェントの認証情報を保護するために、Docker secret を利用して、デプロイ時に環境変数として渡すことができます。以下は、Docker run コマンドの例です。

```
docker run -e CONTRAST__API__URL=https://app.contrastsecurity.com -e \
CONTRAST__API__API_KEY=<value> -e CONTRAST__API__SERVICE_KEY=<value> -e \
CONTRAST__API__USER_NAME=<value> -e CONTRAST__SERVER__NAME=<value> -e \
CONTRAST__SERVER__ENVIRONMENT=<value> image_with_contrast
```

Contrast が実行されているかを確認するには、コンテナのログをチェックしてください。次のようなメッセージが表示されているはずです。

```
2020-05-28 22:36:29,910 [main STDOUT] INFO - Copyright: 2019 Contrast \
Security, Inc
2020-05-28 22:36:29,910 [main STDOUT] INFO - Contact: \
support@contrastsecurity.com
2020-05-28 22:36:29,910 [main STDOUT] INFO - License: Commercial
2020-05-28 22:36:29,910 [main STDOUT] INFO - NOTICE: This Software and the \
patented inventions embodied within may only be used as part of
2020-05-28 22:36:29,910 [main STDOUT] INFO - Contrast Security's \
commercial offerings. Even though it is made available through public
2020-05-28 22:36:29,910 [main STDOUT] INFO - repositories, use of this \
Software is subject to the applicable End User Licensing Agreement
2020-05-28 22:36:29,910 [main STDOUT] INFO - found at https://
www.contrastsecurity.com/enduser-terms-0317a or as otherwise agreed between
2020-05-28 22:36:29,910 [main STDOUT] INFO - Contrast Security and the End \
User. The Software may not be reverse engineered, modified,
2020-05-28 22:36:29,910 [main STDOUT] INFO - repackaged, sold, \
redistributed or otherwise used in a way not consistent with the End User
```

```
2020-05-28 22:36:29,910 [main STDOUT] INFO - License Agreement.
[Contrast] Thu May 28 22:36:30 EDT 2020 Effective instructions: \
Assess=false, Protect=true
[Contrast] Thu May 28 22:36:30 EDT 2020 String Supporter has been disabled
[Contrast] Thu May 28 22:36:30 EDT 2020 Logging security messages to /Users/
usernamehere/.contrast/security.log
[Contrast] Thu May 28 22:36:31 EDT 2020 Starting JVM [1862ms]
```

関連項目

Contrast エージェントオペレータ(Kubernetes オペレータ) (498ページ)

Contrast サポートポータル：AWS Fargate と Contrast エージェント、Docker 環境での Java エージェント

Docker 利用の Gradle プロジェクトに Java エージェントをインストール

ここでは、アプリケーションプラグインと Docker プラグインを含む Gradle プロジェクトをサンプルとして使用し、Java の Web アプリケーションを構築します。 また、Web アプリケーションの動作を検証する JUnit5 の統合テストも実行します。プロセスの一環として、統合テスト時に Contrast Assess でコードが解析されるよう、テストに使用する Docker イメージに Contrast を組み込む手順を解説します。サンプルについては、GitHub リポジトリの Gradle プロジェクトの例を参照してください。

注記

以下の手順において、パッケージングや配布に関して言及している部分はいかなる形式でも、全て組織内での使用を目的としています。Contrast をアプリケーションや Docker コンテナと一緒に組織外に配布しないでください。詳細については、Contrast のサービス利用規約を参照してください。

Docker を使用する既存の Gradle プロジェクトに Contrast Java エージェントを追加する手順：

1. コマンドプロンプトを開き、以下のコマンドを実行して **contrast-java-examples** リポジトリをクローンします。

   ```
   $ git clone https://github.com/Contrast-Security-OSS/contrast-java-
   examples.git
   ```

2. `gradle-docker` ディレクトリに移動します。

   ```
   $ cd contrast-java-examples/gradle-docker
   ```

3. テストビルドを実行し、正常に機能することを確認します。

   ```
   $ ./gradlew build

   BUILD SUCCESSFUL in 3s
   4 actionable tasks: 3 executed, 1 up-to-date
   ```

   注記

   Windows では、代わりに `gradlew.bat build` を実行してください。

4. テストビルドがうまくいかない場合は、Java 11 が正しくインストールされていることを確認して下さい(サンプルアプリケーションをビルドするには、Java 11 以降が必要です。Java エージェントのサポート対象 (75ページ)ページで、Contrast の Java エージェントでサポートされる Java のバージョンを参照できます。)。

```
$ java -version
openjdk version "11.0.18" 2023-01-17
OpenJDK Runtime Environment Temurin-11.0.18+10 (build 11.0.18+10)
OpenJDK 64-Bit Server VM Temurin-11.0.18+10 (build 11.0.18+10, mixed \
mode)
```

5. 変更を加えたら、再度ビルドを実行します。それでも機能しない場合は、この問題の詳細を記載して問題を報告してください。
6. エージェントキー (61ページ)を使用して、Contrast エージェントが Contrast サーバと通信するための設定をします。以下のキーが必要です。
   - Contrast URL：`https://app.contrastsecurity.com/Contrast` か、オンプレミス版かプライベートクラウドの URL になります。
   - 組織用 API キー
   - エージェントユーザ名
   - エージェントサービスキー
7. Gradle のユーザホームディレクトリにある `gradle.properties` ファイルに、Gradle のプロパティとしてキーを追加します。このファイルが存在しない場合は、作成して下さい。
   `<contrast_url>`、`<your_api_key>`、`<agent_user_name>`、および `<agent_user_service_key>`を Contrast URL、API キー、ユーザ名、およびサービスキーの値に置き換えます。

```
contrastUrl=<contrast_url>
contrastAgentUserName=<agent_user_name>
contrastAgentServiceKey=<agent_user_service_key>
contrastApiKey=<your_api_key>
```

8. `build.gradle` の `createDockerfile` タスクを修正して、Contrast エージェントを追加し、それを使用するようにアプリケーションを設定します。

```
task createDockerfile(type: Dockerfile) {
  // ... rest of block omitted

  copyFile(new Dockerfile.CopyFile("/contrast/contrast-agent.jar", "/
contrast.jar").withStage("contrast/agent-java:latest"))
  environmentVariable("JAVA_TOOL_OPTIONS", "-javaagent:/contrast.jar")
}
```

9. `build.gradle` の `createContainer` タスクに以下のコマンドを追加し、Contrast の設定変数をコンテナに渡します。

```
task createContainer(type: DockerCreateContainer) {
    // ... rest of the config omitted

    envVars = [
        CONTRAST__API__URL: project.property("contrastUrl"),
        CONTRAST__API__USER_NAME: \
project.property("contrastAgentUserName"),
        CONTRAST__API__SERVICE_KEY: \
project.property("contrastAgentServiceKey"),
        CONTRAST__API__API_KEY: project.property("contrastApiKey"),
        CONTRAST__APPLICATION__NAME: "${project.name}-how-to"
    ]
}
```

10. ビルドを再度実行します。

```
./gradlew clean build
```

> **注記**
> Windows では、代わりに `gradlew.bat clean build` を実行します。

これで、Docker コンテナで Contrast を有効にしてアプリケーションを実行できるようになりました。総合テストを実行すると、脆弱なエンドポイントが検出され、Contrast に報告されます。報告された脆弱性を参照するには、Contrast Web インターフェイスで脆弱性ページ (692ページ)にアクセスし、アプリケーション名の **gradle-application-how-to** でフィルターをかけます。

VMware Tanzu Application Service で Java エージェントをインストール

VMware Tanzu(Application Service、旧 Pivotal Cloud Foundry)は、コンテナ化された独自の SaaS(Software as a Service)環境です。VMware がリリースする Java ビルドパックにより、Contrast の Java エージェントにアクセスできるようになります。Java アプリケーションを実行するコンテナにビルドパックをインストールしてください。

Contrast サービス

バインドされた Contrast サービスが存在すれば、Java エージェントがアクティブになり、ダウンロードされます。Contrast サービスが存在すると判定されるのは、VCAP_SERVICES ペイロードに、contrast-security を部分文字列として含むサービス名、ラベル、またはタグがある場合です。Contrast サービスは、以下のいずれかの方法を使用して作成します。

- **ユーザー提供サービス**：ユーザー提供サービスは、単一のアプリケーションを Java エージェントにバインドし、認証を設定する簡単な方法です。
- **サービスブローカー(Contrast タイル)**：サービスブローカーを使用すると、複数のアプリケーションをバインドして、Java エージェントへのアクセスと認証ができます。

Contrast サービスがアプリケーションにバインドされると、Contrast を起動する(JVM に `javaagent` フラグを立てる)ために必要な文字列が提供され、Contrast Web インターフェイスへの認証が提供されます。

Java ビルドパック

Java ビルドパックには、コンテナで Java エージェントをダウンロードして設定するために必要な手順と設定情報が含まれています。オフラインまたはオンラインのビルドパックを使用することができます。

- オフラインのビルドパックは、通常、カスタマイズを行なった GitHub リポジトリからフォークされます。このようなリポジトリには、古いバージョンのエージェントが含まれている可能性があります。
- オンラインのビルドパックは、通常、最新バージョンであり、必要に応じて GitHub からプルされます。

必要条件

- ビルドパック
  VMware Tanzu Network の環境でアプリケーションに Contrast エージェントを組み込むには、以下のいずれかのビルドパックをアプリケーションで使用する必要があります。
  - Cloud Foundry Java ビルドパック、バージョン 3.19 以降またはバージョン 4.2 以降
  - IBM Liberty ビルドパック、バージョン 2.7.0.2 以降
- サービスの作成時に指定した `contrast-security` を含む名前またはタグ。
- 認証情報のペイロードに基本の YAML オプションも含める必要があります。

環境変数で設定値を指定する方法など、ビルドパックの設定に関する一般的な情報については、Cloud Foundry Java ビルドパックのドキュメントの Configuration and Extension(設定と拡張)の項目を参照してください。

設定オプション

エージェントのフレームワークを設定するには、フォークしたビルドパックの `config/contrast_security_agent.yml` ファイルを変更します。このフレームワークは、リポジトリユーティリティのサポートを使用し、そこで定義されているバージョンの構文規則に従います。

名前	説明
`repository_root`	Contrast Security のリポジトリインデックスの URL
`version`	使用する Contrast エージェントのバージョン

使用する Java エージェントのバージョンを指定する場合は、環境変数 `JBP_CONFIG_CONTRASTSECURITYAGENT` を設定し、インデックスに記載されているバージョンを指定します。例：

```
JBP_CONFIG_CONTRASTSECURITYAGENT='version: 4.13.1'
```

例

ここでは、ユーザー提供サービスを作成し、それを `spring-petclinic` というアプリケーションにバインドする例について説明します。

1. 次のコマンドで、指定したビルドパックを使用して、アプリケーションを Cloud Foundry にデプロイします(指定しない場合は、その環境のデフォルトのビルドパックが使用されます)。

   ```
   cf push myApp -p target/spring-petclinic-2.4.2.jar \
     -b 'https://github.com/cloudfoundry/java-buildpack.git'
   ```
2. 次のコマンドで、ユーザー提供サービスを作成します。

   ```
   cf create-user-provided-service contrast-security-service -
   p "teamserver_url, username, api_key, service_key"
   ```

   `teamserver_url` の値には、プロトコルとホスト名のみを指定してください。`/Contrast/` や `/Contrast/api` は含めないでください。
3. サービスをアプリケーションにバインドするために、次のコマンドを実行します(このタスクは必須です)。

   ```
   cf bind-service myApp contrast-security-service
   ```
4. 次のコマンドで、Contrast に接続できるように再ステージングします(基本的に、コンテナを再起動します)。

   ```
   cf restage myApp
   ```

関連項目

VMware Tanzu の Contrast サービスブローカータイルを追加 (89ページ)

VMware Tanzu の Contrast サービスブローカーを追加 (87ページ)

Contrast サービスブローカーのプロキシを設定 (90ページ)

VMware Tanzu の Contrast サービスブローカーを追加

手順

1. 以下のコマンドを実行して、サービスブローカーアプリケーションをデプロイします。

   ```
   cf push contrast-security-service-broker
   ```

   サービスブローカーが PCF に表示されるはずです。
2. `CONTRAST_SERVICE_PLANS` 環境変数を使用して、プランを設定します(デフォルトでは、サービスブローカーにはプランがありません)。

Pivotal Ops Manager を使用して、環境変数を設定することもできます。IBM Cloud を使用している場合、アプリケーションのコンソールページで **Runtime** を選択したら **Environment Variables** を選択し、値を設定します。
例：コマンドラインから値を設定するには、以下の例を参考にしてください。

```
cf set-env contrast-security-service-broker CONTRAST_SERVICE_PLANS
    " {
            "ServicePlan1": {
                "name":"ServicePlan1",
                "teamserver_url":"https://yourteamserverurl.com",
                "username":"your_username",
                "org_uuid":"00000000-1111-2222-3333-000000000000",
                "api_key":"your_api_key",
                "service_key":"your_service_key"
            },
            "AnotherServicePlan":{
                "name":"AnotherServicePlan",
                "teamserver_url":"https://yourteamserverurl.com",
                "username":"your_username",
                "org_uuid":"00000000-1111-2222-3333-000000000001",
                "api_key":"your_api_key",
                "service_key":"some_other_service_key"
                }
            } "
```

IBM Cloud でエージェントを実行する場合、以下の例のように、CONTRAST_SERVICE_PLANS 環境変数の値は一重引用符で囲んでください。

```
cf set-env contrast-security-service-broker CONTRAST_SERVICE_PLANS
    " {
            'ServicePlan1': {
                'name':'ServicePlan1',
                'teamserver_url':'https://yourteamserverurl.com',
                'username':'your_username',
                'org_uuid':'00000000-1111-2222-3333-000000000000',
                'api_key':'your_api_key',
                'service_key':'your_service_key'
            },
            'AnotherServicePlan':{
                'name':'AnotherServicePlan',
                'teamserver_url':'https://yourteamserverurl.com',
                'username':'your_username',
                'org_uuid':'00000000-1111-2222-3333-000000000000',
                'api_key':'your_api_key',
                'service_key':'some_other_service_key'
                }
            } "
```

3. アプリケーションを再ステージングするために、以下のコマンドを実行します。

```
cf restage contrast-security-service-broker
```

4. ユーザ名とパスワードの環境変数を設定します。

```
cf set-env contrast-security-service-broker SECURITY_USER_NAME \
aSecureUsername
cf set-env contrast-security-service-broker SECURITY_USER_PASSWORD \
aSecurePassword
```

5. サービスブローカーのインスタンスを作成します。サービスプランを少なくとも 1 つ定義してください。ユーザ名とパスワードは、前の手順で設定したのと同じものを使用する必要があります。

```
cf create-service-broker contrast-security-service-broker USER_NAME \
PASSWORD
<URL of your application>
```

IBM Cloud の場合は、以下の例のように、コマンドの最後に`--space-scoped` を追加してください。

```
cf create-service-broker contrast-security-service-broker USER_NAME \
PASSWORD
<URL of your application> --space-scoped
```

6. 全てのサービスブローカーは最初はプライベートです。以下の例のように、パブリックに変更するためのコマンドを実行します。

```
cf enable-service-access contrast-security-service-broker
```

7. サービスブローカーが動作するようになったので、サービスのインスタンスを作成し、アプリケーションにバインドします。サービスのインスタンスを作成するには、以下のコマンドを実行します。

```
cf create-service contrast-security-service-broker ServicePlan1 \
<name_of_service>
```

8. サービスブローカーをアプリケーションにバインドするために、以下のコマンドを実行します。

```
cf bind-service <app_name> <name_of_service>
```

アプリケーションでエージェントが起動されるのを確認できるはずです。また、Contrast Web インターフェイスにアプリケーションが表示されているはずです。

関連項目

Contrast サービスブローカータイルを追加 (89ページ)

Contrast サービスブローカーのプロキシを設定 (90ページ)

VMware Tanzu の Contrast サービスブローカータイルを追加

Contrast を VMware Tanzu Network(旧 Pivotal Cloud Foundry)と連携させるには、Contrast サービスブローカータイルをインストールします。

手順

1. VMware Tanzu Network から Contrast サービスブローカータイルをダウンロードします。
2. ファイルをローカルに保存し、Pivotal Ops Manager にアクセスします。
3. Ops Manager で **Import a Product**(プロダクトをインポート)ボタンを選択して、ダウンロードした `contrast-security-service-broker-#.#.#.pivotal` タイルを選択します。
   ダウンロードしたファイルの拡張子が ZIP の場合は、ファイル名を `contrast-security-service-broker-#.#.#.pivotal` に変更してください。
4. タイルをデプロイするには、設定がいくつか必要です。デフォルトでは、サービスブローカーにはサービスプランがありません。Contrast サービスブローカータイルをデプロイする前に、少なくともプランを 1 つ追加する必要があります。
   サービスプランを追加するには、Contrast サービスブローカータイルの **Service Plans**(サービスプラン)を選択し、**Add**(追加)ボタンをクリックします。
5. サービスプランに以下の設定パラメータを指定します。
   - **TeamServer**：Contrast サーバへの URL
   - **TeamServer Service Key**：組織の設定にあるサービスキー (61ページ)

- **TeamServer API Key：** 組織の設定にある API キー (61ページ)
- **Organization UUID：** アプリケーションが存在する組織の組織 ID (61ページ)
- **Username：** Contrast ユーザ名
- **Plan Name：**Apps Manager で表示されるプラン名
- **Proxy Host：** サービスブローカーが Contrast と通信するプロキシのホスト名
- **Proxy Port：**プロキシのポート
- **Proxy Username：** プロキシのユーザ名(認証が必要な場合)
- **Proxy Password：** プロキシのパスワード(認証が必要な場合)

> **注記**
> タイルのプロキシ設定に加えて、エージェントのプロキシ通信 (90ページ)も設定する必要があります。

6. **保存**を選択します。
   アプリケーションを別の組織に入れたい場合は、必要となる他のプランを定義してください。
7. ダッシュボードで **Apply Changes**(変更を適用)を選択します。
   この処理が完了するまでに時間がかかる場合があります。
8. サービスブローカーをデプロイしたら、アプリケーションに認証情報をバインドできます。
   Marketplace(マーケットプレイス)にアクセスし、"Contrast Security service broker"(サービスブローカー)を検索します。
9. Pivotal Ops Manager で、"Contrast service broker"オプションを選択すると、作成した利用可能なプランが表示されます。
10. **Select this Plan**(このプランを選択)をクリックして、アプリケーションにバインドするプランを選択します。
11. プランの Instance Name(インスタンス名)を指定します。
    これは、サービスブローカーには影響しません。インスタンスには好きな名前を使用することができます。
12. Bind to App(アプリにバインド)ドロップダウンメニューで、このサービスにバインドするアプリケーションを選択します。そして、アプリケーションを再ステージングします。
    これにより、Contrast から最新のエージェントが取得され、アプリケーションにエージェントが組み込まれます。
13. **オプション：**アプリケーション名などのエージェントのプロパティを上書きしたい場合は、以下の例のようにコマンドを使用して PCF で環境変数を設定することができます。

```
cf set-env APP-NAME JAVA_OPTS " -
Dcontrast.agent.java.standalone_app_name=PivotalSpringApp"
```

関連項目

Contrast サービスブローカーを追加 (87ページ)

Contrast サービスブローカーのプロキシを設定 (90ページ)

Contrast サービスブローカーにエージェントのプロキシ通信を設定

VMware Tanzu で Java エージェントをデプロイする場合に、Contrast サービスブローカータイル (89ページ)の追加時にプロキシを設定することを選択できます。Contrast サービスブローカーが、サービスプラン内に設定されたプロキシ構成を使用するように指定して、Contrast とのバインドを完了することができます。

Contrast サービスブローカータイルの追加時にプロキシを設定する場合、本項で説明するように、エージェントのプロキシ通信も設定する必要があります。これは、アプリケーションごとに設定することも、デプロイされた全てのアプリケーションが使用するように組織レベルで設定することもできます。

手順

1. アプリケーションごとにプロキシ通信を設定する場合は、以下のコマンドを使用します。

```
cf set-env $APP_NAME CONTRAST__API__PROXY__ENABLE "true"
cf set-env $APP_NAME CONTRAST__API__PROXY__URL "scheme://host:port"
```

   または、以下のコマンドを使用できます。

```
cf set-env $APP_NAME JAVA_OPTS "-Dcontrast.api.proxy.enable=true -
Dcontrast.api.proxy.url=scheme://host:port"
```

2. 組織レベルでプロキシ通信を設定するには、以下のコマンドを使用します。

```
cf ssevg '{"CONTRAST__API__PROXY__ENABLE": "true"}'
cf ssevg '{"CONTRAST__API__PROXY__URL": "scheme://host:port"}'
```

関連項目

Contrast サービスブローカータイルを追加 (89ページ)

Contrast サービスブローカーを追加 (87ページ)

AWS Elastic Beanstalk で Java エージェントをインストール

本項での手順を使用して、Java エージェントを設定し、AWS Elastic Beanstalk と連携させることができます。Contrast の Java エージェントをダウンロードして、アプリケーションに組み込んで検査をするために、`.ebextensions` ファイルをどのように作成するかを説明します。

ご利用の環境によっては、本項の手順をカスタマイズする必要があります。

この手順は、DevOps の実践と Docker の仕組みに関して知識があるユーザを対象としています。

開始する前に

- ご利用の Java のツールや環境 (75ページ)が Contrast でサポートされていることを確認すること。
- Contrast の Java エージェントが Contrast サーバに接続する (77ページ)ための必要な情報があること。
- Contrast の Java エージェントをダウンロードして起動したことがあること。
- カスタマイズした設定ファイル`.ebextensions` をインストールするための Beanstalk 環境へのアクセスがあること。

手順 1：Contrast Java エージェントをダウンロードする設定を指定

設定ファイル`.ebextensions` の files セクションに、リモート URL から Contrast エージェントをダウンロードするよう指定します。以下は、Maven リポジトリから Contrast エージェントをダウンロードするための設定の例です。

```
files:
  "/opt/contrast/contrast.jar":
    mode: "000755"
    owner: rootCorporate rule
    group: root
    source: "https://repository.sonatype.org/service/local/artifact/maven/
redirect?r=central-proxy&g=com.contrastsecurity&a=contrast-agent&v=LATEST"
```

Contrast エージェントは`/opt/contrast` に置くことを推奨しますが、必要であれば別の場所を使用することもできます。内部リポジトリからエージェントをダウンロードするように URL を変更することもできます。ビルド時に、希望のエージェントバージョンを指定して、Maven リポジトリからダウンロードできます。

手順 2：Contrast エージェントの設定ファイルを作成

Contrast エージェントの設定には、さまざまな値を使用できます。設定する値には有効になる優先順位 (62ページ)があります。有効になる設定値は、以下の順序で決まります。

1. 社内規定(例、ライセンスの期限切れによる無効化など)
2. システムプロパティ
3. 環境変数
4. YAML 設定ファイル
5. Contrast Web インターフェイスで設定されている値
6. Contrast Security が設定したデフォルト値

エージェントの設定ファイルを作成する方法として、共通の設定とアプリケーション固有の設定を組み合わせて使用することをお勧めします。

- **共通の設定**： 基本となる一連の設定を YAML ファイルに指定します。例：
  - ログをコンソール出力にリダイレクト
  - プロキシの設定(プロキシがある場合)
  - エージェントのアクティビティを制限するパフォーマンスチューニングオプション

  以下は、`.ebextensions` 設定ファイルの一例で、デプロイ時に Contrast エージェントの YAML ファイルを作成し設定する方法を示します。

```
files:
 "/var/contrast/contrast_security.yaml" :
   mode: "000755"
   owner: root
   group: root
   content: |
     api:
       proxy:
         url: https://host:port
     agent:
       java:
         scan_all_classes: false
         scan_all_code_sources: false
       logger:
         stdout: true
```

- **アプリケーション固有の設定**：この方法によって、アプリケーションごとに追加のオプションを指定できます。以下の環境変数を使用します。
  - **アプリケーションのメタデータ**：アプリケーションに関連付けるユーザ定義のメタデータを指定

    ```
    CONTRAST__APPLICATION__METADATA
    ```
  - **アプリケーション名**：Contrast サーバに報告されるアプリケーション名を指定

    ```
    CONTRAST__APPLICATION_NAME
    ```
  - **アプリケーションのセッションメタデータ**：ビルド番号、バージョン、ハッシュなどの情報でセッションの新規作成時に使用されるメタデータを指定

    ```
    CONTRAST__APPLICATION__SESSION_METADATA
    ```

    注記
    詳細については、セッションメタデータの設定 (537ページ)をご覧ください。
  - **アプリケーションのグループ**：アプリケーションを Contrast に追加した時に関連付ける、アプリケーションアクセスグループを指定。アプリケーションアクセスグループは、使用する前に先に作成しておく必要があります。

```
CONTRAST__APPLICATION__GROUP
```

- **サーバの環境**：アプリケーションを実行する環境を指定。このオプションで有効な値：development、qa、 production

```
CONTRAST__SERVER__ENVIRONMENT
```

例 1：環境の作成時に環境変数を設定する方法

```
eb create <environment name> --envvars CONTRAST__API__URL=https://
app.contrastsecurity.com/
Contrast,CONTRAST__API__API_KEY=<value>,CONTRAST__API__SERVICE_KEY=<value>
,CONTRAST__API__USER_NAME=<value>,CONTRAST__SERVER__NAME=<value>,CONTRAST_
_SERVER__ENVIRONMENT=<value>
```

例 2：環境を作成した後に環境変数を設定する方法

```
eb setenv CONTRAST__API__URL=https://app.contrastsecurity.com/Contrast \
CONTRAST__API__API_KEY=<value> CONTRAST__API__SERVICE_KEY=<value> \
CONTRAST__API__USER_NAME=<value> CONTRAST__SERVER__NAME=<value> \
CONTRAST__SERVER__ENVIRONMENT=<value>
```

手順 3：JVM パラメータを更新

Java アプリケーションにプロファイラをロードするために、アプリケーションに`-javaagent` オプションを渡す必要があります。これを行うには、`JAVA_TOOL_OPTIONS` 環境変数を設定します。

これらの変数は、アプリケーション固有の環境変数を設定するのと同じ方法で設定します。以下の例に示すように、エージェントの JAR ファイルと YAML 設定ファイルのパスを使用します。

```
eb setenv JAVA_TOOL_OPTIONS="-javaagent:/opt/contrast/contrast.jar -
Dcontrast.config.path=/var/contrast/contrast_security.yaml"
```

手順 4：`.ebextensions` の設定を使用してアプリケーションをデプロイ

AWS では、Beanstalk のカスタマイズ設定は、デプロイフォルダのルートの .ebextensions フォルダ内に設定ファイルがあることが前提とされています。以下は、`.ebextensions` フォルダを含むディレクトリ構成の例です。Contrast エージェントのダウンロードと YAML 設定が含まれた `contrast.config` ファイルがあります。

```
.ebextensions
   contrast.config
application.jar
```

Linux で自動更新を使用して Java エージェントをインストール

ユーザによっては、Contrast の Java エージェントを自動的に最新バージョンに更新したい場合があります。Linux ユーザの場合、一般的な Linux ツールである `cron` や `curl` を使用して、Maven Central から Java エージェントの更新をスケジュールできます。

ここでは、Ubuntu 18.04 Linux ホストで Java エージェントの更新ジョブをスケジュールするよう設定する方法について説明します。

注記

本項で説明する内容のファイルの作成には、お好きなエディタを使用してください。以下の例では、`tee` コマンドを使用してファイルを作成します。

1. ここでの手順に沿って各ステップを実行したい場合は、Vagrant や VirtualBox を使用して、Ubuntu 18.04 の仮想マシンを新規に作成することもできます。

```
vagrant init ubuntu/bionic64
```

```
vagrant up
```

```
vagrant ssh
```

2. Contrast ソフトウェア用の共有ディレクトリを作成します。

```
sudo mkdir -p /opt/contrast
```

3. 最新の Java エージェントをインストールするためのスクリプトを*/etc/cron.daily* ディレクトリに作成します。このディレクトリ内のスクリプトは毎日 1 回実行され、その結果、ホストで毎日 Java エージェントが最新に更新されます。
4. `tee` を使用してこのスクリプトを作成します。全ての行を入力し終わったら、`CTRL+D` を押します。

```
$ sudo tee -a /etc/cron.daily/install-latest-contrast-agent > /dev/null
#!/bin/bash -u

CONTRAST_DIRECTORY=/opt/contrast
CONTRAST_FILE_NAME=contrast-agent.jar

CONTRAST_VERSION=$(curl --fail --silent 'https://search.maven.org/
solrsearch/select?q=g:com.contrastsecurity+a:contrast-agent' | sed -e 's/
[{}]/''/g' | sed s/\"//g | awk -v RS=',' -F: '$1=="latestVersion"{print \
$2}' | grep -v -e '^$')
curl --fail --silent --location "https://repo1.maven.org/maven2/com/
contrastsecurity/contrast-agent/${CONTRAST_VERSION}/contrast-agent-$
{CONTRAST_VERSION}.jar" -o "contrast-agent-${CONTRAST_VERSION}.jar"
if [ $? -ne 0 ]; then
    echo "Failed to download Contrast Java agent" >&2
    exit 1
fi
mv /tmp/$CONTRAST_FILE_NAME $CONTRAST_DIRECTORY/$CONTRAST_FILE_NAME
```

5. 新しいスクリプトファイルに実行権限を設定します。

```
sudo chmod +x /etc/cron.daily/install-latest-contrast-agent
```

6. スクリプトをテストするために、スクリプトを実行します。`stat` を使用してファイルが存在することを確認します。

```
$ sudo /etc/cron.daily/install-latest-contrast-agent
$ stat /opt/contrast/contrast-agent.jar
stat /opt/contrast/contrast-agent.jar
  File: /opt/contrast/contrast-agent.jar
  Size: 10568283        Blocks: 20648      IO Block: 4096   regular file
Device: 801h/2049d      Inode: 256034      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2019-04-11 02:02:01.265775928 +0000
Modify: 2019-04-11 02:24:47.849796936 +0000
Change: 2019-04-11 02:24:47.849796936 +0000
 Birth: -
```

7. Contrast エージェントには、Contrast サーバと通信するための設定が必要です。エージェントのキー情報はこちら (61ページ)を参照してください。
8. Contrast エージェントを Linux ホストにインストールしたら、通常は、ホスト上で Contrast が有効な各 Web アプリケーションが、Contrast サーバへの接続に必要なパラメータなどの基本的な設定値を共有できるように構成します。通常、Contrast は Linux ホスト上のパス*/etc/contrast/java/contrast_security.yaml* にある YAML ファイルの設定を探します。

9. */etc/contrast/java* ディレクトリを作成してください。

```
sudo mkdir -p /etc/contrast/java
```

10. `tee` コマンドを使用して、設定ファイルを作成します。`<contrast_url>`、`<your_api_key>`、`<agent_user_name>`、`<agent_user_service_key>`を、前述のステップで Contrast から取得した値に置き換えます。

```
$ sudo tee -a /etc/contrast/java/contrast_security.yaml > /dev/null
api:
  url: <contrast_url>
  api_key: <your_api_key>
  user_name: <agent_user_name>
  service_key: <agent_user_service_key>
```

11. 全ての行を入力し終わったら、`CTRL+D` を押してください。
12. Contrast がインストールされ、正しく設定されていることを確認するために、診断テストを実行します。診断テストを実行するには、ホストに Java がインストールされている必要があります。

```
sudo apt install --yes openjdk-11-jre-headless
```

13. 最後に、Java エージェントの診断テストを実行します。エージェントが正しくインストールされ、*/etc/contrast/java/contrast_security.yaml* の設定パラメータを使用して、Contrast サーバと通信できることを確認します。

```
$ java -jar /opt/contrast/contrast-agent.jar diagnostic
*** Contrast Agent (version 3.6.3-SNAPSHOT)
[!] Attempting to connect to the Contrast TeamServer at https://
apptwo.contrastsecurity.com/Contrast (No proxy).
[!] Attempting to resolve domain: apptwo.contrastsecurity.com
        Resolved domain apptwo.contrastsecurity.com to IP Address \
52.200.215.12
[+] Client successfully resolved the DNS of the Contrast TeamServer. No \
proxy needed.
[!] Issuing HTTP request to Contrast...
        Executing request...
        Reading response [200]
        Response size = 4209
        Snippet: <!doctype html> <!--[if gt IE 8]><!--> <html class="no-
js" i
[+] Client can connect directly to the Contrast TeamServer. No proxy \
needed.
```

Scala

Contrast Java エージェントを使用して、Contrast Assess や Contrast SCA で、Scala ベースのアプリケーションを解析することができます。

Java エージェントは、従来のアプリケーションサーバで構築される Scala Web アプリケーションや、Play フレームワークなどで構築されるような新しい Scala Web アプリケーションを解析できます。JVM があれば、Scala に組み込んだ Java エージェントがセキュリティ情報を提供します。

アプリケーションが実行されると、Java エージェントのセンサーがアプリケーションのセキュリティ、アーキテクチャ、ライブラリに関する情報を収集します。エージェントによる解析の結果は、Contrast Web インタフェースで確認できます。

Scala のエージェントは次の機能を提供します。

- ルートカバレッジ
- フローマップ

- SCA ライブラリの検出

Kotlin

Contrast Java エージェントを使用して、Contrast Assess や Contrast SCA で、Kotlin のアプリケーションを解析することができます。

Java エージェントは、従来のアプリケーションサーバで構築された Kotlin の Web アプリケーションや、SpringBoot などの新しい Kotlin のサーバサイドアプリケーションを解析できます。

JVM があれば、Kotlin に組み込む Java エージェントがセキュリティ情報を提供します。アプリケーションが実行されると、Java エージェントのセンサーがアプリケーションのセキュリティ、アーキテクチャ、ライブラリに関する情報を収集します。エージェントによる解析の結果は、Contrast Web インタフェースで確認できます。

Contrast Java エージェントを使用する場合と同様に、アプリケーションを実行します。自動的に Kotlin がサポートされます。

Java アプリケーションサーバ

注記

本ドキュメントでは、サポート対象の内容について説明します。アイコンが付いたリンクは、参考として、他のドキュメントを参照するリンクとなっています。

次のアプリケーションサーバを使用できます。

- Axis2
- Glassfish
- JBoss / Wildfly (96ページ)
- Jetty (98ページ)
- Tomcat (98ページ)
- Weblogic (99ページ)
- WebSphere (100ページ)

関連項目

- Java エージェントのインストール (77ページ)
- Java エージェントのサポート対象テクノロジ (75ページ)
- Java エージェントの設定 (103ページ)

JBoss EAP、JBoss AS、WildFly で Java エージェントを設定する

注意

バージョン番号を混同しないように注意してください。バージョン 7 より前の JBoss EAP は、JBoss AS をベースにしています。JBoss EAP 7.x は、WildFly をベースにしています。

Java エージェントを使用して JBoss を実行

1. 次のいずれかのリポジトリから、Contrast Java エージェント(JAR ファイル)をダウンロードします。
   - Maven Central (78ページ)
   - Debian (79ページ)
   - RPM (80ページ)
2. JBoss は、BAT ファイルから実行するか、ドメインモードで実行できます。
   - **BAT ファイル：***.conf* ファイルを使用して bat ファイル(*domain.bat*、*standalone.bat*、または *run.bat*)から JBoss を実行する場合は、設定ファイル(.conf ファイル)を変更します。Contrast JVM パラメータを有効にして、起動スクリプトに返す必要があります。
     この変更を行うには、<YourContrastJarPath>を Contrast JAR (75ページ)ファイルへのパスに置き換え、ご利用の環境の JBoss サーバディレクトリを指定します。そして、*.conf* ファイルの最後の行にその行を追加します。
     - **Windows：**

       ```
       set "JAVA_OPTS=-javaagent:<YourContrastJarPath> %JAVA_OPTS%"
       ```
     - **UNIX：**

       ```
       JAVA_OPTS="-javaagent:<YourContrastJarPath> $JAVA_OPTS"
       ```
   - **ドメインモード：***domain.bat* または *domain.sh* を使用して、JBoss 6 EAP または JBoss AS 7.x をドメインモードで実行する場合は、*$JBOSS_HOME/domain/configuration/domain.xml* の JVM オプションに`-javaagent` スイッチを追加する必要があります。
     この例では、<YourContrastJarPath>を Contrast JAR (75ページ)ファイルへのパスに置き換えてください。

     ```
     <server-group ...>
       <jvm name="default">
         <jvm-options>
           <option value="-javaagent:<YourContrastJarPath>"/>
         </jvm-options>
       </jvm>
     ...
     </server-group>
     ```

Java 2 セキュリティマネージャでの WildFly の使用

Java 2 セキュリティ (164ページ)で WildFly を使用する場合、Java エージェントを設定できます。WildFly のバージョン 9 から 20 まで対応しています。WildFly 8 はサポートされていません。

Wildfly で Java 2 セキュリティマネージャを有効にするには：

1. コマンドライン引数`-secmgr` を渡すか、環境変数 SECMGR を true に設定します。

   ```
   SECMGR="true"
   ```
2. Java エージェントの権限を有効にするには、以下の Contrast ポリシーを*$JAVA_HOME/jre/lib/security/java.policy* (JDK 6-8 の場合)、または*$JAVA_HOME/lib/security/default.policy* (JDK 9 以降の場合)に追加します。<YourContrastJarPath>は Contrast JAR (75ページ)ファイルへのパスに置き換えて、以下のコマンドを使用します。

   ```
   grant codeBase "file:<YourContrastJarPath>" {
     permission java.security.AllPermission;
   };
   ```
3. エージェントが Wildfly のクラスローダーシステムで機能するようにするには、環境変数 `JBOSS_MODULES_SYSTEM_PKGS`(本来は `org.jboss.byteman`)の値を変更して、Java エージェントのベースパッケージ `com.contrastsecurity.agent`,`org.jboss.byteman` も含めるようにします。

ヒント
詳細については、Java EE 7 セキュリティマネージャを WildFly で使用する方法についてや、デフォルトのポリシー実装やポリシーファイルの構文を参照して下さい。

Jetty で Java エージェントを設定する

Jetty で Java エージェントを設定するには：

1. 次のいずれかのリポジトリから、Contrast Java エージェント(JAR ファイル)をダウンロードします。
   - Maven Central (78ページ)
   - Debian (79ページ)
   - RPM (80ページ)
2. お使いの Jetty の環境に合わせて、<YourContrastJarPath>を Contrast JAR (75ページ)ファイルのパスに置き換えます。 次に、以下の行を*<JettyDirectory>/start.ini* ファイルに追加します。

   ```
   -javaagent:<YourContrastJarPath>
   ```
3. Java 2 セキュリティマネージャを使用する場合は、以下のコードを含む *contrast.policy* ファイルを作成します(<YourContrastJarPath>は Contrast JAR (75ページ)ファイルへのパスに置き換えます)。

   ```
   grant codeBase "file:<YourContrastJarPath>" {
     permission java.security.AllPermission;
   };
   ```

   次に、以下を設定します。
   - **Jetty 7-8：** そのファイルを *JETTY_HOME/lib/policy* フォルダにコピーします。環境変数 JETTY_ARGS --secure を追加します。
   - **Jetty 9 :** 作成したポリシーを、自分で設定したポリシーに追加します。<YourPolicy>の箇所を自分のポリシー名に置き換えると、一般的な環境変数の設定でセキュリティマネージャが有効になります。

     ```
     -Djava.security.manager -Djava.security.policy=<YourPolicy>
     ```

重要
Jetty 9 以降では、セキュリティ管理ポリシーは正式にサポートされていません。

ヒント
Java2 セキュリティマネージャで Jetty を使用する方法についての詳細は、Jetty Policy(Jetty ポリシー)を参照してください。

Tomcat で Java エージェントを設定する

最初に、次のいずれかのリポジトリから、Contrast Java エージェント(JAR ファイル)をダウンロードします。

- Maven Central (78ページ)

- Debian (79ページ)
- RPM (80ページ)

Tomcat で Contrast を実行する方法に応じて、以下を参考にして Java エージェントを設定してください。

Windows または Unix で実行

CATALINA_OPTS 環境変数を使用して、Tomcat サーバを実行する JVM に設定フラグやシステムプロパティを渡します。

Tomcat では、setenv スクリプトを使用して環境変数を指定することを推奨しています。 スクリプトの作成や検索など setenv スクリプトの詳細については、Tomcat のディストリビューションに同梱されている RUNNING.txt を参照してください。

Contrast を有効にするには、Unix 系 OS であれば setenv.sh に、Windows であれば setenv.bat のいずれかで、-javaagent フラグを CATALINA_OPTS に追加します。例えば、以下のようになります。

- **Windows：**

```
set "CATALINA_OPTS=%CATALINA_OPTS% -javaagent:<YourContrastJarPath>"
```

- **Unix：**

```
export CATALINA_OPTS="$CATALINA_OPTS -javaagent:<YourContrastJarPath>"
```

Windows の Tomcat サービスで実行

1. Tomcat をサービスとして実行する場合は、Tomcat サービスマネージャを開き、JVM オプションを変更してエージェントを追加します。
2. システムトレイの Tomcat アイコンをダブルクリックします(または、右クリックして **Configure** を選択します)。(アイコンがない場合は、Tomcat の bin ディレクトリにある *tomcat9w.exe* を実行して手動で起動する必要があります。)
3. **Java** タブに切り替えて、-javaagent フラグを追加する箇所を確認します。

Java2 セキュリティを使用して Tomcat を実行

1. 以下のコードを含む *contrast.policy* ファイルを作成します(もしくは、*catalina.policy* ファイルにコードを追加します)。<YourContrastJarPath>は、Contrast JAR (75ページ)ファイルのパスに置き換えてください。例：

   ```
   grant codeBase "file:<YourContrastJarPath>" {
     permission java.security.AllPermission;
   };
   ```

2. *contrast.policy* ファイルを*$CATALINA_HOME/conf/catalina.policy* ファイルに追加(append)します。追加の設定は必要ありません。コマンドラインで、-security オプションを付けて Tomcat を起動します。

WebLogic で Java エージェントを設定する

最初に、次のいずれかのリポジトリから、Contrast Java エージェント(JAR ファイル)をダウンロードします。

- Maven Central (78ページ)
- Debian (79ページ)
- RPM (80ページ)

WebLogic で Contrast を実行する方法に応じて、以下を参考にして Java エージェントを設定してください。

Unix

1. WebLogic を自分で起動する場合は、インストール先の *bin* ディレクトリにある *startWebLogic* ファイルに Contrast の JVM パラメータを追加してください。UNIX ベースのオペレーティングシステムの場合、このファイルのパスは以下のようになります。

   ```
   /path/to/appserver/userprojects/domains/base_domain/bin/startWebLogic.sh
   ```
2. このファイルで、Java 実行ステップの前に、Contrast エンジンを`-javaagent` として `JAVA_OPTIONS` 環境変数に追加します。`<YourContrastJarPath>`は Contrast JAR (75ページ) ファイルへのパスに置き換えてください。例：

   ```
   export JAVA_OPTIONS="$JAVA_OPTIONS -javaagent:<YourContrastJarPath>"
   ```

Windows

1. Windows システムの場合、パスは以下のようになります。

   ```
   C:\Oracle\Middleware\userprojects\domains\base_domain\bin\startWebLogic.bat
   ```
2. このファイルの先頭で、Contrast エンジンを`-javaagent` として `JAVA_OPTIONS` 環境変数に追加します。`<YourContrastJarPath>`は Contrast JAR (75ページ)ファイルへのパスに置き換えます。お使いの環境に合わせた WebLogic サーバの情報に置き換えてください。例：

   ```
   set "JAVA_OPTIONS=%JAVA_OPTIONS% -javaagent:<YourContrastJarPath>"
   ```

WebLogic で Java2 を使用する

1. 以下のコードを含む *contrast.policy* ファイルを作成します(もしくは、*catalina.policy* ファイルにコードを追加します)。`<YourContrastJarPath>`は Contrast JAR (75ページ)ファイルのパスに置き換えてください。例：

   ```
   grant codeBase "file:<YourContrastJarPath>" {
     permission java.security.AllPermission;
   };
   ```
2. WebLogic には、*@WL_HOME/server/lib/weblogic.policy* というテンプレートファイルがあります。このファイルには、Java セキュリティマネージャを有効にして WebLogic サーバを起動するためのサンプル定義があります。旧バージョン(10 以前)の WebLogic の場合は、テンプレートファイルの *@WL_HOME* を WebLogic をインストールしたルートディレクトリへの実際のパスに置き換える必要があります。
3. セキュリティマネージャを有効にすると、ポリシーファイルである*@WL_HOME/server/lib/weblogic.policy* がデフォルトとして機能します。もしくは、`-Djava.security.policy==<YourPath>`でカスタムのポリシーファイルを指定することもできます。その場合、`<YourPath>`はカスタムファイルへのパスになります。==は、WebLogic が起動する際のデフォルトのパス設定を上書きするため、重要です。

ヒント
詳細は、Java セキュリティを使用して WebLogic のリソースを保護するを参照してください。

WebSphere で Java エージェントを設定する

最初に、次のいずれかのリポジトリから、Contrast Java エージェント(JAR ファイル)をダウンロードします。

- Maven Central (78ページ)
- Debian (79ページ)

- RPM (80ページ)

WebSphere で Contrast を実行する方法に応じて、以下を参考にして Java エージェントを設定してください。

注記
IBM J9 では、共有クラス(Shared Classes)機能を使用する場合、Java の計測 API(Instrumentation API)でコアの Java クラスを変更することができません。JVM パラメータに`-Xshareclasses:none` を指定して、この機能を無効にする必要があります。

同様に、`-Dcom.ibm.oti.shared.enabled=true` が設定されている場合、旧 J9 の JRE でも問題が発生する可能性があります。

Websphere のトラストストアとキーストア

WebSphere は、Java JRE の一部として組み込まれるトラストストアとは別に、独自のトラストストアとキーストアを保持します。エージェントは、WebSphere が初期化される前に起動するため、WebSphere 独自のトラストストアは設定されません。そのため、エージェントは、JVM に特別な設定がされていない限り、 Java の `JRE/lib/security/cacerts` ファイルにあるデフォルトのトラストストアを使用します。

ただし、内部専用や自己署名証明書を使用するプロキシサーバが必要な場合などには、特定の追加手順が必要になります。選択できる方法は、次のとおりです。

1. JRE `cacerts` のトラストストアと WebSphere のトラストストアの両方に必要な証明書をインストールします。これにより、証明書チェーンがエージェントと Web アプリケーションの両方で検証されます。
2. 標準のトラストストアのシステムプロパティを Java に指定し、トラストストアを WebSphere のトラストストアと同じものに変更します。その例を以下に記載します。この方法では、証明書を 1 つの場所(WebSphere のトラストストア)にインストールするだけでよいという利点があります。
   例：

```
-Djavax.net.ssl.trustStore=opt/IBM/WebSphere/AppServer/profiles/AppSrv01/
config/cells/DefaultCell01/nodes/DefaultNode01/trust.p12
-Djavax.net.ssl.trustStoreType=PKCS12
-Djavax.net.ssl.trustStorePassword=secret
```

WebSphere 自体は、パスワードをエンコードする方法をサポートしていますが、WebSphere が起動する前に実行されるため、エージェントのトラストストアのパスワードを設定するときには使用できません。

Websphere で Contrast を追加する

WebSphere を自分で起動する場合は、セルのディレクトリにある *server.xml* ファイルに Contrast の JVM パラメータを追加します。`<CellName>`と`<NodeName>`の箇所を、セルとノードの名前に置き換えます。`<YourContrastJarPath>`は Contrast JAR (75ページ)ファイルへのパスに置き換えてください。例：

```
<WebsphereDirectory>\AppServer\profiles\AppSrv01\config\cells\<CellName>\nod
es\<NodeName>\servers\server1\server.xml

<jvmEntries genericJvmArguments="-javaagent:<YourContrastJarPath> -
Xshareclasses:none">
```

```
   ...
</jvmEntries>
```

WebSphere 管理コンソールで Contrast を追加する

Websphere サポートサイトの手順に従って、WebSphere の管理コンソールから Contrast を追加することもできます。

WebSphere で Java2 を使用する

1. 以下のコードを含む *contrast.policy* ファイルを作成します(もしくは、*server.policy* ファイルにコードを追加します)。`<YourContrastJarPath>`は Contrast JAR (75ページ)ファイルへのパスに置き換えてください。例：

   ```
   grant codeBase "file:<YourContrastJarPath>" {
     permission java.security.AllPermission;
   };
   ```
2. `$WEBSPHERE_HOME/AppServer/profiles/AppSrv01/properties/server.policy` に `contrast.policy` ファイルを追加(append)します。
3. `wsadmin` ツールでセキュリティマネージャを有効にします。
   - **Jacl**：`$AdminTask setAdminActiveSecuritySettings {-enforceJava2Security true}`
   - **Jython**：`AdminTask.setAdminActiveSecuritySettings('-enforceJava2Security true')`

> **ヒント**
> 詳細については、Java セキュリティマネージャやスクリプトを使用した Java 2 セキュリティマネージャの有効/無効化についてのドキュメントを参照してください。

Java クイックスタートガイド

Contrast では、コードの脆弱性を監視するためのセンサーを配置するために、エージェントをインストールする必要があります。エージェントは、開発環境では脆弱性を解析し、実行時の本番環境では攻撃を検知します。

アプリケーションが実行されると、エージェントは情報(HTTP リクエスト、データフロー、バックエンド接続、ライブラリの依存関係など)を解析し、脆弱性や攻撃を Contrast サーバに送信します。Contrast でそれらの情報を参照でき、優先順位を付け、直ちに対策を講じることができます。

このガイドで、Contrast がわずか数分でアプリケーションで動作するようになり、Contrast の機能を確認することができます。

> **ヒント**
> 将来的なインストールに関しては、必要に応じて、組織のビルドツールやデプロイメントパイプライン、セキュリティ目標、Contrast を使用する環境なども考慮してください。ご利用状況に合わせて、Contrast をインストールする他の方法 (726ページ)を参照ください。

前提条件

このガイドで使用する Web アプリケーションは、以下の前提条件を満たすものをお選びください。

- Web アプリケーションがプロキシを経由せずにインターネットにアクセスできること。
- Web アプリケーションが JAR ファイルとしてパッケージ化されていること。
- Contrast でサポートされているバージョン、フレームワーク、ツール (75ページ)を使用していること。

また、コマンドラインインターフェース(エージェントをダウンロードするためのディレクトリを選択するため)と、ご利用になる組織の Contrast サーバのアクセスも必要です。 まだ Contrast をお持ちでない方は、Community Edition (30ページ)を無料でご利用いただけます。

インストール

1. Maven Central から、Contrast エージェント(JAR ファイル)をダウンロードします。 (78ページ)
2. エージェントを追加するための画面から YAML 設定ファイルをダウンロードします(まだダウンロードしていない場合)。これを行うには、Contrast Web インターフェイスで、**新規登録**を選択します。

   アプリケーションのカードを選択します。言語に、**Java** を選択したら、**Java エージェントの contrast_security.yaml をダウンロード**を選択します。 ファイルはローカルにダウンロードされ、アプリケーションを Contrast に接続するための組織固有のエージェントキーが含まれます。
3. YAML 設定ファイルをエディタで開き、エージェントの設定を指定します。

   ```
   api:
     url: https://xxx.contrastsecurity.com/Contrast
     api_key: A2xxxxxxxxxxxxxxxxxxxxxxxxxxxxG9N
     service_key: 88xxxxxxxxxxx5Z
     user_name: agent_xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx@OrgName
   ```
4. コマンドラインインターフェイスで以下のコマンドを入力して、YAML 設定ファイルがどこにあるかをエージェントに伝えます。

   ```
   java -javaagent:./contrast.jar -
   Dcontrast.config.path=contrast_security.yaml -jar <ApplicationJarPath>
   ```

   <ApplicationJarPath>は必ずお使いのアプリケーションへのパスに置き換えてください。
   例：./MyApplication.jar
5. Contrast が機能しているかを確認するために、通常通りにアプリケーションを使用します。 例えば、アプリケーションの Web インターフェイスをクリックしたり、API コマンドを送信してみてください。
   そして、Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。アプリケーションの名前が表示されているはずです。
   また、Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、一覧にサーバ(ローカル)のホスト名が表示されているはずです。

Java エージェントの設定

すべての Contrast エージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。

Java エージェントは、以下を使用して設定できます。

- Java システムプロパティ
- 環境変数 (66ページ)
- Java YAML テンプレート

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

システムで以下のような構成を使用している場合は、Contrast エージェントと効率的に連携させるために、アプリケーションの Java 環境に設定が必要な場合もあります。

- マルチテナントアプリケーション構成：デプロイ中に JVM アプリケーションサーバが複数のアプリケーションをホストしている場合は、アプリケーションをそれぞれ区別して、個々の設定オプションを適用することできます。
  マルチテナントアプリケーション構成：デプロイ中に JVM アプリケーションサーバが複数のアプリケーションをホストしている場合は、アプリケーションをそれぞれ区別して、個々の設定オプションを適用することができます。
- TLS 証明書 (163ページ)
- Java 9 モジュール (164ページ)
- Java 2 セキュリティ (164ページ)
- インテグレーション (726ページ) : Contrast Java エージェントは、プラグインやサードパーティ製ツールを使用したり、外部のシステムと連携することができます。他の製品の動作については、その製品のマニュアルを参照してください。

Java システムプロパティ

<YourContrastJarPath>を Contrast JAR ファイル (75ページ)へのパスに置き換えて以下のコマンドを実行すると、システムプロパティの詳細情報を確認できます。

- 以下のコマンドにより、Contrast エージェントの JAR ファイルを使用して、一般的なプロパティの一覧を表示することができます。

```
java -jar <YourContrastJarPath> properties
```

- コマンドを検索するには、ツール名を指定してコマンドラインを使用します。例えば、以下のコマンドでプロキシ関連のプロパティの一覧が表示されます。
  filter オプションを使用する場合：

```
java -jar <YourContrastJarPath> properties --filter=proxy
```

Java の YAML 設定ファイルのテンプレート

YAML 設定ファイルを使用して Java エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、こちらの説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します。

- **Unix**： `/etc/contrast/java/contrast_security.yaml`
- **Windows**： `C:/ProgramData/contrast/java/contrast_security.yaml`

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==
```

```
# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the timeout for communicating with TeamServer. This property will be
  # respected over the deprecated legacy configuration *contrast.timeout*.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
```

```
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set this property as an alternate for `scheme://host:port`. It takes
    # precedence over the other settings, if specified; however, an error
    # will be thrown if both the URL and individual properties are set.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # \
================================================================================
    # agent.diagnostics.logger
    # The agent diagnostics logger that will
    # stream agent logs to a remote collector
    #
    # \
```

```
# ============================================================================
    # logger:

      # Enables the agent diagnostics logger that
      # will stream agent logs to a remote collector.
      #
      # enable: false

      # The expiration time for diagnostics (in milliseconds since the
      # Unix Epoch, 1970-01-01). Defaults to 1 hour from when diagnostics
      # start. Maximum is 24 hours from when diagnostics start.
      #
      # expires_ms: NEEDS_TO_BE_SET

      # The log level of agent log messages to send to the diagnostics
      # collector. Levels with lower severity will not be sent.
      #
      # level: DEBUG

      # The unique identifier for the current diagnostics logger
      # collection. Defaults to a new UUID if none is provided.
      #
      # uuid: NEEDS_TO_BE_SET

  # \
# ============================================================================
  # agent.reporting
  # Use the following settings to configure reporting to the Contrast UI.
  # \
# ============================================================================
  # reporting:

    # Set the grace period (in milliseconds) after
    # agent shutdown to allow draining pending reports.
    # shutdown_grace_period_ms: 120000

  # \
# ============================================================================
  # agent.effective_config
  # None
  # \
# ============================================================================
  # effective_config:

    # \
# ============================================================================
    # agent.effective_config.reporting
    # None
    # \
# ============================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true
```

```
  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead of
    # the file system. May be combined with the corresponding
    # `stdout` configuration to write to both streams.
    # stderr: false

    # Change the Contrast logger from a file-sized based rolling scheme
    # to a date-based rolling scheme. At midnight server time, the
    # previous day log is renamed to *file_name.yyyy-MM-dd*. Note -
    # this scheme does not have a size limit; manual log pruning is
    # required. You must set this flag to use the backups and size flags.
    # roll_daily: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
```

```
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET
```

```
      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

      # \
=============================================================================
      # agent.security_logger.syslog.heartbeat
      # Define the following properties to
      # set the Syslog heartbeat properties.
      # \
=============================================================================
      # heartbeat:

        # Set to `true` to enable the Syslog heartbeat.
        # The heartbeat will issue a Syslog message at
        # the INFO level after every interval passes.
        # enable: false

        # Set the interval for sending heartbeat messages
        # to the Syslog server (in milliseconds).
        # interval_ms: 60000

  # \
=============================================================================
  # agent.java
  # The following properties apply to any Java agent-wide configurations.
  # \
=============================================================================
  # java:

    # Configure the Java agent to skip its application discovery
    # algorithm, and instead associate all libraries, vulnerabilities,
    # and web traffic to a single application with the name specified
    # by this property. This configuration is preferred when deploying
    # Java SE applications with embedded web servers (e.g., applications
```

```
    # built with Spring Boot, Dropwizard, and embedded Jetty). When used
    # with an application server, this configuration associates all
    # web traffic with the single, standalone application, including
    # web traffic handled by application server-hosted endpoints that
    # would not be associated with a discovered application otherwise.
    #
    # Note - This settings takes preferences
    # over the `application.name` setting.
    #
    # standalone_app_name: NEEDS_TO_BE_SET

    # By default, the Java agent visits all classes at startup to look
    # for vulnerabilities, which the agent may detect by scanning a
    # class (e.g., hardcoded passwords). Set this property to `false`
    # to disable the default behavior. If disabled, the agent will
    # only visit classes which are likely to require sensors; this
    # can improve application startup time, but may produce fewer
    # findings (most likely findings that require static analysis).
    #
    # scan_all_classes: true

    # By default, the Java agent deeply inspects all JAR and WAR files \
loaded
    # by the JVM to build a comprehensive understanding of the type \
hierarchy.
    # This understanding allows Contrast to instrument sensors into types
    # that it might have overlooked. In most cases, this produces a slight
    # increase in accuracy at the cost of increased application startup
    # time. Set this property to `false` to disable this level of \
inspection.
    #
    # scan_all_code_sources: true

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Define a list of directories where libraries are stored.
  # Directories must be formatted as a semicolon-delimited list.
  # Example - `path1;path2;path3`
  #
  # library_dirs: NEEDS_TO_BE_SET

  # Set the maximum archive unpacking depth when analyzing libraries.
  # library_depth: 10
```

```
  # Set the boolean to more aggressively limit the
  # manifest information reported for libraries. If true,
  # the limit is 1,000 characters, otherwise it's 3,000.
  # prune_package_details: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
```

```
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# profile
# Set configuration values under a profile name to enable
# multi-tenant application configuration on web servers. See
# https://support.contrastsecurity.com/hc/en-us/articles/360052187171-Multi-
Application-configuration-with-Contrast-Profiles
# for more details.
# \
================================================================================
==
# profile: {}

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
```

```
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when semantic analysis of the query
      # reveals tautologies used in exfiltration attacks (e.g., "or
      # 1=1" or "or 2<>3"). The agent blocks if blocking is enabled.
      # detect_tautologies: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals the invocation of dangerous functions typically used in
      # weaponized exploits. The agent blocks if blocking is enabled.
      # detect_dangerous_functions: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals chained queries, which is uncommon in normal usage but
      # common in exploit. The agent blocks if blocking is enabled.
      # detect_chained_queries: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals database queries are being made for system tables and
```

```
      # sensitive information. The agent blocks if blocking is enabled.
      # detect_suspicious_unions: false

      # Tell the agent to be more aggressive in detecting user
      # inputs as SQL comments. This enables the agent to better
      # detect SQL Injection input vectors that use comments to
      # terminate queries. The agent blocks if blocking is enabled.
      # aggressive_comment: false

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when the agent sees user parameters being executed as
      # system commands. The agent blocks if blocking is enabled.
      # detect_parameter_command_backdoors: true

      # Detect when a system command is issued which contains
      # chained commands. The agent blocks if blocking is enabled.
      # detect_chained_commands: true

      # Detect when a system command is issued with an argument matching a
      # known dangerous file path. The agent blocks if blocking is enabled.
      # detect_dangerous_path_args: true

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-process-hardening
    # Use the following settings to configure whether
    # the agent blocks all attempts to start an external
    # process. To enable blocking, set to 'true'.
    # \
================================================================================
    # cmd-injection-process-hardening:

      # Set to `true` to enable the agent to block
      # all attempts to start external processes.
      # enable: false
```

```
    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
```

```
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.padding-oracle
    # Use the following properties to configure
    # how the padding-oracle rule works.
    # \
================================================================================
    # padding-oracle: {}

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET
```

```
  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost
```

```
  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
==========================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
==========================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
==========================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
==========================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
```

```
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the timeout for communicating with TeamServer. This property will be
  # respected over the deprecated legacy configuration *contrast.timeout*.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set this property as an alternate for `scheme://host:port`. It takes
    # precedence over the other settings, if specified; however, an error
    # will be thrown if both the URL and individual properties are set.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
```

```
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # \
================================================================================
    # agent.diagnostics.logger
    # The agent diagnostics logger that will
    # stream agent logs to a remote collector
    #
    # \
================================================================================
    # logger:

      # Enables the agent diagnostics logger that
      # will stream agent logs to a remote collector.
      #
      # enable: false

      # The expiration time for diagnostics (in milliseconds since the
      # Unix Epoch, 1970-01-01). Defaults to 1 hour from when diagnostics
      # start. Maximum is 24 hours from when diagnostics start.
      #
      # expires_ms: NEEDS_TO_BE_SET

      # The log level of agent log messages to send to the diagnostics
      # collector. Levels with lower severity will not be sent.
      #
      # level: DEBUG

      # The unique identifier for the current diagnostics logger
      # collection. Defaults to a new UUID if none is provided.
```

```
      #
      # uuid: NEEDS_TO_BE_SET

  # \
================================================================================
  # agent.reporting
  # Use the following settings to configure reporting to the Contrast UI.
  # \
================================================================================
  # reporting:

    # Set the grace period (in milliseconds) after
    # agent shutdown to allow draining pending reports.
    # shutdown_grace_period_ms: 120000

  # \
================================================================================
  # agent.effective_config
  # None
  # \
================================================================================
  # effective_config:

    # \
================================================================================
    # agent.effective_config.reporting
    # None
    # \
================================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log
```

```
    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead of
    # the file system. May be combined with the corresponding
    # `stdout` configuration to write to both streams.
    # stderr: false

    # Change the Contrast logger from a file-sized based rolling scheme
    # to a date-based rolling scheme. At midnight server time, the
    # previous day log is renamed to *file_name.yyyy-MM-dd*. Note -
    # this scheme does not have a size limit; manual log pruning is
    # required. You must set this flag to use the backups and size flags.
    # roll_daily: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET
```

```
    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
```

```
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

      # \
=============================================================================
      # agent.security_logger.syslog.heartbeat
      # Define the following properties to
      # set the Syslog heartbeat properties.
      # \
=============================================================================
      # heartbeat:

        # Set to `true` to enable the Syslog heartbeat.
        # The heartbeat will issue a Syslog message at
        # the INFO level after every interval passes.
        # enable: false

        # Set the interval for sending heartbeat messages
        # to the Syslog server (in milliseconds).
        # interval_ms: 60000

  # \
=============================================================================
  # agent.java
  # The following properties apply to any Java agent-wide configurations.
  # \
=============================================================================
  # java:

    # Configure the Java agent to skip its application discovery
    # algorithm, and instead associate all libraries, vulnerabilities,
    # and web traffic to a single application with the name specified
    # by this property. This configuration is preferred when deploying
    # Java SE applications with embedded web servers (e.g., applications
    # built with Spring Boot, Dropwizard, and embedded Jetty). When used
    # with an application server, this configuration associates all
    # web traffic with the single, standalone application, including
    # web traffic handled by application server-hosted endpoints that
    # would not be associated with a discovered application otherwise.
    #
    # Note - This settings takes preferences
    # over the `application.name` setting.
    #
    # standalone_app_name: NEEDS_TO_BE_SET

    # By default, the Java agent visits all classes at startup to look
    # for vulnerabilities, which the agent may detect by scanning a
    # class (e.g., hardcoded passwords). Set this property to `false`
    # to disable the default behavior. If disabled, the agent will
    # only visit classes which are likely to require sensors; this
    # can improve application startup time, but may produce fewer
    # findings (most likely findings that require static analysis).
    #
    # scan_all_classes: true
```

```
    # By default, the Java agent deeply inspects all JAR and WAR files \
loaded
    # by the JVM to build a comprehensive understanding of the type \
hierarchy.
    # This understanding allows Contrast to instrument sensors into types
    # that it might have overlooked. In most cases, this produces a slight
    # increase in accuracy at the cost of increased application startup
    # time. Set this property to `false` to disable this level of \
inspection.
    #
    # scan_all_code_sources: true

# \
==============================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
==============================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Define a list of directories where libraries are stored.
  # Directories must be formatted as a semicolon-delimited list.
  # Example - `path1;path2;path3`
  #
  # library_dirs: NEEDS_TO_BE_SET

  # Set the maximum archive unpacking depth when analyzing libraries.
  # library_depth: 10

  # Set the boolean to more aggressively limit the
  # manifest information reported for libraries. If true,
  # the limit is 1,000 characters, otherwise it's 3,000.
  # prune_package_details: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
==============================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
==============================================================================
==
# assess:
```

```
  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
```

```
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# profile
# Set configuration values under a profile name to enable
# multi-tenant application configuration on web servers. See
# https://support.contrastsecurity.com/hc/en-us/articles/360052187171-Multi-
Application-configuration-with-Contrast-Profiles
# for more details.
# \
================================================================================
==
# profile: {}

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false
```

```
    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when semantic analysis of the query
      # reveals tautologies used in exfiltration attacks (e.g., "or
      # 1=1" or "or 2<>3"). The agent blocks if blocking is enabled.
      # detect_tautologies: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals the invocation of dangerous functions typically used in
      # weaponized exploits. The agent blocks if blocking is enabled.
      # detect_dangerous_functions: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals chained queries, which is uncommon in normal usage but
      # common in exploit. The agent blocks if blocking is enabled.
      # detect_chained_queries: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals database queries are being made for system tables and
      # sensitive information. The agent blocks if blocking is enabled.
      # detect_suspicious_unions: false

      # Tell the agent to be more aggressive in detecting user
      # inputs as SQL comments. This enables the agent to better
      # detect SQL Injection input vectors that use comments to
      # terminate queries. The agent blocks if blocking is enabled.
      # aggressive_comment: false

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
```

```
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when the agent sees user parameters being executed as
      # system commands. The agent blocks if blocking is enabled.
      # detect_parameter_command_backdoors: true

      # Detect when a system command is issued which contains
      # chained commands. The agent blocks if blocking is enabled.
      # detect_chained_commands: true

      # Detect when a system command is issued with an argument matching a
      # known dangerous file path. The agent blocks if blocking is enabled.
      # detect_dangerous_path_args: true

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-process-hardening
    # Use the following settings to configure whether
    # the agent blocks all attempts to start an external
    # process. To enable blocking, set to 'true'.
    # \
================================================================================
    # cmd-injection-process-hardening:

      # Set to `true` to enable the agent to block
      # all attempts to start external processes.
      # enable: false

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true
```

```
      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
```

```
      # mode: off

    # \
================================================================================
    # protect.rules.padding-oracle
    # Use the following properties to configure
    # how the padding-oracle rule works.
    # \
================================================================================
    # padding-oracle: {}

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
```

```
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
```

```
  #
  # tags: NEEDS_TO_BE_SET
```

Java システムプロパティ

<YourContrastJarPath>を Contrast JAR ファイル (75ページ)へのパスに置き換えて以下のコマンドを実行すると、システムプロパティの詳細情報を確認できます。

- 以下のコマンドにより、Contrast エージェントの JAR ファイルを使用して、一般的なプロパティの一覧を表示することができます。

  ```
  java -jar <YourContrastJarPath> properties
  ```
- コマンドを検索するには、ツール名を指定してコマンドラインを使用します。例えば、以下のコマンドでプロキシ関連のプロパティの一覧が表示されます。
  filter オプションを使用する場合：

  ```
  java -jar <YourContrastJarPath> properties --filter=proxy
  ```

Java の YAML 設定ファイルのテンプレート

YAML 設定ファイルを使用して Java エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、こちらの説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します。

- **Unix**： `/etc/contrast/java/contrast_security.yaml`
- **Windows**： `C:/ProgramData/contrast/java/contrast_security.yaml`

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast
```

```
  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the timeout for communicating with TeamServer. This property will be
  # respected over the deprecated legacy configuration *contrast.timeout*.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set this property as an alternate for `scheme://host:port`. It takes
    # precedence over the other settings, if specified; however, an error
    # will be thrown if both the URL and individual properties are set.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET
```

```
    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # \
================================================================================
    # agent.diagnostics.logger
    # The agent diagnostics logger that will
    # stream agent logs to a remote collector
    #
    # \
================================================================================
    # logger:

      # Enables the agent diagnostics logger that
      # will stream agent logs to a remote collector.
      #
      # enable: false

      # The expiration time for diagnostics (in milliseconds since the
      # Unix Epoch, 1970-01-01). Defaults to 1 hour from when diagnostics
      # start. Maximum is 24 hours from when diagnostics start.
      #
      # expires_ms: NEEDS_TO_BE_SET

      # The log level of agent log messages to send to the diagnostics
      # collector. Levels with lower severity will not be sent.
      #
      # level: DEBUG
```

```
      # The unique identifier for the current diagnostics logger
      # collection. Defaults to a new UUID if none is provided.
      #
      # uuid: NEEDS_TO_BE_SET

  # \
================================================================================
  # agent.reporting
  # Use the following settings to configure reporting to the Contrast UI.
  # \
================================================================================
  # reporting:

    # Set the grace period (in milliseconds) after
    # agent shutdown to allow draining pending reports.
    # shutdown_grace_period_ms: 120000

  # \
================================================================================
  # agent.effective_config
  # None
  # \
================================================================================
  # effective_config:

    # \
================================================================================
    # agent.effective_config.reporting
    # None
    # \
================================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
```

```
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead of
    # the file system. May be combined with the corresponding
    # `stdout` configuration to write to both streams.
    # stderr: false

    # Change the Contrast logger from a file-sized based rolling scheme
    # to a date-based rolling scheme. At midnight server time, the
    # previous day log is renamed to *file_name.yyyy-MM-dd*. Note -
    # this scheme does not have a size limit; manual log pruning is
    # required. You must set this flag to use the backups and size flags.
    # roll_daily: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET
```

```
    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING
```

```
      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

      # \
================================================================================
      # agent.security_logger.syslog.heartbeat
      # Define the following properties to
      # set the Syslog heartbeat properties.
      # \
================================================================================
      # heartbeat:

        # Set to `true` to enable the Syslog heartbeat.
        # The heartbeat will issue a Syslog message at
        # the INFO level after every interval passes.
        # enable: false

        # Set the interval for sending heartbeat messages
        # to the Syslog server (in milliseconds).
        # interval_ms: 60000

  # \
================================================================================
  # agent.java
  # The following properties apply to any Java agent-wide configurations.
  # \
================================================================================
  # java:

    # Configure the Java agent to skip its application discovery
    # algorithm, and instead associate all libraries, vulnerabilities,
    # and web traffic to a single application with the name specified
    # by this property. This configuration is preferred when deploying
    # Java SE applications with embedded web servers (e.g., applications
    # built with Spring Boot, Dropwizard, and embedded Jetty). When used
    # with an application server, this configuration associates all
    # web traffic with the single, standalone application, including
    # web traffic handled by application server-hosted endpoints that
    # would not be associated with a discovered application otherwise.
    #
    # Note - This settings takes preferences
    # over the `application.name` setting.
    #
    # standalone_app_name: NEEDS_TO_BE_SET

    # By default, the Java agent visits all classes at startup to look
    # for vulnerabilities, which the agent may detect by scanning a
    # class (e.g., hardcoded passwords). Set this property to `false`
    # to disable the default behavior. If disabled, the agent will
    # only visit classes which are likely to require sensors; this
    # can improve application startup time, but may produce fewer
    # findings (most likely findings that require static analysis).
    #
```

```
    # scan_all_classes: true

    # By default, the Java agent deeply inspects all JAR and WAR files \
loaded
    # by the JVM to build a comprehensive understanding of the type \
hierarchy.
    # This understanding allows Contrast to instrument sensors into types
    # that it might have overlooked. In most cases, this produces a slight
    # increase in accuracy at the cost of increased application startup
    # time. Set this property to `false` to disable this level of \
inspection.
    #
    # scan_all_code_sources: true

# \
==========================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
==========================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Define a list of directories where libraries are stored.
  # Directories must be formatted as a semicolon-delimited list.
  # Example - `path1;path2;path3`
  #
  # library_dirs: NEEDS_TO_BE_SET

  # Set the maximum archive unpacking depth when analyzing libraries.
  # library_depth: 10

  # Set the boolean to more aggressively limit the
  # manifest information reported for libraries. If true,
  # the limit is 1,000 characters, otherwise it's 3,000.
  # prune_package_details: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
==========================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
==========================================================================
==
```

```
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
```

```
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# profile
# Set configuration values under a profile name to enable
# multi-tenant application configuration on web servers. See
# https://support.contrastsecurity.com/hc/en-us/articles/360052187171-Multi-
Application-configuration-with-Contrast-Profiles
# for more details.
# \
================================================================================
==
# profile: {}

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:
```

```
      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when semantic analysis of the query
      # reveals tautologies used in exfiltration attacks (e.g., "or
      # 1=1" or "or 2<>3"). The agent blocks if blocking is enabled.
      # detect_tautologies: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals the invocation of dangerous functions typically used in
      # weaponized exploits. The agent blocks if blocking is enabled.
      # detect_dangerous_functions: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals chained queries, which is uncommon in normal usage but
      # common in exploit. The agent blocks if blocking is enabled.
      # detect_chained_queries: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals database queries are being made for system tables and
      # sensitive information. The agent blocks if blocking is enabled.
      # detect_suspicious_unions: false

      # Tell the agent to be more aggressive in detecting user
      # inputs as SQL comments. This enables the agent to better
      # detect SQL Injection input vectors that use comments to
      # terminate queries. The agent blocks if blocking is enabled.
      # aggressive_comment: false

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
```

```
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when the agent sees user parameters being executed as
      # system commands. The agent blocks if blocking is enabled.
      # detect_parameter_command_backdoors: true

      # Detect when a system command is issued which contains
      # chained commands. The agent blocks if blocking is enabled.
      # detect_chained_commands: true

      # Detect when a system command is issued with an argument matching a
      # known dangerous file path. The agent blocks if blocking is enabled.
      # detect_dangerous_path_args: true

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-process-hardening
    # Use the following settings to configure whether
    # the agent blocks all attempts to start an external
    # process. To enable blocking, set to 'true'.
    # \
================================================================================
    # cmd-injection-process-hardening:

      # Set to `true` to enable the agent to block
      # all attempts to start external processes.
      # enable: false

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
```

```
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
```

```
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.padding-oracle
    # Use the following properties to configure
    # how the padding-oracle rule works.
    # \
================================================================================
    # padding-oracle: {}

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
```

```
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
```

```
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the timeout for communicating with TeamServer. This property will be
  # respected over the deprecated legacy configuration *contrast.timeout*.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
================================================================================
```

```
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set this property as an alternate for `scheme://host:port`. It takes
    # precedence over the other settings, if specified; however, an error
    # will be thrown if both the URL and individual properties are set.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
```

```
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # \
================================================================================
    # agent.diagnostics.logger
    # The agent diagnostics logger that will
    # stream agent logs to a remote collector
    #
    # \
================================================================================
    # logger:

      # Enables the agent diagnostics logger that
      # will stream agent logs to a remote collector.
      #
      # enable: false

      # The expiration time for diagnostics (in milliseconds since the
      # Unix Epoch, 1970-01-01). Defaults to 1 hour from when diagnostics
      # start. Maximum is 24 hours from when diagnostics start.
      #
      # expires_ms: NEEDS_TO_BE_SET

      # The log level of agent log messages to send to the diagnostics
      # collector. Levels with lower severity will not be sent.
      #
      # level: DEBUG

      # The unique identifier for the current diagnostics logger
      # collection. Defaults to a new UUID if none is provided.
      #
      # uuid: NEEDS_TO_BE_SET

  # \
================================================================================
  # agent.reporting
  # Use the following settings to configure reporting to the Contrast UI.
  # \
================================================================================
  # reporting:

    # Set the grace period (in milliseconds) after
    # agent shutdown to allow draining pending reports.
    # shutdown_grace_period_ms: 120000

  # \
================================================================================
  # agent.effective_config
  # None
```

```
  # \
================================================================================
  # effective_config:

    # \
==============================================================================
    # agent.effective_config.reporting
    # None
    # \
==============================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead of
    # the file system. May be combined with the corresponding
    # `stdout` configuration to write to both streams.
    # stderr: false

    # Change the Contrast logger from a file-sized based rolling scheme
    # to a date-based rolling scheme. At midnight server time, the
    # previous day log is renamed to *file_name.yyyy-MM-dd*. Note -
    # this scheme does not have a size limit; manual log pruning is
    # required. You must set this flag to use the backups and size flags.
```

```
    # roll_daily: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
```

```
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

      # \
================================================================================
      # agent.security_logger.syslog.heartbeat
      # Define the following properties to
      # set the Syslog heartbeat properties.
      # \
================================================================================
      # heartbeat:

        # Set to `true` to enable the Syslog heartbeat.
        # The heartbeat will issue a Syslog message at
        # the INFO level after every interval passes.
        # enable: false

        # Set the interval for sending heartbeat messages
        # to the Syslog server (in milliseconds).
```

```
        # interval_ms: 60000

  # \
================================================================================
  # agent.java
  # The following properties apply to any Java agent-wide configurations.
  # \
================================================================================
  # java:

    # Configure the Java agent to skip its application discovery
    # algorithm, and instead associate all libraries, vulnerabilities,
    # and web traffic to a single application with the name specified
    # by this property. This configuration is preferred when deploying
    # Java SE applications with embedded web servers (e.g., applications
    # built with Spring Boot, Dropwizard, and embedded Jetty). When used
    # with an application server, this configuration associates all
    # web traffic with the single, standalone application, including
    # web traffic handled by application server-hosted endpoints that
    # would not be associated with a discovered application otherwise.
    #
    # Note - This settings takes preferences
    # over the `application.name` setting.
    #
    # standalone_app_name: NEEDS_TO_BE_SET

    # By default, the Java agent visits all classes at startup to look
    # for vulnerabilities, which the agent may detect by scanning a
    # class (e.g., hardcoded passwords). Set this property to `false`
    # to disable the default behavior. If disabled, the agent will
    # only visit classes which are likely to require sensors; this
    # can improve application startup time, but may produce fewer
    # findings (most likely findings that require static analysis).
    #
    # scan_all_classes: true

    # By default, the Java agent deeply inspects all JAR and WAR files \
loaded
    # by the JVM to build a comprehensive understanding of the type \
hierarchy.
    # This understanding allows Contrast to instrument sensors into types
    # that it might have overlooked. In most cases, this produces a slight
    # increase in accuracy at the cost of increased application startup
    # time. Set this property to `false` to disable this level of \
inspection.
    #
    # scan_all_code_sources: true

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
```

```
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Define a list of directories where libraries are stored.
  # Directories must be formatted as a semicolon-delimited list.
  # Example - `path1;path2;path3`
  #
  # library_dirs: NEEDS_TO_BE_SET

  # Set the maximum archive unpacking depth when analyzing libraries.
  # library_depth: 10

  # Set the boolean to more aggressively limit the
  # manifest information reported for libraries. If true,
  # the limit is 1,000 characters, otherwise it's 3,000.
  # prune_package_details: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
```

```
# ============================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
# ============================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
# ============================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
# ============================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
# ============================================================================
==
# profile
# Set configuration values under a profile name to enable
# multi-tenant application configuration on web servers. See
# https://support.contrastsecurity.com/hc/en-us/articles/360052187171-Multi-
Application-configuration-with-Contrast-Profiles
# for more details.
# \
# ============================================================================
==
# profile: {}

# \
# ============================================================================
==
```

```
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when semantic analysis of the query
      # reveals tautologies used in exfiltration attacks (e.g., "or
```

```
      # 1=1" or "or 2<>3"). The agent blocks if blocking is enabled.
      # detect_tautologies: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals the invocation of dangerous functions typically used in
      # weaponized exploits. The agent blocks if blocking is enabled.
      # detect_dangerous_functions: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals chained queries, which is uncommon in normal usage but
      # common in exploit. The agent blocks if blocking is enabled.
      # detect_chained_queries: false

      # Tell the agent to detect when semantic analysis of the query
      # reveals database queries are being made for system tables and
      # sensitive information. The agent blocks if blocking is enabled.
      # detect_suspicious_unions: false

      # Tell the agent to be more aggressive in detecting user
      # inputs as SQL comments. This enables the agent to better
      # detect SQL Injection input vectors that use comments to
      # terminate queries. The agent blocks if blocking is enabled.
      # aggressive_comment: false

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when the agent sees user parameters being executed as
      # system commands. The agent blocks if blocking is enabled.
      # detect_parameter_command_backdoors: true

      # Detect when a system command is issued which contains
      # chained commands. The agent blocks if blocking is enabled.
      # detect_chained_commands: true

      # Detect when a system command is issued with an argument matching a
      # known dangerous file path. The agent blocks if blocking is enabled.
      # detect_dangerous_path_args: true

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
```

```
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-process-hardening
    # Use the following settings to configure whether
    # the agent blocks all attempts to start an external
    # process. To enable blocking, set to 'true'.
    # \
================================================================================
    # cmd-injection-process-hardening:

      # Set to `true` to enable the agent to block
      # all attempts to start external processes.
      # enable: false

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
```

```
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.padding-oracle
    # Use the following properties to configure
    # how the padding-oracle rule works.
    # \
================================================================================
    # padding-oracle: {}

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
```

```
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
```

```
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

スタンドアロンアプリケーションの Java エージェントを設定

注記

Java エージェント 4.x では、スタンドアロンアプリケーションの設定は必要ありません。

バージョン 3.x を使用している場合は、旧 Java エージェント (165ページ)のドキュメントを確認してください。

TLS(Transport Layer Security)

Contrast Java エージェントは、Contrast サーバと通信するために安全な TLS 接続を使用しています。

Contrast の SaaS 版では、よりセキュリティが強化された TLS 1.2 の接続と業界標準の認証局(CA)によって署名された証明書を使用しております。また、オンプレミス版をご利用のお客様は、エンタープライズ CA を使用するために Java エージェントを設定し、Java エージェントが TLS ハンドシェイクでクライアント証明書を送信するよう設定する必要があります。

Contrast Java エージェントは、TLS を構成するために標準の Java 暗号化アーキテクチャ(JCA)を使用します。具体的に、Java エージェントは、システムの「TLS」の javax.net.ssl.SSLContext を使用します。そのため、ほとんどのユーザは、標準の `javax.net.ssl.trustStore` システムプロパティを使用して、エージェントが信頼できる証明書を指定できます。また、標準の `javax.net.ssl.keyStore` システムプロパティを使用して、TLS サーバがクライアント証明書を要求する際にエージェントから送信される証明書を指定することもできます。

以下は、カスタムキーストアとトラストストアを使用して、Java エージェントを設定している例です。

```
java \
  -javaagent:contrast.jar \
  -Djavax.net.ssl.trustStore=/etc/pki/tls/my-enterprise-truststore.p12 \
  -Djavax.net.ssl.trustStorePassword=changeit \
  -Djavax.net.ssl.trustStoreType=PKCS12 \
  -Djavax.net.ssl.keyStore=/etc/pki/tls/server-client-certificate.p12 \
  -Djavax.net.ssl.keyStorePassword=password \
  -Djavax.net.ssl.keyStoreType=PKCS12 \
  -jar my-server.jar
```

JPMS(Java Platform Module System)で Java エージェントを使用

JPMS は Java 9 で導入されており、コードをカプセル化する機能を提供します。Contrast は、JPMS で書かれたモジュールの検査とアプリケーションの起動をサポートします。

Java エージェントを使用するには、アプリケーションの `module-info.java` ファイルに `java.sql` パッケージの利用を宣言する必要があります。

```
module mymodule {
    requires java.sql;
}
```

または、ランタイムに`--add-modules` というコマンドラインの引数で指定します。

```
java -javaagent:/opt/contrast/contrast-agent.jar --add-modules java.sql --
module-path libs --module mymodule/mycompany.App
```

Java 2 セキュリティ

Java 2 セキュリティマネージャを使用すると、システム管理者は、JVM 内の Java コードに対して使用可能な権限を規定するポリシーを適用できます。

Java エージェントで Java 2 セキュリティマネージャを使用する場合は、Java セキュリティポリシーファイルを設定して、Java コードのプリンシパルに権限を適用する必要があります。

Java コードのプリンシパルは、通常 CodeSource (JAR など)によって識別され、まれに署名付き JAR のエンティティによる場合もあります。

例えば、Tomcat のデフォルトの catalina.policy ファイルで、このポリシーは JDBC ドライバの JAR に権限を付与しています。

```
// The permission granted to your JDBC driver
grant codeBase "jar:file:${catalina.base}/webapps/examples/WEB-INF/lib/
driver.jar!/-" {
     permission \
```

```
java.net.SocketPermission "dbhost.mycompany.com:5432", "connect";
};
```

Java 2 セキュリティマネージャは、ユーザがデプロイするコードをシステム管理者が完全に信頼できない場合には便利です。例えば、マルチテナントの Tomcat インスタンスでユーザのアプリケーションをホストする場合、Java 2 セキュリティマネージャを使用して、ユーザのアプリケーションによってサービス全体が停止されないように制御できます(例、`System.exit()`の呼出しを許可しないなど)。

Contrast の Java エージェントで Java 2 セキュリティマネージャを使用する場合は、セキュリティポリシーファイルで、Java エージェントに全権限を付与する必要があります(*java.security.AllPermission*)。これを行うには、`<YourContrastJarPath>`を Contrast JAR (75ページ)ファイルへのパスに置き換えて、以下を使用してください。

```
grant codeBase "file:<YourContrastJarPath>" {
  permission java.security.AllPermission;
};
```

Java 2 セキュリティマネージャと、以下のいずれかの環境を使用する場合は、さらに設定が必要になる場合があります。

- Glassfish
- Jetty (98ページ)
- Tomcat (98ページ)
- WebLogic (99ページ)
- WebSphere (100ページ)
- WildFly (96ページ)

旧 Java エージェント

Java エージェントのバージョン 4.x および EOP のバージョン 3.9.5 のリリースに伴い、次のようなレガシーテクノロジで動作する Java 6· Java 7 の Web アプリケーションを検査するには、Contrast の旧 Java エージェント(3.x 以前)を使用してください。

- Jboss 4.2、6.1、7.1
- Weblogic 9、10、11

旧エージェントには、CVE-45046 に対する保護を提供する JNDI ルールが Contrast Protect に含まれています。

手順

エージェントのバージョンが 3.15 以前の場合、ルートカバレッジなどの特定の機能を利用するには、追加のプロパティの設定が必要です。

1. YAML 設定ファイルをエディタで開きます。
2. あらかじめ入力されている認証キーに加えて、`agent.java.standalone_app_name` プロパティを追加します。
   このプロパティには、Contrast で表示させたいアプリケーションの名前を割り当てます。この例では、`<MyAppName>`を使用したい名前に置き換えてください。

   ```
   api:
     url: https://xxx.contrastsecurity.com/Contrast
     api_key: A2xxxxxxxxxxxxxxxxxxxxxxxxxxxxG9N
     service_key: 88xxxxxxxxxxx5Z
     user_name: agent_xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx@OrgName
   agent:
     java:
       standalone_app_name: <MyAppName>
   ```

旧 Java エージェントは、Maven Central Repository Search よりダウンロードしてください。

Java エージェントのテレメトリ

Contrast の Java エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集され、その後も定期的に(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切である (964ページ)と考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、Contrast に安全に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能はオプションです。テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	収集されるデータ
Java 3.16.x	• オペレーティングシステムとバージョン • エージェントがコンテナ内で実行されているかどうか • JVM に設定されているメモリ制限 • Java のバージョンとベンダ情報 • 利用可能な物理メモリ • CPU 数

注記

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `true` または `1` を設定してください。テレメトリのデータは、`telemetry.java.contrastsecurity.com` に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

.NET Framework エージェント

Contrast .NET Framework エージェントは、ユーザが.NET Web アプリケーションを操作する際のアプリケーションの動きを解析します。

.NET Framework エージェントをインストールすると、IIS にデプロイされた ASP.NET アプリケーションにエージェントが自動的に組み込まれます。エージェントによる解析は、ユーザによって(または自動化されたスクリプトやテストによって)アプリケーションが操作される時に実行されます。

エージェントの解析結果は、Contrast Web インターフェイスで確認できます。Contrast .NET Framework エージェントは、以下のように複数のコンポーネントで構成されています。

- **バックグラウンドの Windows サービス**：(*DotnetAgentService.exe*)このサービスは、エージェントを組み込むための環境を準備し、エージェントコンポーネント間の通信を管理します。主要となるサービスで、エージェントの動作を制御します。このバックグラウンドの Windows サービスを停止することで、Contrast エージェントの組み込みと解析を無効にできます。
- **.NET プロファイラ**：エージェントセンサーへのメソッドの呼び出しに組み込まれて、アプリケーションを計測します。
- **センサー**： 複数のセンサーが、セキュリティ、アーキテクチャ、ライブラリに関する情報を収集します。
- **.NET Framework Contrast トレイ (220ページ)**：Windows システムトレイアプリケーションで、エージェントの状態に関する情報を概要レベルで表示します。

次のステップ：

- .NET Framework エージェントをインストールする (168ページ)
- サポート対象テクノロジを確認する (167ページ)
- システム要件を確認する (168ページ)
- IIS Express で.NET Framework エージェントを使用する (214ページ)
- IIS でアプリケーションプールを使用する (222ページ)

.NET Framework エージェントのサポート対象テクノロジ

Contrast .NET エージェントは、以下のテクノロジで構築される Web アプリケーションの解析をサポートします。

テクノロジ	サポート対象のバージョン	注記
Windows 版 .NET Framework		
アプリケーションのランタイムバージョン	4.5 以降	古いバージョンの.NET 4 をアプリケーションがターゲットにしている場合でも、.NET Framework アプリケーションの互換性により、ほとんどのアプリケーションで最新の.NET Framework エージェントを使用できます。 **サポート対象外：** • Classic ASP Classic ASP のアプリケーションは.NET ランタイムで動作しません。 • Mono ランタイム Contrast エージェントは、CLR のプロファイル API を使用してアプリケーションを計測します。CLR プロファイル API は、CLR によって公開される COM(Component Object Model)ベースのインターフェイスです。Linux では COM をサポートしません。したがって、Mono は CLR プロファイル API をサポートしないため、Contrast は Mono をサポートできません。
サーバのランタイムバージョン	4.7.1, 4.7.2, 4.8	
CLR	CLR4	
Web サーバ	• IIS • IIS Express	
アプリケーションフレームワーク	• ASP.NET MVC 3-5 • ASP.NET Web Forms • ASP.NET Web Pages • IIS-ホストの ASMX ベース Web サービス • IIS ホストの Web API • IIS ホストの WCF サービス • OWIN ホストの Web API(Windows サービスまたはコマンドラインアプリケーション経由)	これらのフレームワークは、テスト済みのものを明示的に記載していますが、フレームワークが単に典型的な ASP.NET クラス(例えば、System.Web.HttpRequest など)をラップしている場合は、他のアプリケーションを解析できる場合があります。 サポート対象外： • .NET Framework の ASP.NET Core アプリケーションの解析(.NET Core アプリケーションを解析するには.NET Core エージェント (224ページ) を使用)。 • 部分的な信頼(trust)レベルで実行されているアプリケーション。

注記

- Azure App Service の場合、.NET Framework アプリケーションは.NET Framework の拡張機能または NuGet パッケージを使用する必要があります。
  - .NET Core アプリケーションは、.NET Core の拡張機能または NuGet パッケージを使用する必要があります。

.NET Framework エージェントのシステム要件

.NET Framework エージェントをインストールする前に、以下のシステム要件を満たす必要があります。

- Web サーバへの管理者権限があり、そのサーバは Contrast のサポート対象であること。
- 検査対象のアプリケーションがデプロイされており、その Web アプリケーションのテクノロジは Contrast のサポート対象であること。
- IIS を再起動できること。
- Web サーバが Contrast とネットワークで接続されていること。
- サーバが最低限の要件を満たしていること。

要件	推奨	備考
サーバのランタイム バージョン	4.7.1 以降	.NET Framework エージェントのインストールには、.NET 4.7.1 以降を必要としますが、.NET Framework アプリケーション自体の互換性により、.NET 4.5 以降をターゲットとするアプリケーションを解析できます。
オペレーティング システム	• Windows 10 • Windows Server 2012、2012 R2、2016、2019 • Azure Virtual Machines、Cloud Services、Mobile Services • Azure App Service	
プロセッサのアーキテクチャ	• 32 ビット • 64 ビット	64 ビットシステムでは、エージェントを使用して 32 ビットと 64 ビットの両方の Web アプリケーションを解析できます。
CPU	4 個以上	最低：2
メモリ	8 GB 以上 .NET エージェントを使用する場合、検査対象のアプリケーションのメモリ要件がおよそ 2 倍になります。.NET エージェントがインストールされていない場合に、アプリケーションが使用するメモリは全メモリ容量の半分以下である必要があります。	最低：4 GB
サーバ	• .NET Framework 4.7.1 • CLR 4 (.NET 4.0 以降)	これより古いバージョンを使用するサーバでは、旧 .NET Framework エージェントを使用してください。

注記

- .NET Framework エージェントでは CLR プロファイル API を使用して、データとコードフローの解析(SQL インジェクション、XSS、脆弱な暗号化の検出など)や、検査対象のアプリケーションが使用するライブラリやテクノロジを検出します。
- Contrast エージェントは、`agent.dotnet.enable_chaining` の設定を有効にすることで、パフォーマンスツールや APM ツールなどの他の.NET プロファイラエージェントと共存 (217ページ)できます。
- 古いバージョンを使用するサーバでは、旧 .NET Framework エージェントを使用してください。

.NET Framework エージェントのインストール

.NET Framework エージェントのインストールは、通常、ほとんどの環境でインストーラか拡張機能を使用して自動的に行うことができます。.NET Framework エージェントの基本的なインストールは、次のようになります。

1. インストーラをダウンロードし、サーバに置きます。
2. インストーラを実行します。
3. 通常通りにアプリケーションを疎通し、Contrast でアプリケーションが認識されていることを確認します。

注記
Windows 2008 を使用していて、.NET Framework のバージョンが 4.7.1 よりも前である、または、アプリケーションが CLR2 を対象としている場合は、旧.NET Framework エージェントのインストーラを使用してください。

具体的に、.NET Framework エージェントをどのようにインストールするかによって、インストール方法は異なります。

- .NET Framework エージェントの Windows インストーラ (169ページ)
- Azure App Service (172ページ)
- Docker などのコンテナ内 (174ページ)
- Web API-OWIN (176ページ)

エージェントを自動アップグレードするには、エージェントアップグレードサービス (179ページ)のオプションを有効にします。

.NET Framework エージェントの Windows インストーラ

Contrast .NET Framework エージェントのインストーラは、標準の MSI を利用して作られた Windows アプリケーションの一般的なインストーラです。インストーラは、対象となるサーバが要件(サーバのオペレーティングシステムがサポート対象であることなど)を満たしているかを検証します。サーバが全ての要件を満たしていれば、インストーラによって以下が行われます。

- .NET Framework エージェントを Windows の標準プログラムとして登録します。
- 指定されたインストール場所(例、 *C:\Program Files\Contrast\dotnet*)にエージェントのファイルを配置します。これには、いくつかのダイナミックリンクライブラリ(DLL)や、エージェントをバックグランドで動作させる Windows サービスなどの実行ファイルが含まれます。
- エージェントのログファイルや設定を主に保存するためのデータディレクトリ(例、*C:\ProgramData\Contrast\dotnet*)を作成します。
- エージェントをバックグラウンドで実行させる Windows サービスをオペレーティングシステムに登録します。
- エージェントのネイティブモジュールを IIS に追加します。ネイティブモジュールは、環境変数によってエージェントのプロファイラコンポーネントを IIS に登録します。これにより、CLR はエージェントのプロファイラをロードし、解析対象のアプリケーションにエージェントが組み込まれます。
- エージェントの Windows サービスと Contrast トレイ (220ページ)アプリケーションを起動します。このサービスは次の処理を行います。
  - ローカルの名前付きパイプによって、プロファイラとセンサコンポーネントとを通信します。

注記

- OWIN を使用したセルフホスト Web API (176ページ)(IIS 以外)でエージェントを使用する場合は、さらに設定が必要です。
- Windows 2008 を使用していて、.NET Framework のバージョンが 4.7.1 よりも前である、または、アプリケーションが CLR2 を対象としている場合は、旧.NET Framework エージェントのインストーラを使用してください。

Contrast Web インターフェイスから.NET Framework エージェントをインストール：

Contrast Web インターフェイスから.NET Framework エージェントをインストール

1. Contrast Web インターフェイスで、右上の**新規登録**を選択します。
2. **アプリケーション**のカードを選択します。
3. アプリケーション言語のドロップダウンメニューで**.NET Framework** を選択し、**IIS でホスト**を選択したら、**エージェントと YAML 設定ファイルをダウンロード**へのリンクを選択します。
4. ダウンロードした ZIP アーカイブを Web サーバに展開し、*ContrastSetup.exe* を実行します。これで、.NET Framework エージェントがインストールされます。
   インストーラによって、*contrast_security.yaml* ファイルがエージェントのデータディレクトリにコピーされます。デフォルトは、*C:\ProgramData\Contrast\dotnet\contrast_security.yaml* になります。コピー先に既に YAML ファイルが存在する場合は、インストーラは YAML ファイルをコピーしません。
   - コマンドラインを使用 (181ページ)すると、.NET Framework エージェントの Windows インストーラでサポートされるその他のオプションを利用できます。
   - この環境で別のプロファイラ(New Relic や AppDynamics などの APM など)を使用している場合は、Contrast プロファイラチェーン (217ページ)を有効にする必要があります。
5. NET Framework エージェントの YAML テンプレート (183ページ)を使用して、.NET Framework エージェントの設定をより詳細に指定できます。
6. 通常通りにアプリケーションを疎通し、Contrast でアプリケーションが認識されていることを確認します。
   解析する必要のないアプリケーションがある場合や、パフォーマンスの有効活用を考える場合には、アプリケーションプール (222ページ)を利用して検査対象のアプリケーションの数を制限することを検討してください。

コマンドラインから.NET Framework エージェントをインストール

コマンドラインを使用してインストールすると、.NET Framework エージェントの Windows インストーラでサポートされるその他のオプションを利用できます。

.NET エージェントは、Windows のインターフェイスを使用してインストールし、Windows の標準機能(コントロールパネルのプログラムと機能や Powershell など)を使用してアンインストールや修復を行うことができます。ただし、自動化されたスクリプトなどの特定のシナリオでは、.NET Framework エージェントの Windows インストーラを使用して、以下の操作が必要な場合があります。

有人モードには、以下のコマンドを使用します。

- **インストール**：`ContrastSetup.exe`
- **アンインストール**：`ContrastSetup.exe -uninstall`
- **修復**：`ContrastSetup.exe -repair`

無人モードまたはサイレントモードには、以下のコマンドを使用します。

- **インストール**：`ContrastSetup.exe -s -norestart`

- **アンインストール：**`ContrastSetup.exe -uninstall -s -norestart`
- **修復：**`ContrastSetup.exe -repair -s -norestart`

コマンドラインを使用してのインストールでは、.NET Framework エージェントの Windows インストーラで利用できるその他のオプションを指定できます。

オプション	説明	例
`INSTALLFOLDER`	インストール先ディレクトリを指定します。このディレクトリに、プログラムファイルが書き込まれます。デフォルトは OS の変数によって異なります。	`INSTALLFOLDER="D:\Programs\Contrast"`
`AGENT_EXPLORER_INSTALLFOLDER`	Agent Explorer のファイルのディレクトリを指定します。	`AGENT_EXPLORER_INSTALLFOLDER="C:\Program Files\Contrast\agent-explorer"`
`DATAFOLDER`	データディレクトリを指定します。このディレクトリに、ログや `contrast_security.yaml` ファイルが書き込まれます。デフォルトは OS の変数によって異なります。	`DATAFOLDER ="D:\Data\Contrast"`
`PathToYaml`	カスタムの YAML 設定ファイルを指定します。デフォルトの値は、インストーラの場所を基準にして置かれている *contrast_security.yaml* ファイルです。	`PathToYaml=c:\contrast_security.yaml`
`SERVICE_STARTUP_TYPE_MANUAL`	このオプションは、エージェントのインストール、更新、修復を行う場合に必要です。このオプションの値を `1` に設定すると、Contrast サービスのスタートアップの種類がに設定されます。デフォルトの値は `0` で、「自動(遅延開始)」です。	`SERVICE_STARTUP_TYPE_MANUAL=1`
`SUPPRESS_SERVICE_START`	このオプションは、エージェントのインストール、更新、修復を行う場合に必要です。このオプションの値を `1` に設定すると、サービスが自動で起動されなくなります。デフォルトの値は、`0` です。	`SUPPRESS_SERVICE_START=1`
`SUPPRESS_RESTARTING_IIS`	このオプションの値を `1` に設定すると、インストーラは IIS を再起動しません。 デフォルトの値は、`0` です。 **注記** • IIS が再起動されるまで、アプリケーションにエージェントは組み込まれません。 • `SUPPRESS_RESTARTING_IIS` を設定すると、アップグレードの実行時に IIS にアクティブなワーカーがない限り、自動アップグレードが実行されなくなります。	`SUPPRESS_RESTARTING_IIS=0`
`USE_VIRTUAL_SERVICE_ACCOUNT`	制限された仮想サービスアカウントで、エージェントサービスを実行します。`SYSTEM` のの代わりに、仮想アカウント `NT Service\DotnetAgentSvc` としてサービスを実行するように構成します。	`USE_VIRTUAL_SERVICE_ACCOUNT=0`
`INSECURE_YAML_FILE`	昇格していないユーザの `contrast_security.yaml` ファイルへの編集を制限します。	`INSECURE_YAML_FILE=0`
`INSTALL_AGENT_EXPLORER`	Agent Explorer をインストールしない場合は、このオプションの値を `0` に設定して下さい。 デフォルトの値は `1` で、Agent Explorer はインストールされます。	`INSTALL_AGENT_EXPLORER=1`
`INSTALL_UPGRADE_SERVICE`	エージェントアップグレードサービスをインストールしない場合は、このオプションの値を `0` に設定して下さい。デフォルトの値は `1` で、エージェントアップグレードサービスはインストールされます。	`INSTALL_UPGRADE_SERVICE=1`

オプション	説明	例
UPGRADE_SERVICE_INSTALLFOLDER	アップグレードサービスのファイルのディレクトリを指定します。	UPGRADE_SERVICE_INSTALLFOLDER="C:\Program Files (x86)\Contrast\upgrade-service"

ヒント
例えば、スクリプトを使用して.NET エージェントをインストールするには、以下のコマンドを使用します。

```
ContrastSetup.exe -s PathToYaml=C:\Temp\custom.yaml
```

このコマンドでは、.NET エージェントのインストールを無人/サイレントモードで行い、YAML 設定ファイルのカスタムパスを使用します。

重要
エージェントをインストールすると、Contrast は自動的に IIS を再起動します。

Azure App Service で.NET Framework エージェントをインストール

注記
拡張機能を利用できない場合は、NuGet を使用して.NET Framework エージェントを手動でインストール (176ページ)できます。

手順

Azure Portal の拡張機能を使用して.NET Framework エージェントをインストールするには：

1. Azure App Service でホストするアプリケーションを作成します。
   これを行うには、Azure アカウントを持っており、ASP.NET Framework の Web アプリケーションを作成する必要があります。アプリケーションを Azure に公開し、Contrast エージェントなしで想定通りに機能することを確認します。
2. アプリケーションのメニューより**構成**を選択し、**アプリケーション設定**の画面で以下の値を追加し、エージェントが Contrast サーバに接続するための設定をします。

キー	値
CONTRAST__API__USER_NAME	自分のエージェントユーザ名 (61ページ)を指定します。
CONTRAST__API__SERVICE_KEY	自分のエージェントサービスキー (61ページ)を指定します。
CONTRAST__API__API_KEY	自分の API キー (61ページ)を指定します。
CONTRAST__API__URL	デフォルトは、https://app.contrastsecurity.com です。別の場所でホストされている Contrast サーバを使用する場合は、その URL を指定します。

3. App Service にデプロイ中のアプリケーションを選択します。
4. メニュー画面より、**拡張機能**を選択します。
5. **+追加**を選択します。

6. 拡張機能の選択をクリックし、一覧より **Contrast .NET Site Extension** を選択します。 これが、.NET Framework アプリケーション用の拡張機能になります。
7. 法律条項に同意して、**OK** ボタンをクリックします。
8. 拡張機能のインストールが完了するまでしばらく待ったら、正しくインストールされていることを確認します。
9. アプリケーションの概要に戻り、アプリケーションを**再起動**します。
10. アプリケーションを疎通し、検査結果が Contrast に報告されることを確認します。

ヒント

アプリケーションの SCM サイト(Kudu)の「Site Extensions」メニューからも Contrast エージェントをインストールできます。

重要

Contrast エージェントの新しいバージョンが利用可能な場合、 Azure Portal または Kudu のダッシュボードに通知されます。エージェントの更新を行う前にサイトを停止してください。停止しない場合、更新が失敗する可能性があります。

注記

サイトの拡張機能により、以下のような環境変数がいくつか設定されます。

```
COR_ENABLE_PROFILING=1
COR_PROFILER={EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}
COR_PROFILER_PATH_32=D:\home\siteextensions\Contrast.Net.Azure.S
iteExtension\ContrastAppService\runtimes\win-
x86\native\ContrastProfiler.dll
COR_PROFILER_PATH_64=D:\home\siteextensions\Contrast.Net.Azure.S
iteExtension\ContrastAppService\runtimes\win-
x64\native\ContrastProfiler.dll
CONTRAST_INSTALL_DIRECTORY=D:\home\siteextensions\Contrast.Net.A
zure.SiteExtension\ContrastAppService\
MicrosoftInstrumentationEngine_ConfigPath32_ContrastX86Config=D:
\home\siteextensions\Contrast.Net.Azure.SiteExtension\runtimes\w
in-x86\ContrastCieProfiler.config
MicrosoftInstrumentationEngine_ConfigPath64_ContrastX64Config=D:
\home\siteextensions\Contrast.Net.Azure.SiteExtension\runtimes\w
in-x64\ContrastCieProfiler.config
```

CLR Instrumentation Engine(CIE)がアプリケーションに設定されている場合(例えば、Application Insights が有効になっているために)、Azure は自動的に `COR_PROFILER*` 変数を上書きし、CIE のプロファイラを指すようにします。

そして、CIE では、`MicrosoftInstrumentationEngine_*`変数を使用して Contrast エージェントをロードするようになります。

CIE がアプリケーションに設定されていない場合は、Contrast エージェントのロードに標準の `COR_PROFILER*`変数が使用されます。

コンテナを使用して.NET Framework エージェントをインストール

インストールを行う前に

本項では、Docker を例として、コンテナ化されたアプリケーションに Contrast .NET Framework エージェントをインストールするための一般的な手順について説明します。

コンテナや関連ソフトウェアの仕組みを基本的に理解している必要があります。必要に応じて、お客様の環境に合わせて手順を調整してください。

手順 1：エージェントをインストール

コンテナイメージにアプリケーションを追加する前でも後でもインストールすることができます。推奨される方法は、名前を付けたマルチステージビルドを使用することです。 例：

```
FROM mcr.microsoft.com/dotnet/framework/sdk:4.8

# Hidden for brevity...

# Copy the required agent files from the official Contrast agent image.
COPY --from=contrast/agent-dotnet-framework:latest C:\Contrast C:\Contrast
```

この例では、最新の.NET Framework エージェントをコピーしています。 利用可能なタグは DockerHub で確認してください。

手順 2：エージェントを設定

Contrast エージェントは、複数のソースからの設定を受け入れますが、設定の優先順位 (62ページ)は優先順位セクションに記載されています。

設定方法を組み合わせて利用することをお勧めします。

- YAML ファイルを使用して、複数のアプリケーションで共有する共通の設定を指定します。
- アプリケーション固有の設定値や、YAML ファイルで指定した値を上書きする場合や、実行時に組み込む機密情報などには、環境変数を使用します。

YAML ファイルの設定：

YAML ファイル (63ページ)を使用してエージェントを設定する場合に、環境変数の `CONTRAST_CONFIG_PATH` を使用して、YAML ファイルがコンテナ内のどこにあるかを指定することもできます。

例えば、`contrast_security.yaml` という名前の YAML 設定ファイルが Docker のビルドコンテキストに存在するとします。

```
agent:
  logger:
    path: /var/tmp
    level: WARN
```

環境変数 `CONTRAST_CONFIG_PATH` を使用して、以下のようにコンテナイメージにエージェント YAML ファイルを追加することができます。

```
FROM mcr.microsoft.com/dotnet/framework/sdk:4.8

# Hidden for brevity...

# Add the Contrast agent to the image.
COPY --from=contrast/agent-dotnet-framework:latest C:\Contrast C:\Contrast

# Copy the contrast_security.yaml file from Docker build context.
```

```
COPY ./contrast_security.yaml /contrast_security.yaml

# Finally configure configure the agent to use the YAML file previously \
copied.
ENV CONTRAST_CONFIG_PATH=/contrast_security.yaml
```

環境変数の設定：

アプリケーション固有の設定を指定するために、環境変数 (66ページ)を使用します。以下は、一般的によく使用される設定オプションです。

項目	用途	環境変数
アプリケーション名	Contrast サーバに報告されるアプリケーション名を指定します。	CONTRAST__APPLICATION__NAME
アプリケーショングループ	オンボード時にこのアプリケーションと関連付けるアクセスグループを指定します。 **注記** この変数を使用する前に、Contrast でアプリケーションアクセスグループを作成してください。	CONTRAST__APPLICATION__GROUP
アプリケーションのタグ	アプリケーションにタグを追加します。	CONTRAST__APPLICATION__TAGS
サーバ名	Contrast に報告されるサーバ名を指定します。	CONTRAST__SERVER__NAME
サーバの環境	アプリケーションを実行する環境を指定します。このオプションで有効な値：Development、QA、Production	CONTRAST__SERVER__ENVIRONMENT
サーバのタグ	サーバにタグを追加します。	CONTRAST__SERVER__TAG

手順 3：プロファイル変数と認証情報を追加

アプリケーションに.NET エージェントを組み込んで検査を行うには、さらに環境変数 (66ページ)が必要です。COR_CLR 変数はエージェントをロードし、CONTRAST_ 変数は Contrast サーバに対するエージェントの認証用です。

前述の Dockerfile の例を使用すると、以下のようになります。

```
FROM mcr.microsoft.com/dotnet/framework/sdk:4.8

COPY --from=contrast/agent-dotnet-framework:latest C:\Contrast C:\Contrast

ENV COR_ENABLE_PROFILING=1 \
    COR_PROFILER={EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1} \
    COR_PROFILER_PATH_32=C:\Contrast\runtimes\win-
x86\native\ContrastProfiler.dll \
    COR_PROFILER_PATH_64=C:\Contrast\runtimes\win-
x64\native\ContrastProfiler.dll
```

さらに、サーバに対するエージェント認証のために、以下の環境変数 (61ページ)が必要です。

```
CONTRAST__API__URL=https://app.contrastsecurity.com/Contrast
CONTRAST__API__API_KEY={API }
CONTRAST__API__SERVICE_KEY={}
```

API の値(エージェントキー (61ページ))は Contrast Web インターフェイスで確認するか、.NET Framework エージェント用の YAML ファイルをダウンロード (183ページ)することで取得できます。

例

こちらの GitHub リポジトリで、設定が完了したコードの例を参照できます。特に、以下の ASP.NET アプリケーションのサンプルなどを参考にしてください。

- デフォルトのアプリケーションプールを使用：
  - Dockerfile
  - エントリポイントスクリプト
- カスタムのアプリケーションプールを使用：
  - Dockerfile
  - エントリポイントスクリプト

関連項目

Contrast サポートポータル：Kubernetes での Contrast エージェント

Contrast サポートポータル：AWS Fargate と Contrast エージェント

NuGet で.NET Framework エージェントを手動インストール

.NET Framework エージェントは、NuGet を使用して手動でインストールすることもできます。このインストール方法は、Azure App Service の拡張機能 (172ページ)を利用できない場合や、.NET Framework エージェントを依存関係に含めたい場合などに便利です。

1. アプリケーションに Contrast の NuGet パッケージをインストールします。
   Visual Studio で、ソリューションエクスプローラーのアプリケーションのプロジェクトで**参照**を右クリックし、**NuGet パッケージの管理**を選択します。
   Contrast.Net.Azure.AppService パッケージを検索して選択し、プロジェクトに追加します。
   アプリケションをビルドします。Contrast アセンブリ(例えば、`ContrastProfiler.dll`)が、アプリケーションのルートディレクトリに新規に作成された `contrastsecurity` フォルダにあることを確認します。
2. Contrast サーバに対するアプリケーションの認証情報を追加します。
   認証情報の設定は、VisualStudio の「Azure App Service に公開する」の App Service 設定画面から追加するか、Azure App Service ポータルから直接追加できます。
   エージェントが Contrast に接続するために必要な Contrast 認証キーを設定し、**保存**を選択します。
   プロファイル画面でキーを見つける (61ページ)ことができます。
3. dotnet ソースコードリポジトリ のビルドプロセスに従います。
4. Azure Portal でお使いのアプリケーションの**アプリケーション設定**セクションにアクセスします。
   エージェントが Contrast に接続するために必要な Contrast 認証キーを設定し、**保存**を選択します。
5. Visual Studio を使用して、Azure にアプリケーションを公開します。
   アプリケーションがロードされたら、アプリケーションを疎通してから、Contrast を開き、サーバとアプリケーションがアクティブになっていること、および何か脆弱性が表示されていることを確認します。

Web API および Owin で.NET Framework エージェントをインストール

.NET Framework エージェントは、Open Web Interface for .NET (OWIN)でセルフホストされる Web API アプリケーションの解析をサポートします。Web API は、コマンドラインアプリケーションや Windows サービスとしてデプロイできます。

注記

SystemWeb の HttpModule を使用して IIS 統合パイプラインにホストされている Web API アプリケーションや、OWIN ホストでデプロイされている Web API アプリケーションは**サポート対象外**です。

1. .NET Framework エージェントの Windows インストーラ (169ページ)を使用して、.NET Framework エージェントをインストールします。
2. OWIN ホストの Web API をどのようにデプロイするかによって、環境変数を設定します。
   - **コマンドラインアプリケーションとしてデプロイ**：OWIN をセルフホストにするために使用されるコマンドラインアプリケーションを実行する前に、以下の環境変数を設定します。

環境変数	値
COR_ENABLE_PROFILING	1
COR_PROFILER	{EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}
COR_PROFILER_PATH_32	C:\Program Files\Contrast\dotnet\runtimes\win-x86\native\ContrastProfiler.dll
COR_PROFILER_PATH_64	C:\Program Files\Contrast\dotnet\runtimes\win-x64\native\ContrastProfiler.dll

注記
COR_PROFILER_PATH_32 / COR_PROFILER_PATH_64 は、.NET Framework エージェントのインストールで選択したインストールディレクトリと一致する必要があります。

   - **Windows サービスとしてデプロイ**：
     Web API アプリケーションを含むサービスをインストールします。サービスの名前を確認します。
     サービスのレジストリキーの下に、Environment という名前の REG_MULTI_SZ 値を作成します。既に Environment 値がある場合は、既存の値の下に新しい値を追加します。
     必要な環境変数を設定します。各環境変数はキーと値をペアにして、それぞれ改行してください。各サービスに固有の環境変数は、そのサービスのレジストリキーの下に設定できます。サービスのレジストリキーは次の場所にあります：
     HKLM\SYSTEM\CurrentControlSet\Services\YourServiceName

環境変数	値
COR_ENABLE_PROFILING	1
COR_PROFILER	{EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}
COR_PROFILER_PATH_32	C:\Program Files\Contrast\dotnet\runtimes\win-x86\native\ContrastProfiler.dll
COR_PROFILER_PATH_64	C:\Program Files\Contrast\dotnet\runtimes\win-x64\native\ContrastProfiler.dll
CONTRAST_CONFIG_PATH	C:\ProgramData\contrast\dotnet\contrast_security.yaml

注記
COR_PROFILER_PATH_32 / COR_PROFILER_PATH_64 は、.NET Framework エージェントのインストールで選択したインストールディレクトリと一致する必要があります。

3. サービスを再起動して、新しい値をロードします。必要な環境変数を設定するために、以下の PowerShell スクリプトを使用できます。

```
param (
    # Name of the service that it was given at installation.
    [Parameter(Mandatory=$true)]
    [string]
    $ServiceName,
```

```
    # Path to the 64-bit Contrast profiler DLL.
    # Defaults to: "C:\Program Files\Contrast\dotnet\runtimes\win-
x64\native\ContrastProfiler.dll"
    [string]
    $ProfilerPath64 = "C:\Program Files\Contrast\dotnet\runtimes\win-
x64\native\ContrastProfiler.dll",

    # Path to the 32-bit Contrast profiler DLL.
    # Defaults to: "C:\Program Files\Contrast\dotnet\runtimes\win-
x86\native\ContrastProfiler.dll"
    [string]
    $ProfilerPath32 = "C:\Program Files\Contrast\dotnet\runtimes\win-
x86\native\ContrastProfiler.dll",

    # Path to the Contrast agent configuration YAML file.
    # Defaults \
to: "C:\ProgramData\contrast\dotnet\contrast_security.yaml"
    [string]
    $ConfigYamlPath \
= "C:\ProgramData\contrast\dotnet\contrast_security.yaml"
)

if (-Not (Test-Path -Path $ProfilerPath64 -PathType Leaf)) {
    Write-Host "Cannot find 64-bit profiler DLL at path \
`"$ProfilerPath64`"."
    exit 1
}

if (-Not (Test-Path -Path $ConfigYamlPath -PathType Leaf)) {
    Write-Host "Cannot find configuration YAML file at path \
`"$ConfigYamlPath`"."
    exit 1
}

if (-Not (Test-Path -Path $ProfilerPath32 -PathType Leaf)) {
    Write-Host "Cannot find 32-bit profiler DLL at path \
`"$ProfilerPath32`"."
    exit 1
}

# Check if there is a service with the specified name installed.
$service = Get-Service -Name $ServiceName -ErrorAction Ignore

if ($null -Eq $service) {
    Write-Host "The service `"$ServiceName`" was not found."
    exit 2
}

# Create value for multiline registry string.
$values = @(
    "COR_ENABLE_PROFILING=1",
    "COR_PROFILER={EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}",
    "COR_PROFILER_PATH_64=$ProfilerPath64",
    "COR_PROFILER_PATH_32=$ProfilerPath32",
    "CONTRAST_CONFIG_PATH=$ConfigYamlPath"
```

```
)

$registryKey = "HKLM:\SYSTEM\CurrentControlSet\Services\$ServiceName"

# Check if the Environment value already exists.
$environmentValue = Get-ItemProperty -Path $registryKey -
Name "Environment" -ErrorAction Ignore

if ($null -Ne $environmentValue) {
    # Add the Contrast environment variables to the existing variables.
    $existingValues = \
[System.Collections.ArrayList]@($environmentValue.Environment)
    foreach ($item in $values) {
        $idx = $existingValues.Add($item)
    }
    $values = $existingValues
}

# Set the environment variables for the service.
Set-ItemProperty -Path $registryKey -Type MultiString -
Name "Environment" -Value $values

# Restart the service so it picks up the new environment variables.
Restart-Service -Name $ServiceName
```

エージェントアップグレードサービス

エージェントアップグレードサービスは、Windows のバックグラウンドサービスで、Windows 上の.NET Framework エージェントおよび IIS 用.NET Core エージェントを最新のバージョンに自動的に更新することができます。エージェントアップグレードサービスは、.NET Framework エージェントのインストーラと IIS 用.NET Core エージェントのインストーラに含まれており、エージェントインストーラは以下の 2 つの製品をインストールします。

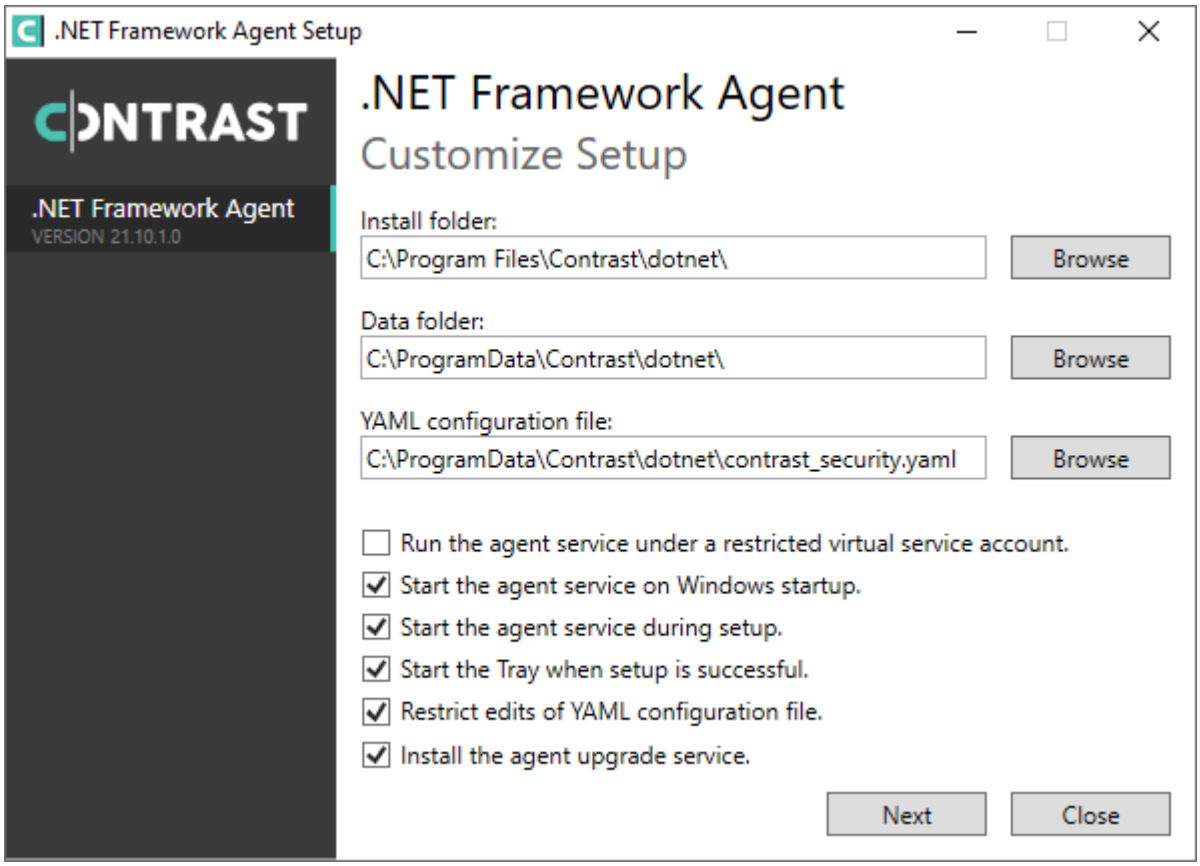
- 該当のエージェント
- エージェントアップグレードサービス

デフォルトでは、エージェントアップグレードサービスは、サービスの初回起動時(Windows Server の再起動時)に、NuGet にリリースされている新しいエージェントをチェックします。新しいエージェントのバージョンが見つかった場合、アップグレードサービスは、新しいバージョンをダウンロードし、インストーラの署名を検証してから、最後にインストーラを実行します。

注記
新しいバージョンのエージェントがインストールされると、IIS は再起動されます。

エージェントアップグレードサービスはオプションコンポーネントであり、エージェントの Assess および Protect 機能には必須ではありません。

- エージェントアップグレードサービスを使用しない場合は、**Install the agent upgrade service**(エージェントアップグレードサービスをインストールする)のチェックボックスをオフにしてください。
- コマンドラインでエージェントをインストールする場合は、`INSTALL_UPGRADE_SERVICE=0` の引数を追加すれば、エージェントアップグレードサービスはインストールされません。



エージェントアップグレードサービスの動作は、Contrast のデータディレクトリにあるエージェント用の設定ファイルで変更できます。デフォルトの場所は、`C:\ProgramData\Contrast\upgrade-service` です。

.NET Core エージェントをアップグレードする際の設定は、.NET Core エージェントの YAML ファイル内にあります。

```
enable: true #  Set to `true` for the agent to automatically upgrade to \
newer versions.
checks: Startup # Set the frequency with which the agent checks for \
updates. Valid values are `daily` for every 24 hours and on startup, or \
`startup` for *only* when service starts up.
timeout_ms: 60000 # Set the time allocated to execute the downloaded agent \
installer before cancelling.
nuget_repository_url: https://api.nuget.org/v3/index.json # Set the URL of \
the Nuget repository to be used for the .NET Core Agent for IIS Installer
nuget_package_name: Contrast.CoreIIS.Installer # Set the name of the .NET \
Core Agent for IIS Nuget package.
installer_upgrade_code: 82468c04-dfc0-4a4c-9eb9-c4b314c67fdc # Used \
internally to retrieve the current installed agent version from Windows.
```

注記

エージェントアップグレードサービスは、エージェントインストーラにのみ含まれます。手動で取得する.NET Core エージェント、エージェントの NuGet パッケージ、Azure App Service のサイト拡張機能には含まれません。

.NET Framework エージェントのアップデート

.NET Framework エージェントをアップデートするには、次のいずれかの方法を使用します。

- 自動的にエージェントを更新
- Contrast API を使用してコマンドでダウンロードとインストール

開始する前に

- .NET Framework アプリケーションが、Contrast の.NET Framework エージェントなしで正常に動作することを確認してください。
- Contrast の.NET Framework エージェントを正常にインストールしたことがあること。
- 変更管理ポリシーと使用する環境に基づいて、エージェントをアップデートする方法とタイミングを決めておくこと。
- PowerShell を含む Windows スクリプトにある程度の知識があること。

エージェントを自動更新

エージェントは、エージェントアップグレードサービス (179ページ)によって自動的に更新されます。対象のエージェントは、バージョン 20.5.1 以降です。ただし、サポートされていない環境 (167ページ)の場合、エージェントは自動更新されません。

Contrast API によるエージェントのダウンロード

1. Contrast API から直接 Contrast エージェントをダウンロードします。この手順は、Contrast の SaaS 版とオンプレミス版の両方で有効な方法です。
   Contrast API にアクセスする(サービスまたはユーザ)ためのアカウントが必要です。
2. エージェントのダウンロード後、コマンドラインからサイレントインストールを行います。
   アカウントの資格情報は、組織のキーと個人のキーの確認 (522ページ)で参照できます。

関連項目

- エージェントアップグレードサービス (179ページ)

.NET Framework エージェントを設定する

エージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。

.NET Framework エージェントを設定します：

- Azure App Service で設定する場合 (181ページ)
- `web.config` ファイルで設定する場合 (182ページ)
- YAML テンプレートを使用する場合 (183ページ)

ヒント

エージェント設定エディタ (64ページ)を使用すると、YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

Azure App Service での.NET Framework エージェントの設定

Azure Portal で Azure App Service の.NET Framework エージェントを設定するには、3 つの方法があります。

- Contrast エージェントの環境変数の表記規則を使用します。Azure Portal の**構成**画面より**アプリケーション設定**セクションに、エージェントの環境変数の構文 (66ページ)を使用して全ての設定を追加します。

- アプリケーションの web.config ファイルに、`application` という設定オプションを指定します。web.config に指定したアプリケーションの設定を Contrast エージェントに認識させるためには、これらの設定をアプリケーション の web.config ファイルのルートの `appSettings` セクション内に記述する必要があります。詳細は、web.config ファイルで設定 (182ページ)を参照してください。
- Azure Portal で個々のオプションを設定する代わりに、Contrast の設定を含む YAML 設定ファイルを使用することもできます。まず、ファイルをアプリケーションのデプロイメントに含めるか、Kudu コンソールを使用して、Azure Web アプリケーションにこのファイルをアップロードします。次に、このファイルの場所を指す設定である `CONTRAST_CONFIG_PATH` を追加します。
  例えば、アプリケーションのルートにある *contrast_security.yaml* ファイルを使用する場合は、`CONTRAST_CONFIG_PATH` というキーに *D:\Home\site\wwwroot\contrast_security.yaml* という値で、アプリケーションの設定を追加します。Azure App Service のアプリケーションファイルは、*D:\home\site\wwwroot* にデプロイされます。

.NET Framework エージェントを web.config ファイルで設定

エージェントの設定オプションは、アプリケーションの *web.config* ファイルまたは YAML 設定ファイルを使用して指定できます。*web.config* に指定したアプリケーションの設定を Contrast エージェントに認識させるためには、これらの設定をアプリケーション *web.config* ファイルのルート<configuration>の `appSettings` セクション内に記述する必要があります。

例えば、同じアプリケーションプールにホストされている 2 つのアプリケーションは、各アプリケーションの *web.config* ファイルで `appSettings` に `contrast.server.name` プロパティを設定すると異なるサーバとして報告されます。また、 *web.config* を使用して次のように `contrast.application.name` を指定することもできます。

```
<configuration>
   <appSettings>
     <add key="contrast.application.name" value="MyWebAppName" />
     <add key="contrast.application.version" value="1.2.3" />
   </appSettings>
   <system.web>
     ...
```

その他の利用可能なプロパティの説明は、.NET Framework エージェントの YAML テンプレート (183ページ)を参照してください。

エージェントのバージョンが 21.1.4 より前の場合、*web.config* で設定できるのは、以下のように一部のプロパティのみとなります。

プロパティ	導入された.NET Framework エージェントのバージョン
`contrast.application.code`	19.6.3
`contrast.application.group`	19.1.3
`contrast.application.metadata`	19.1.3
`contrast.application.name`	19.1.3
`contrast.application.session_id`	20.6.6
`contrast.application.session_metadata`	20.6.6
`contrast.application.tags`	19.1.3
`contrast.application.version`	19.1.3
`contrast.assess.tags`	19.1.3
`contrast.inventory.tags`	19.1.3

注記

`contrast.application.name` が指定されていない場合、.NET Framework エージェントは、アプリケーションの仮想パスをアプリケーション名として使用します。アプリケーションがサイトのルートでホストされている場合(つまり、仮想パスが/の場合)、.NET Framework エージェントはこのサイトの名前をアプリケーション名として使用します。

重要

エージェントのバージョン 21.1.4 以降、ほとんどのエージェントの設定が、アプリケーションの web.config ファイル、またはアプリケーションと同じディレクトリにある `contrast_security.yaml` で指定できるようになりました。例えば、同じアプリケーションプールにホストされている 2 つのアプリケーションは、各アプリケーションの web.config ファイルの `appSettings` に `contrast.server.name` を設定すると異なるサーバとして報告されます。

以下の設定オプションは、*process* レベルで適用され、アプリケーションごとに個別に指定できません。これらのプロパティは、*web.config* を使用して設定できません。別の方法(YAML など)で設定する必要があります。

- agent.dotnet.app_pool_denylist
- agent.dotnet.app_pool_allowlist
- agent.dotnet.enable_instrumentation_optimizations
- agent.dotnet.enable_jit_inlining
- agent.dotnet.enable_transparency_checks
- agent.dotnet.enable_struct_dataflow
- assess.enable_control_detection

また、以下のキーについては、エージェントのプロファイラコンポーネントはプロセスレベルの設定を使用し、エージェントのセンサーコンポーネントはアプリケーション個別の設定(指定されている場合)を使用します。

- agent.logger.level
- agent.logger.stdout

.NET Framework エージェントの YAML テンプレート

YAML 設定 (63ページ)ファイルを使用して、.NET Framework エージェントを設定します。

contrast_security.yaml ファイルは、インストーラによってエージェントのデータディレクトリ にコピーされます(デフォルトでは、*C:\ProgramData\Contrast\dotnet\contrast_security.yaml*)。コピー先に既に YAML ファイルが存在する場合は、インストーラは YAML ファイルをコピーしません。

以下のテンプレートには、このエージェントで有効な YAML オプションがすべてあります。例えば、このファイルを使用して.NET Framework エージェントによって報告されるサーバ名を設定することができます。その場合、*contrast_security.yaml* ファイルに新しい行として以下のコードを追加してファイルを更新してから、通常通りにインストールを続けます。

```
server:
  name: MyServerName
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the version of the TLS protocol the agent uses to communicate with \
the
  # Contrast UI. The .NET agent default behavior is \
(SecurityProtocolType.Tls
  # | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls12).
  # tls_versions: tls|tls11|tls12

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
```

```
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Determine the location from which the agent loads a client
    # certificate. Value options include `File` or `Store`.
    # certificate_location: NEEDS_TO_BE_SET

    # Set the absolute path to the client certificate's
    # .CER file for communication with Contrast UI. The
    # `certificate_location` property must be set to `File`.
    # cer_file: NEEDS_TO_BE_SET

    # Specify the name of certificate store to open. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `AuthRoot`, `CertificateAuthority`,
    # `My`, `Root`, `TrustedPeople`, or `TrustedPublisher`.
    # store_name: NEEDS_TO_BE_SET

    # Specify the location of the certificate store. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `CurrentUser` or `LocalMachine`.
    # store_location: NEEDS_TO_BE_SET

    # Specify the type of value the agent uses to find the certificate
    # in the collection of certificates from the certificate store.
    # The `certificate_location` property must be set to `Store`.
    # Value options include `FindByIssuerDistinguishedName`,
    # `FindByIssuerName`, `FindBySerialNumber`,
    # `FindBySubjectDistinguishedName`, `FindBySubjectKeyIdentifier`,
    # `FindBySubjectName`, or `FindByThumbprint`.
    # find_type: NEEDS_TO_BE_SET

    # Specify the value the agent uses in combination with
    # `find_type` to find a certification in the certificate store.
    #
    # Note - The agent will use the first certificate from
    # the certificate store that matches this search criteria.
    #
    # find_value: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:
```

```
    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
==========================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
==========================================================================
==
# agent:

  # \
==========================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
==========================================================================
  # logger:

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10
```

```
  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING
```

```
    # Set the log level of Suspicious attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_suspicious: WARNING

    # Set the connection type used for Syslog messages.
    # Value options are `UNENCRYPTED` and `ENCRYPTED`.
    # connection_type: UNENCRYPTED

  # \
================================================================================
  # agent.dotnet
  # The following properties apply to any .NET agent-wide configurations.
  # \
================================================================================
  # dotnet:

    # Set a list of application pool names that the agent does not
    # instrument or analyze. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_denylist: NEEDS_TO_BE_SET

    # Set a list of application pool names that the agent instruments or
    # analyzes. If set, other application pools are ignored. Allowlist takes
    # precedence over denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_allowlist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent does not
    # analyze. (The applications are still instrumented).
    # Names must be formatted as a comma-separated list.
    # New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_denylist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent analyzes.
    # If set, other applications are not analyzed, but are
    # still instrumented. Allowlist takes precedence over
    # denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_allowlist: NEEDS_TO_BE_SET

    # Enable a profiler chaining feature to allow Contrast to
    # work alongside other tools that use the CLR Profiling
    # API. Defaults to `true`. New after .NET Framework 19.1.3
    # (Installed Only) and .NET Core 1.9.3 (Installed Only).
    # enable_chaining: true

    # Indicate that the agent should produce a report that
    # summarizes application hosting on the server (e.g.,
    # CLR versions, bitness or pipeline modes). Defaults to
    # `true`. New after .NET Framework 19.1.3 (Installed Only).
    # enable_dvnr: true

    # Indicate that the agent should allow CLR optimizations
    # of JIT-compiled methods. Defaults to `true`. New
```

```
# after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_instrumentation_optimizations: true

# Indicate that the agent should allow the CLR to inline
# methods that are not instrumented by Contrast. Defaults to
# `true`. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_jit_inlining: true

# Indicate that the agent should allow the CLR to perform
# transparency checks under full trust. Defaults to `false`.
# New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_transparency_checks: false

# Responses for request paths (e.g., HttpRequest.Path)
# that match this regex are not analyzed. Defaults to
# `WebResource.axd`. New after .NET Framework 19.1.3.
# web_module_allowlist: WebResource.axd

# Set to display ASCII art to std::out on agent startup. Defaults
# to `true`. New after .NET Framework 20.6.3 and .NET Core 1.0.0.
# enable_cat: true

# Sets the maximum amount of time a Protect regular expression
# is allowed to run before being cancelled. Set to -1 to never
# cancel regular expression execution. Defaults to `20_000`.
# New after .NET Framework 20.4.3 and .NET Core 1.5.0.
# protect_searchers_single_pattern_deadline_ms: 20_000

# Sets the maximum amount of time a 'Probe Analysis' Protect
# regular expression is allowed to run before being cancelled. Set
# to -1 to never cancel regular expression execution. Defaults to
# `5_000`. New after .NET Framework 20.7.3 and .NET Core 1.5.11.
# protect_searchers_probe_analysis_single_pattern_deadline_ms: 5_000

# Sets the maximum amount of time a Protect rule is
# allowed to run before being cancelled. Set to -1 to never
# cancel Protect rule execution. Defaults to `60_000`.
# New after .NET Framework 20.4.3 and .NET Core 1.5.0.
# protect_searchers_total_rule_deadline_ms: 60_000

# Sets the maximum amount of time a 'Probe Analysis' Protect
# rule is allowed to run before being cancelled. Set to -1 to
# never cancel Protect rule execution. Defaults to `10_000`.
# New after .NET Framework 20.7.3 and .NET Core 1.5.11.
# protect_searchers_probe_analysis_total_rule_deadline_ms: 10_000

# Sets the maximum duration of time agent log files should be kept
# since last write before being deleted by the agent. Defaults to
# `604_800_000`. New after .NET Framework 20.6.1 and .NET Core 1.5.5.
# log_cleanup_maximum_age_ms: 604_800_000

# Suppresses gathering process-level metrics (process level metrics are
# gathered by default), used to identify performance problems. Metric
# counters may further decrease the stability of already unstable
# systems and can be disabled (set to true) if issues occur. Defaults
```

```
    # to `false`. New after .NET Framework 20.6.6 and .NET Core 1.5.10.
    # suppress_metric_counters: false

    # Enable file based application watching. Set to false if
    # file watching is causing locking issues. Defaults to `true`.
    # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # enable_file_based_app_watching: true

    # Enables HttpClient isolation using AppDomain remoting. This can be \
used
    # to workaround .NET TLS version limitations at the cost of performance
    # and stability. Enabled by default on applications targeting .NET
    # Framework < 4.7.0, else disabled. New after .NET Framework 21.5.1.
    # enable_http_client_app_domain_isolation: false

    # Enables LINQ optimizations to improve performance
    # at the cost of possible false negatives. Defaults
    # to `true`. New after .NET Framework 50.0.1.
    # enable_linq_optimizations: true

    # \
================================================================================
    # agent.dotnet.file_analysis_time_ms
    # Controls the interval in milliseconds to perform file
    # analysis for supported rules. Setting a value > 0 will
    # result in the job running at that interval and not just when
    # the application loads. If set to `-1`, the job just runs
    # once. Defaults to `-1`. New after .NET Framework 50.0.15.
    # \
================================================================================
    # file_analysis_time_ms: {}

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Specifies the cloud provider from which the agent should gather metadata
  # (such as resource identifiers). Options are `AWS`, `Azure`, or `GCP`.
  #
  # gather_metadata_via: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
```

```
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false
```

```
    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-chaining
    # Use the following properties to configure how the
    # sql injection semantic analysis chaining rule works.
    # \
================================================================================
    # sql-injection-semantic-chaining:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-dangerous-functions
    # Use the following properties to configure how the sql
    # injection semantic analysis dangerous functions rule works.
    # \
================================================================================
    # sql-injection-semantic-dangerous-functions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-suspicious-unions
    # Use the following properties to configure how the sql
    # injection semantic analysis suspicious unions rule works.
    # \
================================================================================
    # sql-injection-semantic-suspicious-unions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off
```

```
    # \
================================================================================
    # protect.rules.sql-injection-semantic-tautologies
    # Use the following properties to configure how the sql
    # injection semantic analysis tautologies rule works.
    # \
================================================================================
    # sql-injection-semantic-tautologies:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
    # \
================================================================================
```

```
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
```

```
# ============================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
# ============================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
# ============================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
# ============================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
# ============================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
# ============================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
# ============================================================================
    # xxe:

      # Set the mode of the rule. Value options are
```

```
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
```

```
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET
```

```
  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Override the reported server path. New after
  # .NET Framework v21.3.1 and .NET Core v1.8.0.
  # path: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET
```

```
  # Set the version of the TLS protocol the agent uses to communicate with \
the
  # Contrast UI. The .NET agent default behavior is \
(SecurityProtocolType.Tls
  # | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls12).
  # tls_versions: tls|tls11|tls12

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Determine the location from which the agent loads a client
    # certificate. Value options include `File` or `Store`.
    # certificate_location: NEEDS_TO_BE_SET

    # Set the absolute path to the client certificate's
    # .CER file for communication with Contrast UI. The
    # `certificate_location` property must be set to `File`.
    # cer_file: NEEDS_TO_BE_SET

    # Specify the name of certificate store to open. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `AuthRoot`, `CertificateAuthority`,
    # `My`, `Root`, `TrustedPeople`, or `TrustedPublisher`.
    # store_name: NEEDS_TO_BE_SET

    # Specify the location of the certificate store. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `CurrentUser` or `LocalMachine`.
    # store_location: NEEDS_TO_BE_SET

    # Specify the type of value the agent uses to find the certificate
    # in the collection of certificates from the certificate store.
    # The `certificate_location` property must be set to `Store`.
    # Value options include `FindByIssuerDistinguishedName`,
    # `FindByIssuerName`, `FindBySerialNumber`,
    # `FindBySubjectDistinguishedName`, `FindBySubjectKeyIdentifier`,
    # `FindBySubjectName`, or `FindByThumbprint`.
    # find_type: NEEDS_TO_BE_SET

    # Specify the value the agent uses in combination with
    # `find_type` to find a certification in the certificate store.
    #
    # Note - The agent will use the first certificate from
    # the certificate store that matches this search criteria.
    #
```

```
    # find_value: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO
```

```
    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
```

```
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked: NOTICE

    # Set the log level of Blocked At Perimeter
    # attacks. Value options are `ALERT`, `CRITICAL`,
    # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked_perimeter: NOTICE

    # Set the log level of Probed attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_probed: WARNING

    # Set the log level of Suspicious attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_suspicious: WARNING

    # Set the connection type used for Syslog messages.
    # Value options are `UNENCRYPTED` and `ENCRYPTED`.
    # connection_type: UNENCRYPTED

  # \
================================================================================
  # agent.dotnet
  # The following properties apply to any .NET agent-wide configurations.
  # \
================================================================================
  # dotnet:

    # Set a list of application pool names that the agent does not
    # instrument or analyze. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_denylist: NEEDS_TO_BE_SET

    # Set a list of application pool names that the agent instruments or
    # analyzes. If set, other application pools are ignored. Allowlist takes
    # precedence over denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_allowlist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent does not
    # analyze. (The applications are still instrumented).
    # Names must be formatted as a comma-separated list.
    # New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_denylist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent analyzes.
    # If set, other applications are not analyzed, but are
    # still instrumented. Allowlist takes precedence over
    # denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_allowlist: NEEDS_TO_BE_SET

    # Enable a profiler chaining feature to allow Contrast to
    # work alongside other tools that use the CLR Profiling
    # API. Defaults to `true`. New after .NET Framework 19.1.3
```

```
# (Installed Only) and .NET Core 1.9.3 (Installed Only).
# enable_chaining: true

# Indicate that the agent should produce a report that
# summarizes application hosting on the server (e.g.,
# CLR versions, bitness or pipeline modes). Defaults to
# `true`. New after .NET Framework 19.1.3 (Installed Only).
# enable_dvnr: true

# Indicate that the agent should allow CLR optimizations
# of JIT-compiled methods. Defaults to `true`. New
# after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_instrumentation_optimizations: true

# Indicate that the agent should allow the CLR to inline
# methods that are not instrumented by Contrast. Defaults to
# `true`. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_jit_inlining: true

# Indicate that the agent should allow the CLR to perform
# transparency checks under full trust. Defaults to `false`.
# New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_transparency_checks: false

# Responses for request paths (e.g., HttpRequest.Path)
# that match this regex are not analyzed. Defaults to
# `WebResource.axd`. New after .NET Framework 19.1.3.
# web_module_allowlist: WebResource.axd

# Set to display ASCII art to std::out on agent startup. Defaults
# to `true`. New after .NET Framework 20.6.3 and .NET Core 1.0.0.
# enable_cat: true

# Sets the maximum amount of time a Protect regular expression
# is allowed to run before being cancelled. Set to -1 to never
# cancel regular expression execution. Defaults to `20_000`.
# New after .NET Framework 20.4.3 and .NET Core 1.5.0.
# protect_searchers_single_pattern_deadline_ms: 20_000

# Sets the maximum amount of time a 'Probe Analysis' Protect
# regular expression is allowed to run before being cancelled. Set
# to -1 to never cancel regular expression execution. Defaults to
# `5_000`. New after .NET Framework 20.7.3 and .NET Core 1.5.11.
# protect_searchers_probe_analysis_single_pattern_deadline_ms: 5_000

# Sets the maximum amount of time a Protect rule is
# allowed to run before being cancelled. Set to -1 to never
# cancel Protect rule execution. Defaults to `60_000`.
# New after .NET Framework 20.4.3 and .NET Core 1.5.0.
# protect_searchers_total_rule_deadline_ms: 60_000

# Sets the maximum amount of time a 'Probe Analysis' Protect
# rule is allowed to run before being cancelled. Set to -1 to
# never cancel Protect rule execution. Defaults to `10_000`.
# New after .NET Framework 20.7.3 and .NET Core 1.5.11.
```

```
    # protect_searchers_probe_analysis_total_rule_deadline_ms: 10_000

    # Sets the maximum duration of time agent log files should be kept
    # since last write before being deleted by the agent. Defaults to
    # `604_800_000`. New after .NET Framework 20.6.1 and .NET Core 1.5.5.
    # log_cleanup_maximum_age_ms: 604_800_000

    # Suppresses gathering process-level metrics (process level metrics are
    # gathered by default), used to identify performance problems. Metric
    # counters may further decrease the stability of already unstable
    # systems and can be disabled (set to true) if issues occur. Defaults
    # to `false`. New after .NET Framework 20.6.6 and .NET Core 1.5.10.
    # suppress_metric_counters: false

    # Enable file based application watching. Set to false if
    # file watching is causing locking issues. Defaults to `true`.
    # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # enable_file_based_app_watching: true

    # Enables HttpClient isolation using AppDomain remoting. This can be \
used
    # to workaround .NET TLS version limitations at the cost of performance
    # and stability. Enabled by default on applications targeting .NET
    # Framework < 4.7.0, else disabled. New after .NET Framework 21.5.1.
    # enable_http_client_app_domain_isolation: false

    # Enables LINQ optimizations to improve performance
    # at the cost of possible false negatives. Defaults
    # to `true`. New after .NET Framework 50.0.1.
    # enable_linq_optimizations: true

    # \
================================================================================
    # agent.dotnet.file_analysis_time_ms
    # Controls the interval in milliseconds to perform file
    # analysis for supported rules. Setting a value > 0 will
    # result in the job running at that interval and not just when
    # the application loads. If set to `-1`, the job just runs
    # once. Defaults to `-1`. New after .NET Framework 50.0.15.
    # \
================================================================================
    # file_analysis_time_ms: {}

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true
```

```
  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Specifies the cloud provider from which the agent should gather metadata
  # (such as resource identifiers). Options are `AWS`, `Azure`, or `GCP`.
  #
  # gather_metadata_via: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
```

```
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET
```

```
    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-chaining
    # Use the following properties to configure how the
    # sql injection semantic analysis chaining rule works.
    # \
================================================================================
    # sql-injection-semantic-chaining:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-dangerous-functions
    # Use the following properties to configure how the sql
    # injection semantic analysis dangerous functions rule works.
    # \
================================================================================
    # sql-injection-semantic-dangerous-functions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
```

```
================================================================================
    # protect.rules.sql-injection-semantic-suspicious-unions
    # Use the following properties to configure how the sql
    # injection semantic analysis suspicious unions rule works.
    # \
================================================================================
    # sql-injection-semantic-suspicious-unions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-tautologies
    # Use the following properties to configure how the sql
    # injection semantic analysis tautologies rule works.
    # \
================================================================================
    # sql-injection-semantic-tautologies:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:
```

```
      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
    # \
================================================================================
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
```

```
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
================================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off
```

```
    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET
```

```
  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost
```

```
  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Override the reported server path. New after
  # .NET Framework v21.3.1 and .NET Core v1.8.0.
  # path: NEEDS_TO_BE_SET
```

証明書の例外について

証明書の例外が発生していて、無視しても問題ないと思われる場合は、YAML 設定ファイルに次の設定を追加します。

```
api:
  certificate:
    ignore_cert_errors: true
```

サポートポータルの記事にて、証明書の問題に対処するための詳細を参照できます。

IIS Express で.NET Framework エージェントを使用 する

.NET エージェントは、IIS Express でホストされる ASP.NET アプリケーションを解析できますが、IIS Express でエージェントの組み込みを有効にするには、作業が少々必要です。IIS Express がサーバにインストールされている場合は、Contrast トレイに IIS Express のタブが表示されます。

Contrast トレイ (220ページ)には、IIS Express ホストのアプリケーションでエージェントの組み込みを有効にするための **Set Environment Variables**(環境変数を設定)ボタンが最初に表示されます。このボタンを選択すると、現在のユーザの環境変数が設定され、新しい IIS Express プロセスによって.NET エージェントのプロファイラがロードされ、エージェントの組み込みと解析が行われるようになります。

環境変数を設定すると、トレイに **Remove Environment Variables**(環境変数を削除)ボタンが表示されます。このボタンを使用すると IIS Express ホストのアプリケーションで Contrast の解析を無効にできます。

IIS Express で実行中のアプリケーションでエージェントが組み込まれたものがあれば、観測された URL(`querystring` なし)の件数とともに IIS Express タブに表示されます。

注記
IIS Express のプロセスインスタンスは、通常、Visual Studio やコマンド画面などの他のプログラムから起動されます。上記のユーザ環境変数を設定した後は、これらのプログラムを再起動してください。ユーザ環境変数を設定する前に実行されていたプログラム(Visual Studio など)は、環境変数なしで IIS Express を起動していることになりますので、その IIS Express でホストされるアプリケーションにはエージェントが組み込まれず解析されません。

また、ユーザ環境変数を設定すると、そのユーザが起動する全ての.NET アプリケーションで Contrast プロファイラがロードされることになります。Contrast プロファイラは、IIS 以外および IIS Express 以外のプロセスから安全にデタッチされます。Windows では、プロファイラ DLL のデタッチが Windows イベントログでエラーメッセージとして処理されますが、このエラーは無視して構いません。

Azure のアプリケーションで.NET Framework エージェントを使用 する

Contrast .NET Framework エージェントを使用して、Azure Virtual Machines (VM)、Azure Cloud Services、Azure Mobile Services、Azure App Service (旧 Azure Web Sites)で実行される ASP.NET アプリケーションを解析できます。

Azure Virtual Machines で.NET Framework エージェントをイントールするには：

1. 通常どおり Azure VM または Azure Cloud Services をセットアップし、検査対象の ASP.NET アプリケーションをデプロイします。
2. Contrast Web インターフェイスにログインして、.NET Framework エージェントの ZIP ファイルをダウンロードします。
3. Azure VM または Azure Cloud Services インスタンスにリモートデスクトップ接続します。
4. .NET Framework エージェントの ZIP ファイルを、Azure VM または Azure Cloud Services インスタンスにコピーし、アーカイブを解凍します。
5. .NET Framework エージェントのインストーラ(*ContrastSetup.exe*)を実行します。
6. アプリケーションを疎通すると、Contrast で解析が行われます。

ヒント
Azure App Service(旧 Azure Web Apps)でインストールを行う場合は、NuGet (176ページ)か Azure Portal の拡張機能 (172ページ)を使用します。

Azure Service Fabric で.NET Framework または.NET Core エージェントを使用する

コンテナイメージを使用する場合は、コンテナにインストール (174ページ)する手順に従ってください。そうでない場合は、Azure Service Fabric サービスに Contrast .NET Framework または.NET Core エージェントを追加します。

ヒント
スタンドアロンの実行サービスの場合、`ServiceManifest.xml` ファイルは最上位の Azure Service Fabric プロジェクト(例、`sfproj` ファイル)にあります。

1. 適切な NuGet パッケージをサービスのメインプロジェクトにインストールします。
   - **.NET Framework の場合：** `Contrast.NET.Azure.AppService` をインストールします。`contrastsecurity` フォルダにあるすべてのファイルは、プロパティがに設定されている必要があります。
   - **.NET Core の場合：** `Contrast.SensorsNetCore` をインストールします。`contrast` フォルダにあるすべてのファイルは、プロパティがに設定されている必要があります。
2. `ServiceManifest.xml` の `ServiceManifest/CodePackage/EntryPoint/ExeHost/WorkingDirectory` を `CodePackage` に設定します。

```
<CodePackage Name="Code" Version="1.0.0">
          <EntryPoint>
          <ExeHost>
            <Program>DemoNetFxStatelessService.exe</Program>
            <WorkingFolder>CodePackage</WorkingFolder>
```

3. `ServiceManifest.xml` で環境変数を定義して、プロファイラを指定します。
   - **.NET Framework の場合：**

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="COR_ENABLE_PROFILING" \
Value="1"/>
            <EnvironmentVariable Name="COR_PROFILER" \
Value="{EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}"/>
            <EnvironmentVariable Name="COR_PROFILER_PATH_32" \
Value=".\contrastsecurity\runtimes\win-
x86\native\ContrastProfiler.dll" />
            <EnvironmentVariable Name="COR_PROFILER_PATH_64" \
Value=".\contrastsecurity\runtimes\win-
x64\native\ContrastProfiler.dll" />
            <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

   - **.NET Core の場合：**

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="CORECLR_ENABLE_PROFILING" \
Value="1"/>
            <EnvironmentVariable Name="CORECLR_PROFILER" \
Value="{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}"/>
            <EnvironmentVariable Name="CORECLR_PROFILER_PATH_32" \
Value="contrast\runtimes\win-x86\native\ContrastProfiler.dll"/>
            <EnvironmentVariable Name="CORECLR_PROFILER_PATH_64" \
Value="contrast\runtimes\win-x64\native\ContrastProfiler.dll"/>
            <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

4. 次のいずれかで、エージェントを設定します。
   - **YAML 設定ファイル：**このファイルをサービスのメインプロジェクトに追加します。ファイルのプロパティが、に設定されていることを確認してください。 以下のように、ファイルの場所を指定する環境変数を `ServiceManifest.xml` に追加します。

```
<CodePackage>
              <EnvironmentVariables>
                <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

   - **環境変数：** 以下のように、環境変数を `ServiceManifest.xml` に追加します。

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="CONTRAST__API__URL" \
Value="https://teamserver-staging.contsec.com"/>
            <EnvironmentVariable Name="CONTRAST__API__API_KEY" \
Value="aBcD0123"/>
            <EnvironmentVariable Name="CONTRAST__API__SERVICE_KEY" \
Value="ABCD0123"/>
            <EnvironmentVariable Name="CONTRAST__API__USER_NAME" \
Value="agent_123@Team"/>
```

5. 通常どおり、Azure Service Fabric アプリケーションをデプロイします。

.NET Framework エージェントのプロファイラチェーン

プロファイラチェーンを使用すると、パフォーマンスツールや APM ツールなどの、.NET プロファイラエージェントと併せて.NET Framework エージェントを実行することができます。

Contrast .NET Framework エージェントは、以下のプロファイリングツールとの互換性をテストおよび実証済みです。

プロファイリングツール	検証済バージョン
AppDynamics	4.5.18.1
Dynatrace OneAgent	1.253.245
New Relic	8.23.107
リバーベッド SteelCentral Aternity APM	12.9.0
Datadog	2.35.0

注記

その他のプロファイリングツールでも、CLR のプロファイル API の規則に準拠しており、プロファイリング環境に関する前提条件がない場合は、エージェントはそのツールと互換性がある可能性があります。

プロファイラチェーンは、デフォルトでは有効になっています。プロファイラチェーンを無効にするには、`agent.dotnet.enable_chaining` オプションを `false` にするよう、.NET Framework エージェントを設定 (181ページ)します。例えば、次のように YAML 設定を使用して指定できます。

```
agent:
  dotnet:
    enable_chaining: false
```

設定が完了したら、エージェントを再起動する必要があります。再起動すると、Contrast .NET Framework エージェントは、IIS に登録されている他のプロファイリングツールの存在を自動的に検出し、Contrast .NET Framework エージェントとサードパーティの両方のプロファイラをロードするよう環境を構成します。

重要
プロファイラチェーンの使用方法に応じて、以下の各手順に従ってください。

- **IIS：**
  サードパーティのエージェントをインストールしてから、Contrast .NET Framework エージェントをインストールしてください。
  (サードパーティのエージェントの前に Contrast .NET Framework エージェントをインストールした場合は、**Windows サービス**で Contrast.NET Main Service を再起動する必要があります。)
- **IIS 以外：**
  以下のようなアプリケーションでプロファイラチェーンを使用している場合、`agent.dotnet.enable_chaining` オプションは機能しません。
  - IIS 以外でホストされているアプリケーション
  - (インストールされたエージェントではなく)サードパーティエージェントの Nuget パッケージを使用するアプリケーション

  上記に該当する場合は、プロファイリングツールの CLR 環境変数を CONTRAST_CCC_COR バージョンに置き換えてください。以下の環境変数名はいずれも置き換える必要があります。

変更すべき変数名	変更後
COR_PROFILER	CONTRAST_CCC_COR_PROFILER
COR_PROFILER_PATH	CONTRAST_CCC_COR_PROFILER_PATH
COR_PROFILER_PATH_32	CONTRAST_CCC_COR_PROFILER_PATH_32
COR_PROFILER_PATH_64	CONTRAST_CCC_COR_PROFILER_PATH_64

  次に、ご利用の環境とアプリケーションに合わせて、通常のセットアップ手順を行ってください。
- **Azure App Service の Application Insights：**
  バージョン 20.9.3 より、Contrast .NET Framework Site Extension は、Application Insights との互換性をサポートするようになりました(CLR Instrumentation Engine (CIE)を使用)。
  Contrast.NET Framework Site Extension で、Application Insights を使用するために追加の操作は必要ありません。.NET Framework エージェントのプロファイラは、(Application Insights が有効になっているなどの理由で)Azure AppService インスタンスでプロファイリングツールとして登録されている場合、CIE によってロードされます。

.NET エージェントエクスプローラ

重要
.NET Core エージェント 4.0.0 および.NET Framework エージェント 51.0.0 より、Contrast トレイアプリケーションを.NET エージェントエクスプローラに置き換えました。

.NET エージェントエクスプローラは、.NET Core エージェントおよび.NET Framework エージェントの稼働状況に関する概要情報を表示するアプリケーションです。このアプリケーションを使用すると、エ

ージェントが想定通りに動作しているかを確認できます。特にエージェントを最初にインストールした後にご利用ください。

エージェントをインストールすると、このアプリケーションもインストールされます。両方の種類のエージェントをインストールした場合は、エージェントエクスプローラのインスタンスは 1 つだけインストールされます。

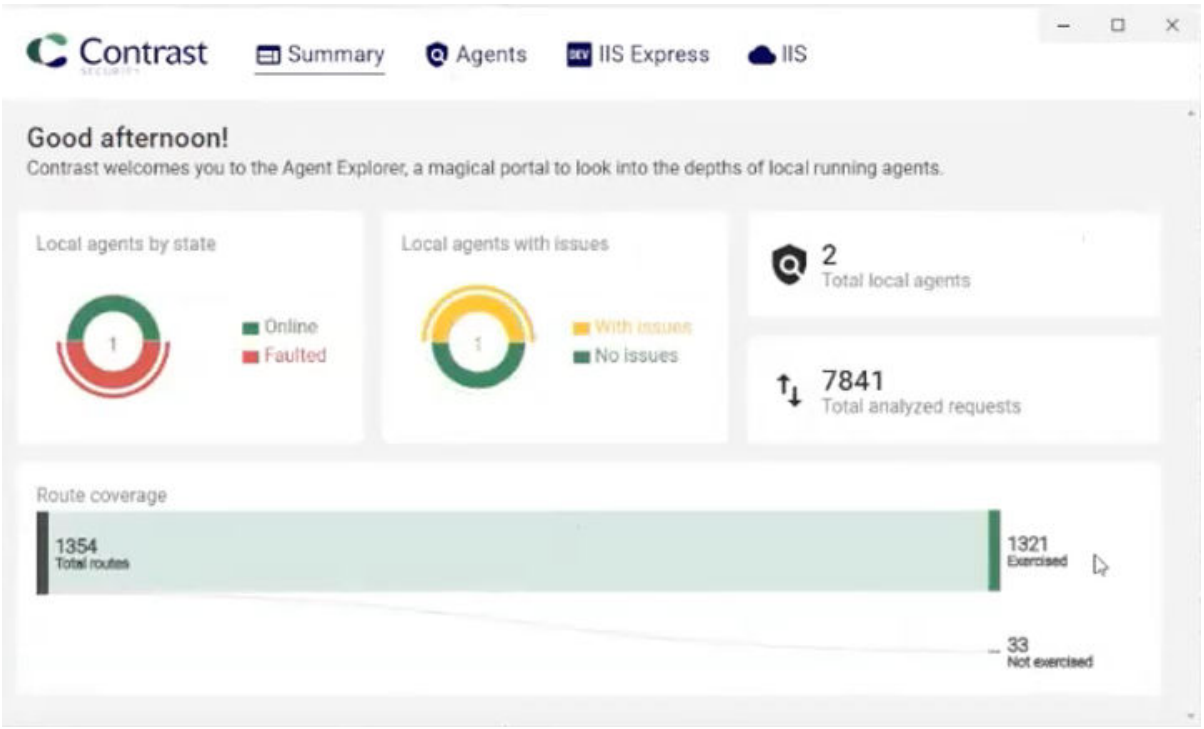
エージェントエクスプローラへのアクセス

.NET Core エージェントか.NET Framework エージェントをインストールした後、エージェントエクスプローラのアイコン( )がトレイに表示されます。アイコンを右クリックすると、アプリケーションが開きます。

エージェントエクスプローラでの情報

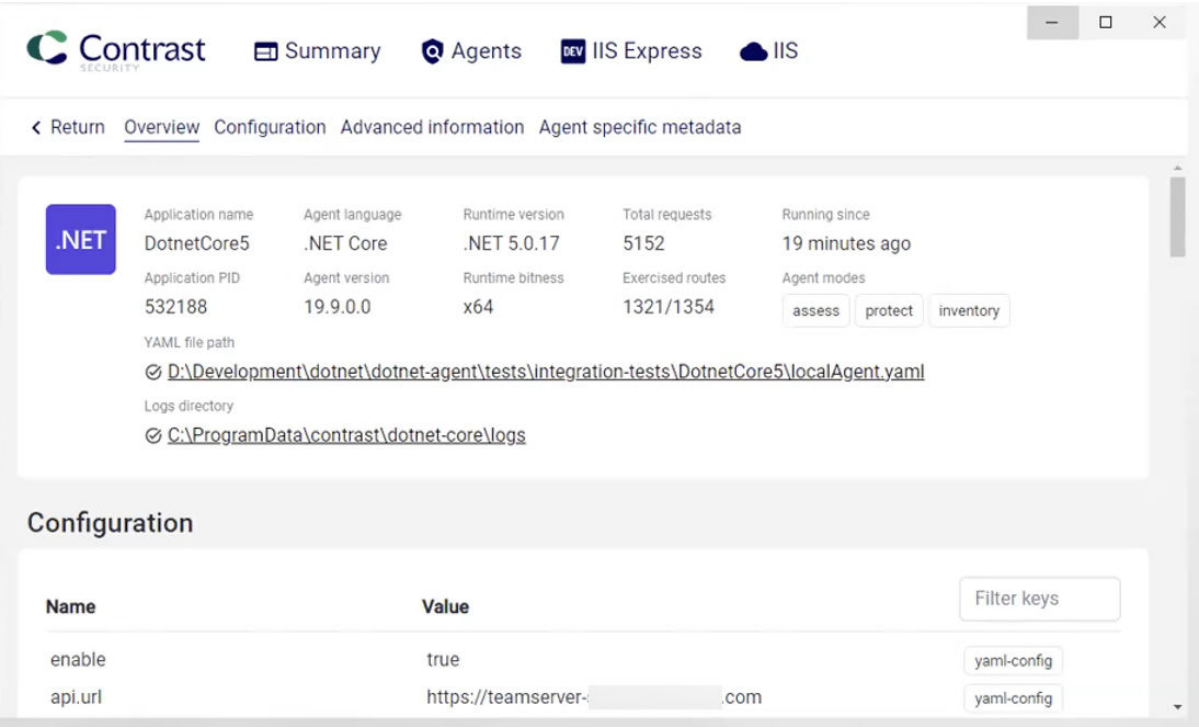
エージェントエクスプローラには、以下の情報が表示されます。

- **Summary(サマリー)**



このダッシュボードには、エージェントのステージ、問題のあるエージェントの有無、ルートカバレッジなど、エージェントに関する概要レベルの情報が表示されます。

- **Agents(エージェント)**



このタブには、.NET Core エージェントおよび.NET Framework エージェントの稼働状況に関する情報が表示されます。Configuration(設定)セクションには、特定の問題が発生していることをエージェントエクスプローラが検知した場合に、メッセージが表示されます。
Overview(概要)セクションにあるリンクから、エージェントの設定ファイル(YAML ファイル)に直接アクセスすることができます。下にスクロールすると、エージェントの設定内容、詳細情報、セッションメタデータの情報などが表示されます。

- **IIS Express**
  このタブには、IIS Express で実行されている Web アプリケーションの情報が表示されます。
- **IIS**
  このタブには、IIS サーバで実行されている Web アプリケーションの情報が表示されます。

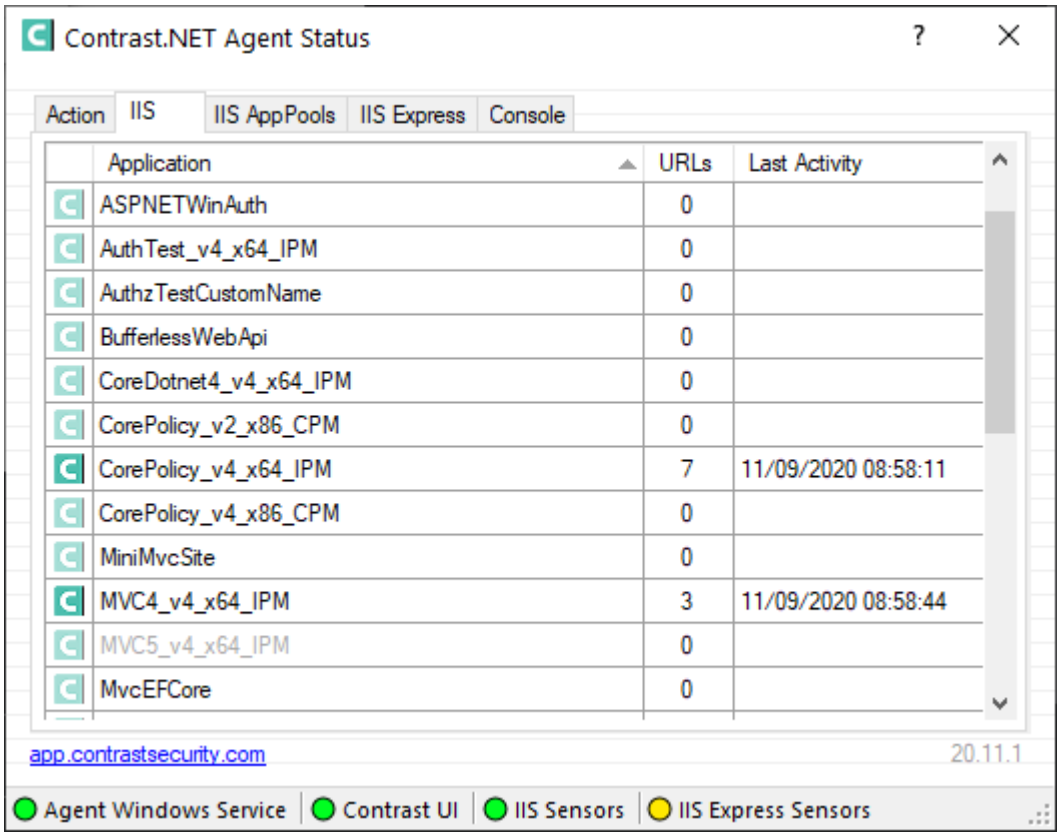
.NET Framework Contrast トレイ

Contrast トレイをサポートするエージェントの最後のバージョンは、50.0.0 です。それより後のエージェントのバージョンでは、Contrast エージェントエクスプローラ (218ページ)を使用します。

.NET Framework Contrast トレイは、Windows システムトレイアプリケーション(*ContrastTray.exe*)で、エージェントの状態に関する情報を概要レベルで表示します。

注記

Contrast を使用してアプリケーションを解析するために、.NET Framework Contrast トレイを実行する必要はありません。Contrast トレイは、エージェントのステータス情報を提供するためだけのものです。特に最初にエージェントをインストールした時などに、エージェントが正しく動作しているかを確認できます。



Contrast トレイには以下のステータスインジケータがあります。

- **Agent Windows Service**(エージェントの Windows サービス) : Contrast Service が正しくインストールされて実行されていると、緑色に点灯します。
- **Contrast UI** : エージェントの Windows サービスが Contrast サーバと通信できると、緑色に点灯します。通信に失敗する最も一般的なエラーは、プロキシの設定の誤りです。
- **IIS Sensors**(IIS センサー) : IIS でホストされているアプリケーションにエージェントが正しく組み込まれると、緑色に点灯します。黄色に点灯する場合は、アプリケーションにエージェントは組み込まれていますが、IIS でまだそのアプリケーションがロードされていないことを示します。
- **IIS Express Sensors**(IIS Express センサー) : IIS Express でホストされているアプリケーションがロードされ、Contrast エージェントが正しくインストールされて実行されると、緑色に点灯します。黄色に点灯する場合は、アプリケーションに Contrast エージェントは正しくインストールされていますが、IIS Express でまだそのアプリケーションがロードされていないことを示します。赤色の点灯は、IIS Express で環境変数が設定されていないことを示します。

Contrast トレイの各タブを選択すると、詳細情報が表示されます。

- **Action**(アクション) : .NET Framework エージェントを使用するためのユーザへの指示が概要レベルで表示されます。ここで表示される指示は、エージェントの状態により変わります。例えば、エージェントが Contrast サーバに接続できない場合、Action タブにはエラーの詳細情報と問題を解決する方法が提示されます。
- **IIS** : IIS サーバで実行されている全ての Web アプリケーションが一覧で表示されます。表示されるアプリケーション名は、カスタムのアプリケーション名を指定 (181ページ)しない限り、IIS がアプリケーションを識別するために使用するエイリアスと同じものになります。URL 列には、エージェントがアプリケーションで観測したユニークな URL (クエリストリングを含まない)の件数が表示されます。Last Activity(最後のアクティビティ)列には、そのアプリケーションに対してエージェントが最後に解析したリクエストの時間が表示されます。
- **IIS AppPools**(IIS アプリケーションプール) : IIS サーバ上の全てのアプリケーションプールが一覧で表示されます。各アプリケーションプールの構成の詳細(アーキテクチャ、パイプラインモード、CLR バージョン、ID)を確認できます。また、このアプリケーションプールのアプリケーションが Contrast

で解析されるかどうかも確認できます。IIS で、.NET Framework エージェントのアプリケーションプールのフィルターを設定 (222ページ)できます。
- **IIS Express** : IIS Express で実行されている全ての Web アプリケーションが一覧で表示されます。
- **Console**(コンソール) : Contrast .NET エージェントの問題のトラブルシュートに役立つ、ステータスやエラーメッセージが表示されます。

IIS でアプリケーションプールを使用する

.NET エージェントは、IIS にデプロイされている全ての ASP.NET アプリケーションに自動的に組み込まれます。.NET エージェントがインストールされ、エージェントのバックグラウンドの Windows サービスが実行されていることが確認されると、IIS でホストされる全てのアプリケーションに対してエージェントが組み込まれます。

以下のような理由などにより、一部のアプリケーションでエージェントの組み込みを対象外にすることもできます。

- いくつかのアプリケーションは、セキュリティやライブラリ情報を収集する必要がない
- アプリケーションがあるサーバにリソースの制約があるか、あるいは Contrast エージェントを組み込むために追加のパフォーマンス要件は避けたい

IIS でホストされる Web アプリケーションは、アプリケーションプールで実行されます。アプリケーションの.NET エージェントを無効にする必要がある場合、そのアプリケーションが実行されるアプリケーションプールを拒否 (222ページ)します。

特定のアプリケーションが実行されるアプリケーションプールを検索する方法は 3 つあります。

- **インターネット インフォメーション サービス (IIS) マネージャー**
  `%windir%\system32\inetsrv\InetMgr.exe` コマンドを使用して、IIS マネージャーを起動します。対象の Web アプリケーションを選択し、**基本設定**を選択します。アプリケーションプール名を表すフィールドが表示されます。
- **`AppCmd.exe`**
  管理者権限がある場合は、`cmd.exe` を実行します。*C:\Windows\System32\inetsrv* に移動します。`appcmd list apps` と入力すると、アプリケーションの一覧ととそれぞれのアプリケーションプールが表示されます。
- **Contrast .NET ログ**
  Contrast .NET エージェントを起動します。アプリケーションを操作します。そして ログディレクトリに移動します。Windows の場合、*C:\ProgramData\Contrast\dotnet\LOGS* に移動します。最新のプロファイルログ*(XXXXX_Profiler_[AppDomain]XXXXX[XX].log)*を開きます。アプリケーションプール名は、**ApplicationPool Name** で始まる行にあります。

アプリケーションプールの拒否と許可リスト

拒否リストと許可リストは、アプリケーションプール名に基づいて設定されます。アプリケーションプールの拒否リストと許可リストには、*もワイルドカード文字として使用できます(`AppPool*`は、`AppPool1` や `AppPool_arb` などと一致します)。

許可リストは、拒否リストよりも優先されます。両方のリストを満たすアプリケーションプールは、解析されません。

特定のアプリケーションのエージェントを無効にするには、*C:\ProgramData\Contrast\dotnet\contrast_security.yaml* で、`agent.dotnet.app_pool_denylist` オプションに該当のアプリケーションプールを指定します。

```
# Comma-separated list of application pools ignored by Contrast
agent:
  dotnet:
    app_pool_denylist: ExampleAppPoolName
```

IIS でホストされるアプリケーションで特定のアプリケーションにのみエージェントを有効にするには、特定のアプリケーションプールのみを解析するよう `agent.dotnet.app_pool_allowlist` を設

定します。アプリケーションプールが許可リストに指定されている場合、エージェントは一致するプールを解析します。他のアプリケーションへの影響はありません。

特定のアプリケーションプールにのみエージェントを有効にするには、*C:\ProgramData\Contrast\dotnet\contrast_security.yaml* で、`agent.dotnet.app_pool_allowlist` オプションに該当のアプリケーションプールを指定します。

```
# Comma-separated list of application pools exclusively profiled by Contrast
agent:
  dotnet:
    app_pool_allowlist: ExampleAppPoolName
```

ヒント
詳細については、YAML 設定 (63ページ)を参照してください。

.NET Framework および.NET Core のテレメトリ

.NET Framework エージェントおよび.NET Core エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集され、その後も定期的(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切 (964ページ)であると考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、安全に Contrast に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	データ	
.NET Framework 2020.8.3 以降	エージェントのバージョン	
.NET Core 1.5.15 以降	オペレーティングシステムとバージョン	
	エージェントがコンテナ内で実行されているかどうか	
	エージェントが Azure App Services で実行されているかどうか	
	ハッシュ化されたメディアアクセス制御(MAC)アドレス：暗号化(SHA256)された、匿名の一意なマシン ID	
	カーネルのバージョン	
	プロセスの実行時間	
	Assess が有効であるかどうか	
	Protect が有効であるかどうか	
	.NET Framework 2020.8.3 以降	.NET Framework ランタイムのバージョン
	.NET Core 1.5.15 以降	.NET Core ランタイムのバージョン
.NET Framework 20.9.1 以降 .NET Core 1.5.17 以降	Contrast インスタンスが SaaS 版かオンプレミス版であるか	
.NET Framework 20.9.3 以降	CLR Instrumentation Engine (CIE)の使用状況	
.NET Core 1.5.19 以降	アプリケーションのフレームワーク	

エージェントのバージョン	データ
	連携しているプロファイリングツール
.NET Framework 20.10.1 以降	プロセスのホスティングモード
.NET Core 1.5.20 以降	CIE ネイティブプロファイラフックの使用状況
.NET Framework 20.10.2 以降	デフォルト以外の値が指定されている設定名
.NET Core 1.5.21 以降	無効にされている Assess ルール名
.NET Framework 20.12.2 以降 .NET Core 1.7.2 以降	エージェントのプロファイラコンポーネントが初期化されるまでに経過した時間
	エージェントから Contrast Web インターフェイスへの最初のリクエストまでに経過した時間
	エージェントのプロファイラコンポーネントが初期化されるまでに経過した時間
	エージェントの初期化から最初のリクエストが終了するまでに経過した時間
.NET Framework 21.1.1 以降	IIS でホストされているアプリケーションに関する測定値： • 合計アプリケーション数 • 解析対象となるアプリケーション数(アプリケーションの許可/拒否リストで設定) • CLR4 のアプリケーションプールでホストされているアプリケーション数 • CLR2 のアプリケーションプールでホストされているアプリケーション数
	IIS のアプリケーションプールに関する測定値： • 合計数 • エージェントが接続されている数 • CLR4 の数 • CLR2 の数
	単独のアプリケーションプールでのアプリケーションの最小数
	単独のアプリケーションプールでのアプリケーションの最大数
	全てのアプリケーションプールにあるアプリケーション数の中央値
.NET Framework 21.1.2 以降 .NET Core 1.7.5 以降	Protect の各ルールのモード(例、監視やブロック)
.NET Framework 21.4.2 以降 .NET Core 1.8.4 以降	エージェントのセンサーコード内でスローされキャッチされた例外。ログメッセージ、例外タイプ、例外のメッセージ、`System` メソッドおよび `Contrast` メソッドのスタックトレースフレームなど。
.NET Framework21.7.1 以降	• プロセスアーキテクチャ(x86/x64)
.NET Core1.9.7 以降	OS アーキテクチャ(x86/x64)
	Azure App Service における以下の環境変数の値： • `WEBSITE_PHYSICAL_MEMORY_MB` • `WEBSITE_PLATFORM_VERSION` • `WEBSITE_SKU`
.NET Framework21.9.2 以降 .NET Core2.0.1 以降	YAML 設定ファイルを読み込んだ場所(環境変数で指定したパス、デフォルトの場所、アプリケーションディレクトリなど)に関する説明

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、*telemetry.dotnet.contrastsecurity.com* に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

.NET Core エージェント

Contrast .NET エージェントは、ユーザが.NET Core の Web アプリケーションを操作する際のアプリケーションの動きを解析します。

ホストプロセスにプロファイリング用の環境変数やアプリケーション起動プロファイルが設定されると、エージェントは自動的に ASP.NET Core アプリケーションに組み込まれます。

Contrast .NET Core エージェントは 2 つのコンポーネントで構成され、どちらもアプリケーションと同じプロセス内で動作します。

- **.NET プロファイラ**は、アプリケーションによって使用されるセキュリティ関連の API とその依存関係に Contrast センサーコードへの呼び出しを追加することで、アプリケーションに組み込まれます (IL Weaving とも呼ばれます)。
- **センサー**は、セキュリティ情報、アーキテクチャ情報、ライブラリ情報を収集します。

.NET Core エージェントをインストール (227ページ)したら、ユーザがアプリケーションを実行する際に、エージェントのセンサーがアプリケーションのセキュリティ、アーキテクチャ、ライブラリに関する情報を収集します。エージェントによる解析結果は、Contrast Web インターフェイスで確認できます。エージェントには、こちらのサポート対象テクノロジ (225ページ)とシステム要件 (226ページ)で動作します。

.NET Core のサポート対象テクノロジ

このエージェントでは、以下のテクノロジをサポートします。

テクノロジ	サポート対象バージョン	備考
アプリケーションフレームワーク	• ASP.NET Core(3.1.x、5.0.x、6.0.x、7.0.x、8.0.x) • Model-View-Controller (MVC) • Razor Pages	**限定サポート：** ASP .NET Core 3.1.x、5.0.x **サポート対象外：** • .NET Core または ASP.NET Core バージョン 2.1 以前 • .NET Framework (Windows)または Mono (Linux/Windows)上で動作する ASP.NET Core アプリケーション

ランタイム	• .NET Core ランタイム：3.1.x、5.0.x、6.0.x、7.0.x 、 8.0.x
	• .NET Core ターゲットフレームワーク ：
	  • netcoreapp3.1
	  • net5.0
	  • net6.0
	  • net7.0
	  • net8.0
		限定サポート：
		.NET Core ランタイム 3.1.x、5.0.x
		サポート対象外：
		• ランタイムよりも上位バージョンの ASP.NET Core アプリケーションでの実行(例、ASP.NET Core 5.0 を参照する.NET Core 3.1 ランタイムのアプリケーションなど)
		• 参照する ASP.NET Core のバージョンとコンパイル時に選択したターゲットのランタイムが一致しない.NET Core アプリケーションでの実行

Windows 版.NET Core		
Windows オペレーティングシステム	• Windows Server (LTSC) (x86, x64)：2012 R2、2016、2019、2022 • Windows Server (SAC) (x64)：1809、1903 • Windows ワークステーション (x86、x64)：7、8/8.1、10	64 ビットシステムでは、エージェントを使用して 32 ビットと 64 ビットの両方の Web アプリケーションを解析できます。 **サポート対象外：** • ARM 版 Windows
サーバコンテナ	Kestrel、IISHttpServer	**サポート対象外：** Http.sys (旧 WebListener)
ホスティングコンテナ	セルフホスト、IIS、IIS Express	
Linux OS 版.NET Core		
Linux オペレーティングシステム	• Ubuntu：18.04 以降(x64、ARM64) • Debian：10 以降(x64、ARM64) • openSUSE：15 以降(x64) • Alpine：3.13 以降(x64、ARM64) • CentOS：7(x64) • CentOS Stream：8(x64) • Red Hat Enterprise Linux：7 以降(x64)	**サポート対象外：** Red Hat Enterprise Linux 6

テクノロジ	サポート対象バージョン	備考
サーバコンテナ	Kestrel	
ホスティングコンテナ	セルフホスト	

重要

- .NET Core 2.2 は、.NET Core エージェントのバージョン 1.5.20 以降ではサポートされません。.NET Core 2.2 を使用している場合、アプリケーションの.NET Core ランタイムをアップグレードするまでは、.NET Core エージェントのバージョン 1.5.20 以前のものを使用してください。
- NET Core エージェントのバージョン 1.9.9 をもって、.NET Core 2.1 のサポートを終了しました。.NET Core 2.1 を使用している場合、アプリケーションの.NET Core ランタイムをアップグレードするまでは、.NET Core エージェントのバージョン 1.9.9 以前のものを使用してください。
- Microsoft は、2022 年 5 月 10 日をもって.NET 5.0 のサポートを終了し、2022 年 12 月 13 日をもって.NET Core 3.1 のサポートを終了しました。.NET 5.0 および.NET Core 3.1 に関する Contrast のサポートは、.NET Core エージェントのバージョン 3.0.0 による限定的なサポートとなります。Contrast の限定サポートでは、サポート対象の言語バージョンで再現可能な問題のみを解決します。お使いのアプリケーションを.NET のサポート対象バージョンにアップグレードすることを強く推奨します。

注記

.NET Core エージェントは、`System.Runtime` および ASPNET Core を参照していないアプリケーションをサポートしません。また、エージェントが依存しているアセンブリをコンパイラがトリムする可能性があるため、エージェントはトリミングされた自己完結型のデプロイと実行可能ファイルもサポートしません。

.NET Core のシステム要件

.NET Core エージェントをインストールする前に、以下の要件を満たしていることを確認してください。

- サーバへの管理者権限があり、そのサーバは Contrast のサポート対象 (225ページ)であること。
- 検査対象のアプリケーションがデプロイされており、その Web アプリケーションのテクノロジは Contrast のサポート対象 (225ページ)であること。
- Web サーバが Contrast とネットワークで接続されていること。
- サーバが最低限の要件(以下に記載)を満たしていること。

要件	推奨	最小	備考
CPU	4 個以上	2	
メモリ	8 GB 以上	4 GB	Assess でエージェントを実行するには、検査対象のアプリケーションのメモリ要件がおよそ 2 倍になります。エージェントがインストールされていない場合に、アプリケーションが使用するメモリは全メモリ容量の半分以下である必要があります。
オペレーティングシステム	• Windows • Linux		macOS には対応していません。

要件	推奨	最小	備考
プロセッサのアーキテクチャ	• 32 ビット x86 プロセッサ • 64 ビット x86 プロセッサ • 64 ビット ARM プロセッサ		ARM プロセッサ上の Windows には対応していません。

.NET Core エージェントのインストール

.NET Core エージェントをインストールするには：

1. エージェントのコンポーネントをサーバのファイルシステムに置きます。
2. .NET ランタイムがエージェントのプロファイラコンポーネントをロードするように環境変数を設定します。
3. 通常通りにアプリケーションを疎通し、Contrast でアプリケーションが認識されていることを確認します。

利用する環境に合わせて、以下のいずれかのインストール方法を使用してください。

- 手動インストール (227ページ) (Windows、Linux または Docker で実行するセルフホスト Web アプリケーションを使用する場合)
- IIS 用.NET Core エージェントのインストーラ (234ページ) (IIS を使用している場合)
- Azure App Service (232ページ)
- NuGet (231ページ)

エージェントを自動アップグレードするには、エージェントアップグレードサービス (179ページ)のオプションを有効にします。

.NET Core エージェントを手動でインストール

IIS でホストされている Web アプリケーションを使用している、もしくは、Windows、Linux、Docker 上でセルフホストの Web アプリケーションを実行している場合は、この方法で.NET Core エージェントをインストールします。

注記

コンテナへのインストールは複雑になる可能性があり、本項での手順がお客様の利用状況に合わない可能性もあります。Docker でのインストールについては、こちらの記事を参照してください。

開始する前に

システム要件 (226ページ)とサポート対象テクノロジ (225ページ)を確認し、インストールが可能で最適なパフォーマンスを得られる環境であるかを確認してください。

手順

1. Contrast Web インターフェイスの右上で**新規登録**を選択します。**アプリケーション**のカードを選択します。 **.NET Core** を選択し、 .NET Core エージェントをダウンロードするリンクを選択します。
2. Web サーバ上のディレクトリで、ダウンロードした ZIP アーカイブ(例、*Contrast.NET.Core_1.0.1.zip*)を解凍します。ディレクトリには、アプリケーションがアクセスするのに十分な権限を付与してください。
3. アプリケーションのプロセスに次の環境変数を設定します。ご利用のオペレーティングシステムに適した CORECLR_PROFILER_PATH の設定を指定します。<UnzippedDirectoryRoot>をアーカイブディレクトリに置き換えます。

- **Windows**

環境変数	値
CORECLR_PROFILER_PATH_64	`<UnzippedDirectoryRoot>\runtimes\win-x64\native\ContrastProfiler.dll`
CORECLR_PROFILER_PATH_32	`<UnzippedDirectoryRoot>\runtimes\win-x86\native\ContrastProfiler.dll`
CORECLR_PROFILER	`{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}`
CORECLR_ENABLE_PROFILING	1
CONTRAST_CONFIG_PATH	`<path_to_contrast_security.yaml>`

重要
同じサーバで、.NET Core エージェントと.NET Framework エージェントを実行している場合、CONTRAST_CONFIG_PATH オプションが両方のエージェントのロードパス (62ページ)適用されます。各エージェントに対して個別のパスを適用するには、次のオプションを使用してデータディレクトリを指定します。

- CONTRAST_CORECLR_DATA_DIRECTORY
- CONTRAST_DATA_DIRECTORY

- **Linux x64**

環境変数	値
CORECLR_PROFILER_PATH_64	`<UnzippedDirectoryRoot>/runtimes/linux-x64/native/ContrastProfiler.so`
CORECLR_PROFILER	`{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}`
CORECLR_ENABLE_PROFILING	1
CONTRAST_CONFIG_PATH	`<path_to_contrast_security.yaml>`

- **Linux ARM64**

環境変数	値
CORECLR_PROFILER_PATH_64	`<UnzippedDirectoryRoot>/runtimes/linux-arm64/native/ContrastProfiler.so`
CORECLR_PROFILER	`{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}`
CORECLR_ENABLE_PROFILING	1
CONTRAST_CONFIG_PATH	`<path_to_contrast_security.yaml>`

4. アプリケーションのランタイムユーザが以下のパスにアクセスできることを確認してください。

パス	用途	カスタマイズ可能	権限
.NET Core エージェントの YAML (248ページ)へのパス	エージェントを設定	可、環境変数 CONTRAST_CONFIG_PATH を設定	読取り
<UnzippedDirectoryRoot>	「インストール」のルートディレクトリ、エージェントバイナリを格納	不可	読取り
• **Windows:** *%ProgramData%\Contrast\dotnet-core\logs* • **Linux:** */var/tmp/contrast/dotnet-core/logs*	Contrast エージェントログのディレクトリ、ない場合は生成される	可、環境変数 CONTRAST_CORECLR_LOGS_DIRECTORY を設定	読取り/書込み (または親ディレクトリから継承)

> **注記**
> IIS で実行する場合、アプリケーションプールがこれらのパスにアクセスできることを確認してください。
>
> 例えば、デフォルトの ID である `ApplicationPoolIdentity` を使用する `Default Web Site` という名前のアプリケーションプールがある場合、ユーザの `IIS AppPool\Default Web Site` に、解凍されたディレクトリのルートへの有効な読み取り権限があることを確認します。

5. Contrast サーバに接続するための認証情報やプロキシ情報をエージェントに設定 (247ページ)します。
6. アプリケーションがロードされたら、アプリケーションを疎通して、Contrast でサーバとアプリケーションがアクティブになり、脆弱性が報告されることを確認します。

> **ヒント**
> エージェントを更新 (245ページ)するには、エージェントディレクトリ内のエージェントファイルを置き換え、アプリケーションを再起動します。エージェントはアプリケーションと一緒に実行されているため、エージェント自体を更新することはできません。
>
> 環境が適切に設定されていれば、エージェントはアプリケーションと一緒に自動的に起動します。
>
> エージェントを停止するには、アプリケーションを停止し、環境からエージェントを削除します。または、`CORECLR_ENABLE_PROFILING` の設定を `0` に変更することもできます。

環境変数の設定は、以下の例を参考にしてください。

- IIS (229ページ)
- Bash (Linux) (230ページ)
- Powershell または Powershell Core(Windows) (230ページ)
- 起動プロファイル(dotnet.exe) (230ページ)

IIS および IIS Express

以下のいずれかの方法によって、環境変数を設定します：

- アプリケーションの *web.config* の `environmentVariables` セクション

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.webServer>
        <!-- ... -->
        <aspNetCore processPath="dotnet" \
arguments=".\ExampleNetCoreApp.dll" stdoutLogEnabled="false" \
stdoutLogFile=".\logs\stdout">
            <environmentVariables>
                <environmentVariable name="CORECLR_PROFILER_PATH_64" \
value="C:\contrast\dotnetcore\runtimes\win-
x64\native\ContrastProfiler.dll" />
                <environmentVariable name="CORECLR_PROFILER_PATH_32" \
value="C:\contrast\dotnetcore\runtimes\win-
x86\native\ContrastProfiler.dll" />
                <environmentVariable name="CORECLR_ENABLE_PROFILING" \
```

```
value="1" />
                <environmentVariable name="CORECLR_PROFILER" \
value="{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}" />
                <environmentVariable name="CONTRAST_CONFIG_PATH" \
value="C:\contrast\dotnet-core\contrast_security.yaml" />
            </environmentVariables>
        </aspNetCore>
    </system.webServer>
</configuration>
```

- サーバのアプリケーションプール設定

Bash (Linux)

Linux x64：

```
export CORECLR_PROFILER_PATH_64=/usr/local/contrast/runtimes/linux-x64/
native/ContrastProfiler.so
export CORECLR_ENABLE_PROFILING=1
export CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
export CONTRAST_CONFIG_PATH=/etc/contrast/contrast_security.yaml
```

Linux ARM64：

```
export CORECLR_PROFILER_PATH_64=/usr/local/contrast/runtimes/linux-arm64/
native/ContrastProfiler.so
export CORECLR_ENABLE_PROFILING=1
export CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
export CONTRAST_CONFIG_PATH=/etc/contrast/contrast_security.yaml
```

次に、アプリケーションを実行します。

```
dotnet ./MyAppWithContrastAgent.dll
```

Powershell または Powershell Core (Windows)

```
$env:CORECLR_PROFILER_PATH_64 = 'C:\contrast\dotnetcore\runtimes\win-
x64\native\ContrastProfiler.dll'
$env:CORECLR_PROFILER_PATH_32 = 'C:\contrast\dotnetcore\runtimes\win-
x86\native\ContrastProfiler.dll'
$env:CORECLR_ENABLE_PROFILING = '1'
$env:CORECLR_PROFILER = '{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}'
$env:CONTRAST_CONFIG_PATH = 'C:\contrast\dotnet-
core\contrast_security.yaml'
```

次に、アプリケーションを実行します。

```
dotnet .\MyAppWithContrastAgent.dll
```

起動プロファイル(dotnet.exe)

```
{
    "MyAppWithContrastAgent": {
        "environmentVariables": {
            "CORECLR_PROFILER_PATH_64": "C:\\contrast\\dotnetcore\\runtimes\
\win-x64\\native\\ContrastProfiler.dll",
            "CORECLR_PROFILER_PATH_32": "C:\\contrast\\dotnetcore\\runtimes\
\win-x86\\native\\ContrastProfiler.dll",
            "CORECLR_ENABLE_PROFILING": "1",
            "CORECLR_PROFILER": "{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}",
```

```
            "CONTRAST_CONFIG_PATH": "c:\\contrast\\config\\MyApp\
\contrast_security.yaml"
        }
    }
}
```

次に、アプリケーションを実行します。

```
dotnet run --launch-profile MyAppWithContrastAgent
```

NuGet で.NET Core エージェントを手動でインストール

.NET Core エージェントは、NuGet を使用して手動でインストールすることができます。このインストール方法は、Azure App Service の拡張機能 (232ページ)が利用できない場合や、.NET Core エージェントを依存関係に含めたい場合などに便利です。

開始する前に

重要
Contrast エージェントを実行中の Web アプリケーションを再デプロイする際に、*ContrastProfiler.dll* でファイルが使用中というエラーが発生することがあります。これは、エージェントの DLL ファイルが.NET によってロックされており、アプリケーションの実行中には上書きできないために発生します。

手順

1. アプリケーションに Contrast の NuGet パッケージをインストールします。
   dotnet コマンドラインを使用する場合：

   ```
   dotnet add package Contrast.SensorsNetCore
   ```

   Visual Studio を使用する場合：
   - ソリューションエクスプローラーで、アプリケーションのプロジェクトまたは**参照**のどちらかを右クリックし、**NuGet パッケージの管理**を選択します。
   - **Contrast.SensorsNetCore** パッケージを検索して選択し、プロジェクトに追加します。
   - アプリケーションをビルドします。プロジェクトに *contrast* フォルダが表示されることを確認してください。アプリケーションが公開されると、このフォルダはビルドの出力ディレクトリにも表示されます。
2. .NET ランタイムでエージェントのプロファイラコンポーネントがロードされるよう環境変数を設定します。
   Windows：

   ```
   CORECLR_ENABLE_PROFILING: 1
   CORECLR_PROFILER: {8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
   CORECLR_PROFILER_PATH_32: <application directory>\contrast\runtimes\win-
   x86\native\ContrastProfiler.dll
   CORECLR_PROFILER_PATH_64: <application directory>\contrast\runtimes\win-
   x64\native\ContrastProfiler.dll
   ```

   Linux x64：

   ```
   CORECLR_ENABLE_PROFILING: 1
   CORECLR_PROFILER: {8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
   ```

```
CORECLR_PROFILER_PATH: <application directory>/contrast/runtimes/linux-x64/native/ContrastProfiler.so
```

Linux ARM64：

```
CORECLR_ENABLE_PROFILING: 1
CORECLR_PROFILER: {8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
CORECLR_PROFILER_PATH: <application directory>/contrast/runtimes/linux-arm64/native/ContrastProfiler.so
```

3. YAML 設定ファイル (248ページ)または環境変数 (247ページ)を使用して、基本のオプションを設定します。例：

```
CONTRAST_CONFIG_PATH: [Path to yaml config file]
```

最低限、以下の環境変数が必要です。

```
CONTRAST__API__URL: [IF USING ANOTHER SERVER THAN THE DEFAULT: https://app.contrastsecurity.com]
CONTRAST__API__USER_NAME: [REPLACE WITH YOUR AGENT USERNAME]
CONTRAST__API__SERVICE_KEY: [REPLACE WITH YOUR AGENT SERVICE KEY]
CONTRAST__API__API_KEY: [REPLACE WITH YOUR AGENT API KEY]
```

4. 前の手順で指定した設定を使用して、アプリケーションをデプロイします。
5. アプリケーションがロードされたら、アプリケーションを疎通して、Contrast でサーバとアプリケーションがアクティブになり、脆弱性が報告されることを確認します。

Azure App Service で.NET Core エージェントをインストール

インストールを開始する前に、システム要件 (226ページ)とサポート対象テクノロジ (225ページ)を確認し、インストールが可能な環境であり、最適なパフォーマンスを得られることを確認してください。

Azure Portal の拡張機能を使用して、.NET Core エージェントを簡易にインストールするには、以下の手順を行います。

1. Azure アカウントがない場合は、アカウントを作成してください。
2. .NET Core Web アプリケーションを作成し、Azure App Service にデプロイします。
3. アプリケーションを Azure に公開し、Contrast エージェントなしで想定どおりに機能することを確認します。
4. アプリケーションが Windows プランを使用してデプロイされていることを確認します(Linux プランの場合は、拡張機能を利用できません)。

> **注記**
> 拡張機能を利用できない場合は、NuGet を使用して.NET Core エージェントを手動でインストール (231ページ)できます。

5. App Servcie にデプロイ中のアプリケーションを選択します。
6. アプリケーションのメニューより**構成**を選択し、**アプリケーション設定**の画面でエージェントが Contrast サーバに接続するための設定をします。
7. アプリケーションのメニューより**構成**を選択し、**アプリケーション設定**の画面で以下の値を追加し、エージェントが Contrast サーバに接続するための設定をします。

キー	値
CONTRAST__API__USER_NAME	自分のエージェントユーザ名 (61ページ)を指定します。
CONTRAST__API__SERVICE_KEY	自分のエージェントサービスキー (61ページ)を指定します。
CONTRAST__API__API_KEY	自分の API キー (61ページ)を指定します。

キー	値
CONTRAST__API__URL	デフォルトは、*https://app.contrastsecurity.com* です。別の場所でホストされている Contrast サーバを使用する場合は、その URL を指定します。

8. メニュー画面より**拡張機能**を選択します。

9. **+ 追加**を選択します。
10. 拡張機能の選択をクリックし、一覧より **Contrast .NET Core Site Extension for Azure App Service** を選択します。これが、.NET Core アプリケーション用の拡張機能になります。
11. 法律条項に同意して、**OK** ボタンをクリックします。
12. 拡張機能のインストールが完了するまでしばらく待ったら、正しくインストールされていることを確認します。
13. アプリケーションの概要に戻り、アプリケーションを**再起動**します。
14. アプリケーションを疎通し、検査結果が Contrast に報告されることを確認します。

ヒント

アプリケーションの SCM サイト(Kudu)の「Site Extensions」メニューからも Contrast エージェントをインストールできます。

重要

.NET Core エージェントの新しいバージョンが利用可能な場合は、Azure Portal または Kudu のダッシュボードに通知されます。エージェントの更新を行う前にサイトを停止してください。停止しない場合、更新が失敗する可能性があります。

注記
サイトの拡張機能により、以下のような環境変数がいくつか設定されます。

```
CORECLR_ENABLE_PROFILING=1
CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
CORECLR_PROFILER_PATH_32=D:\Home\siteextensions\Contrast.NetCore
.Azure.SiteExtension\ContrastNetCoreAppService\runtimes\win-
x86\native\ContrastProfiler.dll
CORECLR_PROFILER_PATH_64=D:\Home\siteextensions\Contrast.NetCore
.Azure.SiteExtension\ContrastNetCoreAppService\runtimes\win-
x64\native\ContrastProfiler.dll
CONTRAST_INSTALL_DIRECTORY=D:\Home\siteextensions\Contrast.NetCo
re.Azure.SiteExtension\ContrastNetCoreAppService\
MicrosoftInstrumentationEngine_ConfigPath32_ContrastCoreX86Confi
g=D:\Home\siteextensions\Contrast.NetCore.Azure.SiteExtension\Co
ntrastCieCoreClrProfiler-32.config
MicrosoftInstrumentationEngine_ConfigPath64_ContrastCoreX64Confi
g=D:\Home\siteextensions\Contrast.NetCore.Azure.SiteExtension\Co
ntrastCieCoreClrProfiler-64.config
```

CLR Instrumentation Engine (CIE)がアプリケーションに設定されている場合(例えば、Application Insights が有効になっているために)、自動的に Azure で CORECLR_PROFILER*変数が上書きされて、CIE のプロファイラが指定されます。

そして、CIE は MicrosoftInstrumentationEngine_*変数を使用して Contrast エージェントをロードするようになります。

アプリケーションに対して CIE が設定されていない場合は、Contrast エージェントのロードに標準の CORECLR_PROFILER*変数が使用されます。

IIS 用.NET Core エージェントのインストーラで.NET Core エージェントをインストール

IIS 用.NET Core エージェントのインストーラは、標準の MSI を利用して作られた Windows アプリケーションの一般的なインストーラです。インストーラは、対象となるサーバが要件(サーバのオペレーティングシステムがサポート対象であることなど)を満たしているかを検証します。サーバが全ての要件を満たしていれば、インストーラによって以下が行われます。

- IIS 用.NET Core エージェントを Windows の標準プログラムとして登録します。
- 指定された場所(例、`C:\Program Files\Contrast\dotnet-core`)にエージェントのファイルを配置します。これには、いくつかのダイナミックリンクライブラリ(DLL)などが含まれます。
- エージェントのログファイルや設定を主に保存するためのデータディレクトリ(例、`C:\ProgramData\Contrast\dotnet-core`)を作成します。
- IIS に.NET Core エージェントのネイティブモジュールを追加します。

インストールを行う前に

インストールを開始する前に、システム要件 (226ページ)とサポート対象テクノロジ (225ページ)を確認し、インストールを行うことができ、最適なパフォーマンスを得られることを確認してください。

Contrast を使用してエージェントをインストール

1. Contrast Web インターフェイスで、**新規登録**を選択します。
2. **エージェントを選択**のドロップダウンメニューから.NET Core を選択します。

3. **IIS でのインストール**の下で、**IIS 用.NET Core エージェントインストーラ**のリンク を選択します。ZIP アーカイブがダウンロードされます。
4. ダウンロードした ZIP アーカイブを Web サーバ上で解凍したら、`contrast-dotnet-core-agent-for-iis-installer.exe` を実行します。これで、IIS 用の.NET Core エージェントがインストールされます。

> **ヒント**
> コマンドラインを使用してインストールすると、IIS 用.NET Core エージェントインストーラでサポートされるその他のオプションを利用できます。

5. YAML 設定ファイルを使用して.NET Core エージェントを設定 (248ページ)し、認証キー (61ページ)およびアプリケーション固有の設定を指定します。
6. YAML ファイルを `C:\ProgramData\Contrast\dotnet-core` にコピーします(まだコピーしていない場合)。
7. IIS を再起動して変更を反映します。
8. 通常通りにアプリケーションを疎通し、Contrast でアプリケーションが認識されていることを確認します。

コマンドラインを使用してエージェントをインストール

コマンドラインを使用すると、IIS 用.NET Core エージェントインストーラでサポートされるその他のオプションを利用できます。

IIS 用.NET Core エージェントは、Windows のインターフェイスを使用してインストールでき、Windows の標準機能(コントロールパネルのプログラムと機能や Powershell など)を使用して、アンインストールや修復を行うことができます。ただし、自動化されたスクリプトなどの特定のシナリオでは、IIS 用.NET Core エージェントのインストーラを使用して、以下の操作を実行する場合があります。

有人モードには、以下のコマンドを使用します。

- **インストール：** `contrast-dotnet-core-agent-for-iis-installer.exe`
- **アンインストール：** `contrast-dotnet-core-agent-for-iis-installer.exe -uninstall`
- **修復：** `contrast-dotnet-core-agent-for-iis-installer.exe -repair`

無人モードまたはサイレントモードには、以下のコマンドを使用します。

- **インストール：** `contrast-dotnet-core-agent-for-iis-installer.exe -s SUPPRESS_RESTARTING_IIS=1`
- **アンインストール：** `contrast-dotnet-core-agent-for-iis-installer.exe -uninstall -s SUPPRESS_RESTARTING_IIS=1`
- **修復：** `contrast-dotnet-core-agent-for-iis-installer.exe -repair -s SUPPRESS_RESTARTING_IIS=1`

コマンドラインを使用してインストールすると、IIS 用.NET Core エージェントインストーラで利用できるその他のオプションがあります。

オプション	説明	例
`INSTALLFOLDER`	エージェントファイルのインストールディレクトリを指定します。	`INSTALLFOLDER=C:Program Files\Contrast\dotnet-core`
`AGENT_EXPLORER_INSTALLFOLDER`	Agent Explorer のファイルのディレクトリを指定します。	`AGENT_EXPLORER_INSTALLFOLDER="C:\Program Files\Contrast\agent-explorer"`

オプション	説明	例
INSTALL_AGENT_EXPLORER	Agent Explorer をインストールしない場合は、このオプションの値を 0 に設定して下さい。 デフォルトの値は 1 で、 Agent Explorer はインストールされます。	INSTALL_AGENT_EXPLORER=1
DATAFOLDER	エージェントのログファイルと設定ファイルのデフォルトの場所を指定します。	DATAFOLDER=C:\ProgramData\Contrast\dotnet-core
SUPPRESS_RESTARTING_IIS	このオプションの値を 1 に設定すると、インストーラは IIS を再起動しません。 デフォルトの値は、0 です。 **注記** • IIS が再起動されるまで、アプリケーションにエージェントは組み込まれません。 • SUPPRESS_RESTARTING_IIS を設定すると、アップグレードの実行時に IIS にアクティブなワーカーがない限り、自動アップグレードが実行されなくなります。	SUPPRESS_RESTARTING_IIS=0
INSTALL_UPGRADE_SERVICE	エージェントアップグレードサービスをインストールしない場合は、このオプションの値を 0 に設定して下さい。 デフォルトの値は 1 で、エージェントアップグレードサービスはインストールされます。	INSTALL_UPGRADE_SERVICE=1
UPGRADE_SERVICE_INSTALLFOLDER	アップグレードサービスのファイルのディレクトリを指定します。	UPGRADE_SERVICE_INSTALLFOLDER="C:\Program Files (x86)\Contrast\upgrade-service"

重要

IIS 用.NET Core エージェントインストーラは、エージェントを初めてインストールする際に IIS を自動的に再起動します。必要があれば、IIS の再起動時に警告を発するよう Web サーバ監視ツールの設定を変更してください。

.NET プロファイル API では、プロファイルするプロセスをプロファイラで起動する必要があります。そのため、.NET Core エージェントは Contrast プロファイラをロードするために IIS(および IIS のワーカープロセス)を再起動する必要があります。この処理は、他のプロファイル製品(例、メモリやパフォーマンスプロファイラなど)の動作と同様です。

コンテナを使用して.NET Core エージェントをインストール

インストールを行う前に

- 本項では、Docker を例として、コンテナ化されたアプリケーションに Contrast .NET Core エージェントをインストールするための一般的な手順について説明します。
- コンテナや関連ソフトウェアの仕組みを基本的に理解している必要があります。必要に応じて、お客様の環境に合わせて手順を調整してください。
- Kubernetes を使用している場合は、エージェントオペレータ (498ページ)を使用してエージェントを設定することを検討してください。

手順 1：エージェントをインストール

Contrast エージェントは、アプリケーションをコンテナイメージに追加する前または後のどちらでも追加できます。推奨される方法は、名前を付けたマルチステージビルドを使用することです。例：

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0

# Hidden for brevity...

# Copy the required agent files from the official Contrast agent image.
COPY --from=contrast/agent-dotnet-core:latest /contrast /contrast
```

この例では、最新の.NET Core エージェントを使用します(利用可能なタグは DockerHub で確認してください)。

手順 2：エージェントを設定

Contrast エージェントは、複数のソースからの設定を受け入れますが、設定の優先順位は優先順位 (62ページ)セクションに記載されています。

設定方法を組み合わせて利用することをお勧めします。

- YAML ファイルを使用して、複数のアプリケーションで共有する共通の設定を指定します。
- アプリケーション固有の設定値や、YAML ファイルで指定した値を上書きする場合や、実行時に組み込む機密情報などには、環境変数を使用します。

YAML ファイルの設定：

YAML ファイル (63ページ)を使用してエージェントを設定する場合に、環境変数の `CONTRAST_CONFIG_PATH` を使用して、YAML ファイルがコンテナ内のどこにあるかを指定することもできます。

例えば、`contrast_security.yaml` という名前の YAML 設定ファイルが Docker のビルドコンテキストに存在するとします。

環境変数 `CONTRAST_CONFIG_PATH` を使用して、YAML ファイルの場所を指定することもできます。

```
agent:
  logger:
    path: /var/tmp
    level: WARN
```

YAML ファイルは、以下のようにコンテナイメージに追加することができます。

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0

# Hidden for brevity...

# Add the Contrast agent to the image.
COPY --from=contrast/agent-dotnet-core:latest /contrast /contrast

# Copy the contrast_security.yaml file from Docker build context.
COPY ./contrast_security.yaml /contrast_security.yaml

# Finally configure configure the agent to use the YAML file previously \
copied.
ENV CONTRAST_CONFIG_PATH=/contrast_security.yaml
```

環境変数の設定：

アプリケーション固有の設定を指定するには、環境変数 (66ページ)を使用します。以下は、一般的によく使用される設定オプションです。

項目	用途	環境変数
アプリケーション名	Contrast サーバに報告されるアプリケーション名を指定します。	`CONTRAST__APPLICATION__NAME`
アプリケーショングループ	オンボード時にこのアプリケーションと関連付けるアクセスグループを指定します。 **注記** アプリケーションのアクセスグループは、先に Contrast で作成しておく必要があります。	`CONTRAST__APPLICATION__GROUP`
アプリケーションのタグ	アプリケーションにタグを追加します。	`CONTRAST__APPLICATION__TAGS`
サーバ名	Contrast に報告されるサーバ名を指定します。	`CONTRAST__SERVER__NAME`
サーバの環境	アプリケーションを実行する環境を指定します。このオプションで有効な値：Development、QA、Production	`CONTRAST__SERVER__ENVIRONMENT`
サーバのタグ	サーバにタグを追加します。	`CONTRAST__SERVER__TAG`

手順 3：プロファイル変数と認証情報を追加

アプリケーションに.NET エージェントを組み込んで検査を行うには、さらに環境変数 (227ページ)が必要です。`CORECLR_` 変数はエージェントをロードし、`CONTRAST_` 変数は Contrast サーバに対するエージェントの認証用です。

前述の Dockerfile の例を使用すると、以下のようになります。

x64

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0

# Hidden for brevity...

COPY --from=contrast/agent-dotnet-core:latest /contrast /contrast

# Required variables to load the agent.
ENV CORECLR_PROFILER_PATH_64=/contrast/runtimes/linux-x64/native/
ContrastProfiler.so \
    CORECLR_ENABLE_PROFILING=1 \
    CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
```

ARM64

```
FROM mcr.microsoft.com/dotnet/aspnet:6.0

# Hidden for brevity...

COPY --from=contrast/agent-dotnet-core:latest /contrast /contrast

# Required variables to load the agent.
ENV CORECLR_PROFILER_PATH_64=/contrast/runtimes/linux-arm64                \
                                                                           \
                                                                           \
                                                                           \
```

```
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                                                                \
                                  /native/ContrastProfiler.so \
    CORECLR_ENABLE_PROFILING=1 \
    CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
```

サーバへのエージェントの認証には、以下の環境変数に API の値 (61ページ)を設定する必要があります。

```
CONTRAST__API__URL=https://app.contrastsecurity.com/Contrast
CONTRAST__API__API_KEY={Your API KEY here}
CONTRAST__API__SERVICE_KEY={Your Service key here}
CONTRAST__API__USER_NAME={Your agent username here}
```

API の値(エージェントキー (61ページ))は Contrast Web インターフェイスで確認するか、.NET Core エージェント用の YAML ファイルをダウンロードすることで取得できます。

重要

API_KEYSERVICE_KEY、USER_NAME の各キーは、機密データとなりますので、取り扱いにはご注意ください。Contrast では、これらのデータは、シークレットストア(例 : Kubernetes Secret など)から実行時に取り込むことをお勧めします。

手順 4 : アプリケーションを検査

これで、Contrast を有効にしてアプリケーションイメージを実行できます。アプリケーションの起動時に Contrast エージェントが組み込まれて解析が始まり、Contrast サーバに脆弱性が報告されます。Contrast が実行されているかを確認するには、コンテナをチェックしてください。

Terraform で.NET エージェントをインストール

本手順では、Terraform を使用して Azure にデプロイする際に、Contrast の.NET Framework エージェントおよび.NET Core エージェントをインストールする方法について説明します。この手順は、ご利用の環境に合わせてカスタマイズする必要がある場合があります。

Contrast エージェントを Azure App Service にデプロイするのに最適な方法は、サイト拡張です。これは、Azure Portal、ARM ポリシー、または Azure API を使用する必要があります。本手順で説明する Terraform でのインストール方法は、後者の 2 つの方法を直接または間接的に使用します。

開始する前に

- Azure App Service で実行する.NET Framework や.NET Core エージェントに対して、利用する OS やランタイムスタックが Contrast でサポートされていることを確認してください。
  - .NET Core エージェントのサポート対象テクノロジ (225ページ)
  - .NET Framework エージェントのサポート対象テクノロジ (167ページ)
- 以下の要件を満たしていることを確認してください。
  - Contrast への口グインアクセスがあること
  - Terraform と Azure CLI がインストールされているシステムへのコンソールアクセスがあること
  - Azure CLI の `az login` も含む、Azure Portal への口グインアクセスがあること
  - 本手順のコマンドを実行するシステムに Python がインストールされていること
  - Azure App Service (232ページ)の一部として Contrast エージェントが含まれていること

手順 1：エージェントを設定

1. Contrast からエージェントの設定ファイルをダウンロードします。
   まず、Contrast Web インターフェイスで「新規登録」を選択して、表示される手順に従って YAML 設定ファイルをダウンロードします。
   a. Contrast Web インターフェイスで、**新規登録**を選択します。
   b. 「アプリケーション」で、**新規登録**を選択し、次へをクリックします。
   c. 使用するエージェントを選択し、表示されている手順に従って、YAML 設定ファイルをダウンロードします。
2. YAML 設定ファイルに、以下の値を設定します。
   - .NET Core エージェント

```
CORECLR_ENABLE_PROFILING:1
CORECLR_PROFILER:{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
CORECLR_PROFILER_PATH_32:
 D:\\home\\SiteExtensions\\Contrast.NetCore.Azure.SiteExtension\
\ContrastNetCoreAppService\\contrast\\runtimes\\win-x86\\native\
\ContrastProfiler.dllCORECLR_PROFILER_PATH_64: D:\\home\
\SiteExtensions\\Contrast.NetCore.Azure.SiteExtension\
\ContrastNetCoreAppService\\\contrast\\runtimes\\win-x64\\native\
\ContrastProfiler.dll
```

   - .NET Framework エージェント

```
COR_ENABLE_PROFILING: 1
COR_PROFILER: {EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}
COR_PROFILER_PATH_32: D:\\home\\SiteExtensions\
\Contrast.NET.Azure.SiteExtension\\ContrastAppService\
\ContrastProfiler-32.dllCOR_PROFILER_PATH_64: D:\\home\\SiteExtensions\
\Contrast.NET.Azure.SiteExtension\\ContrastAppService\
\ContrastProfiler-64.dll
```

手順 2：Terraform でサイト拡張を設定

サイト拡張のデプロイは、Azure Portal か、ARM ポリシー、または Azure API を使用してのみ標準にサポートされているため、Terraform はサイト拡張を追加や削除するための便利なコマンドラインメソッドです。以下の例に示すように、ARM ポリシーを使用して拡張機能を設定します。

以下の手順で、アプリケーションに Contrast エージェントを組み込みます。

1. 手順 1 で準備した YAML 設定ファイルの名前が、`contrast_security.yaml` であることを確認します。
2. Terraform をこちらよりインストールします：https://www.terraform.io/downloads.html
3. 以下のコマンドで PyYAML をインストールします。

```
pip install PyYAML
```

4. Azure CLI ツールをこちらよりインストールします：https://docs.microsoft.com/en-us/cli/azure/install-azure-cli
5. `az login` を使用して、Azure にログインし、認証情報がキャッシュできることを確認します。
6. YAML をパースする以下のスクリプト `parseyaml.py` を使用して、Contrast の YAML ファイルから値を抜き出し、プロビジョニングされた Azure App Service にそれらの値を追加します。

```
import yaml,
 jsonwith open('./contrast_security.yaml') as f:
    config = yaml.load(f)
    print(json.dumps(config['api']))
```

7. `main.tf` という Terraform のドキュメントを以下のように修正します。

```
provider "azurerm" {
  features {}
}
# Create a resource group
resource "azurerm_resource_group" "personal" {
  name     = <name>
  location = <location>
}
# Create an app service plan
resource "azurerm_app_service_plan" "app_service-plan"{
  name                = <name>
  resource_group_name = azurerm_resource_group.personal.name
  location            = <location>
}
# Create an app service
resource "azurerm_app_service" "app_service" {
  name                = <name>
  location            = <location>
  resource_group_name = azurerm_resource_group.personal.name
  app_service_plan_id =azurerm_app_service_plan.app_service-plan.id
  site_config {
    dotnet_framework_version = "v4.0"
    default_documents        = ["Default.aspx"]
  }
# CONTRAST .NET FRAMEWORK AGENT SETUP
# Contrast env vars will be passed to the app service here.
  app_settings = {
    "COR_ENABLE_PROFILING"                    = "1"
    "COR_PROFILER"                            = "{EFEB8EE0-6D39-4347-
A5FE-4D0C88BC5BC1}"
    "COR_PROFILER_PATH_32"                    = "D:\\home\
\SiteExtensions\\Contrast.NET.Azure.SiteExtension\\ContrastAppService\
\ContrastProfiler-32.dll"
    "COR_PROFILER_PATH_64"                    = "D:\\home\
\SiteExtensions\\Contrast.NET.Azure.SiteExtension\\ContrastAppService\
\ContrastProfiler-64.dll"
    "CONTRAST_INSTALL_DIRECTORY"              = "D:\\home\
\SiteExtensions\\Contrast.NET.Azure.SiteExtension\\ContrastAppService\\"
    "CONTRAST__API__URL"                      = \
data.external.yaml.result.url
    "CONTRAST__API__USER_NAME"                 = \
data.external.yaml.result.user_name
    "CONTRAST__API__SERVICE_KEY"              = \
data.external.yaml.result.service_key
```

```
    "CONTRAST__API__API_KEY"                  = \
data.external.yaml.result.api_key
    # USE THESE SETTING FOR .NET CORE AGENT
    #"CORECLR_ENABLE_PROFILING" = 1
    #"CORECLR_PROFILER"         = {8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
    #"CORECLR_PROFILER_PATH_32" = D:\\home\\SiteExtensions\
\Contrast.NetCore.Azure.SiteExtension\\ContrastNetCoreAppService\
\contrast\\runtimes\\win-x86\\native\\ContrastProfiler.dll
    #"CORECLR_PROFILER_PATH_64" = D:\\home\\SiteExtensions\
\Contrast.NetCore.Azure.SiteExtension\\ContrastNetCoreAppService\\
\contrast\\runtimes\\win-x64\\native\\ContrastProfiler.dll
  }
}
#Extract the connection from the normal yaml file to pass to the app \
container
data "external" "yaml" {
  program = [var.python_binary, "${path.module}/parseyaml.py"]
}
# Deploy the extension template
resource "azurerm_template_deployment" "extension" {
  name                = <name>
  resource_group_name = <resource_group_name>
  template_body       = <<BODY
{
   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/
deploymentTemplate.json#",
   "contentVersion": "1.0.0.0",
   "parameters": {
       "siteName": {
           "type": "string",
           "metadata": {
               "description": "The Azure App Service Name"
           }
       },
       "extensionName": {
           "type": "string",
           "metadata": {
               "description": "The Site Extension Name."
           }
       }
   },
  "resources": [
       {
           "type": "Microsoft.Web/sites/siteextensions",
           "name": "[concat(parameters('siteName'),
'/', parameters('extensionName'))]",
           "apiVersion": "2019-08-01",
           "location": "[resourceGroup().location]"
       }
   ]
}
  BODY  parameters = {
    "siteName"          = azurerm_app_service.<app_service>.name
    #.NET Framework
    "extensionName"     = "Contrast.NET.Azure.SiteExtension"
```

```
    #.NET Core
    # "extensionName"     = "Contrast.NetCore.Azure.SiteExtension"
 }
  deployment_mode     = "Incremental"
}
```

エージェントアップグレードサービス

エージェントアップグレードサービスは、Windows のバックグラウンドサービスで、Windows 上の.NET Framework エージェントおよび IIS 用.NET Core エージェントを最新のバージョンに自動的に更新することができます。エージェントアップグレードサービスは、.NET Framework エージェントのインストーラと IIS 用.NET Core エージェントのインストーラに含まれており、エージェントインストーラは以下の 2 つの製品をインストールします。

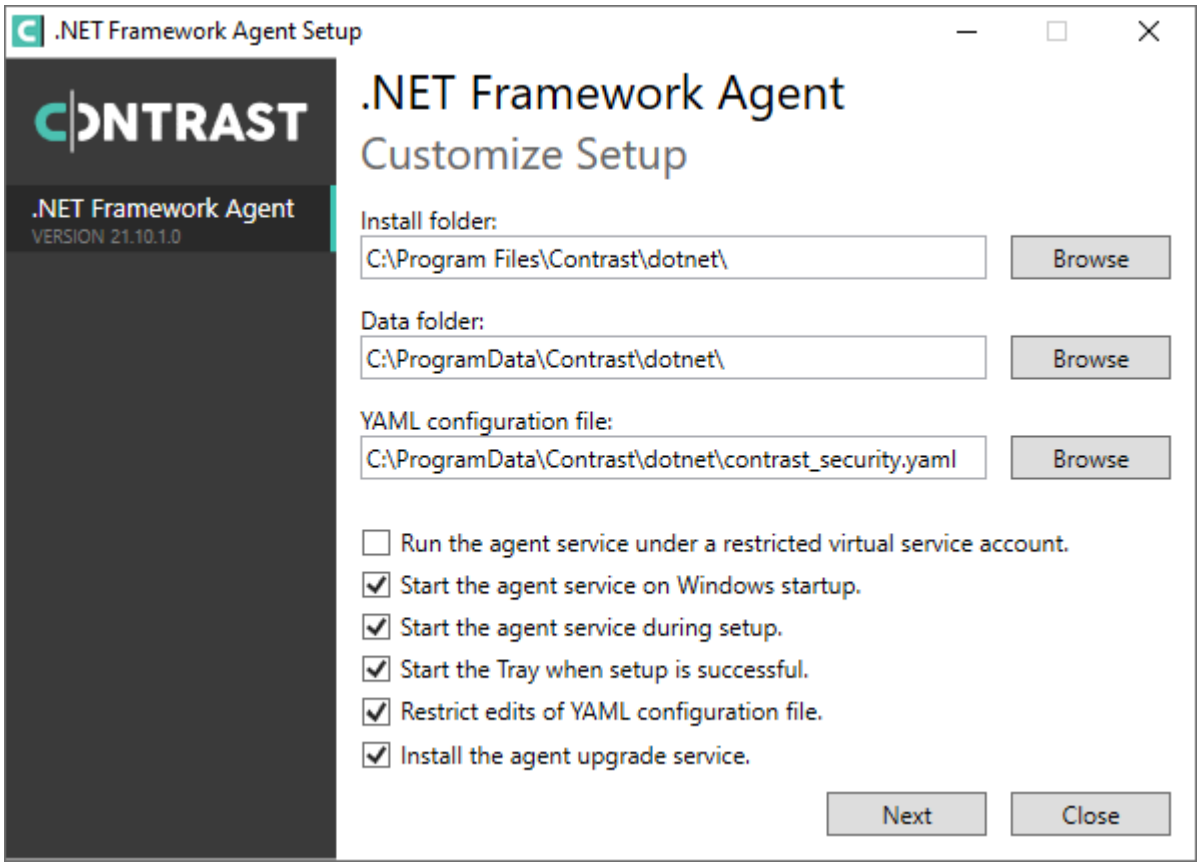
- 該当のエージェント
- エージェントアップグレードサービス

デフォルトでは、エージェントアップグレードサービスは、サービスの初回起動時(Windows Server の再起動時)に、NuGet にリリースされている新しいエージェントをチェックします。新しいエージェントのバージョンが見つかった場合、アップグレードサービスは、新しいバージョンをダウンロードし、インストーラの署名を検証してから、最後にインストーラを実行します。

注記
新しいバージョンのエージェントがインストールされると、IIS は再起動されます。

エージェントアップグレードサービスはオプションコンポーネントであり、エージェントの Assess および Protect 機能には必須ではありません。

- エージェントアップグレードサービスを使用しない場合は、**Install the agent upgrade service**(エージェントアップグレードサービスをインストールする)のチェックボックスをオフにしてください。
- コマンドラインでエージェントをインストールする場合は、`INSTALL_UPGRADE_SERVICE=0` の引数を追加すれば、エージェントアップグレードサービスはインストールされません。



エージェントアップグレードサービスの動作は、Contrast のデータディレクトリにあるエージェント用の設定ファイルで変更できます。デフォルトの場所は、`C:\ProgramData\Contrast\upgrade-service` です。

.NET Core エージェントをアップグレードする際の設定は、.NET Core エージェントの YAML ファイル内にあります。

```
enable: true #  Set to `true` for the agent to automatically upgrade to \
newer versions.
checks: Startup # Set the frequency with which the agent checks for \
updates. Valid values are `daily` for every 24 hours and on startup, or \
`startup` for *only* when service starts up.
timeout_ms: 60000 # Set the time allocated to execute the downloaded agent \
installer before cancelling.
nuget_repository_url: https://api.nuget.org/v3/index.json # Set the URL of \
the Nuget repository to be used for the .NET Core Agent for IIS Installer
nuget_package_name: Contrast.CoreIIS.Installer # Set the name of the .NET \
Core Agent for IIS Nuget package.
installer_upgrade_code: 82468c04-dfc0-4a4c-9eb9-c4b314c67fdc # Used \
internally to retrieve the current installed agent version from Windows.
```

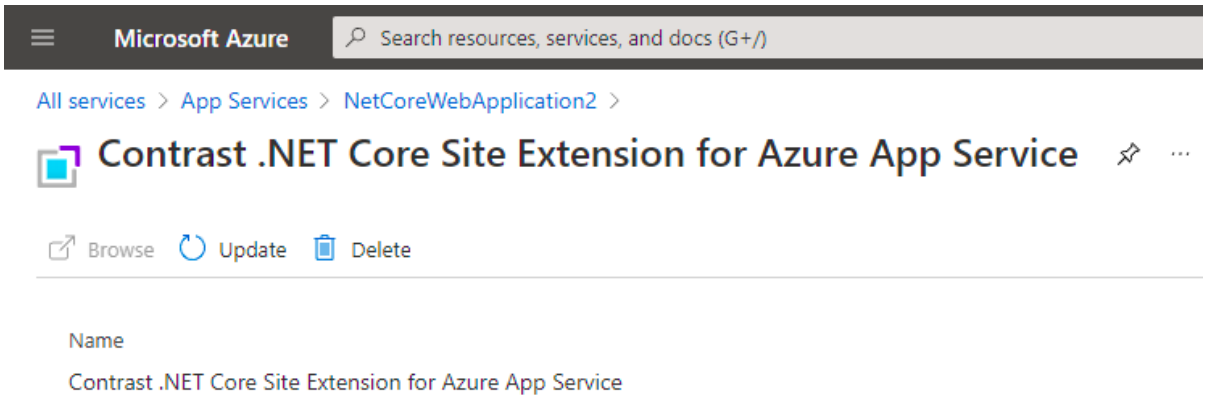
注記

エージェントアップグレードサービスは、エージェントインストーラにのみ含まれます。手動で取得する.NET Core エージェント、エージェントの NuGet パッケージ、Azure App Service のサイト拡張機能には含まれません。

.NET Core エージェントのアップデート

Contrast エージェントは新しいバージョンが頻繁にリリースされますが、ここでは、NuGet パッケージや手動で.NET Core エージェントを簡単にアップデートし最新にする方法について説明します。 インストールの種別により、以下の方法でアップデートします。

- **IIS 用.NET Core エージェントのインストーラ**：エージェントアップグレードサービス (179ページ)を使用します。
- **Azure App Service**：Azure Portal を使用します。



- **手動インストール**： 以下の手順に従い、独自に自動化を設定します。

開始する前に

- Contrast .NET Core エージェントを使用しない状態で.NET Core アプリケーションが正常に実行されることを確認します。
- 事前に Contrast.NET Core エージェントをインストールしておきます。
- 変更管理ポリシーと使用する環境に基づいて、エージェントをアップデートする方法とタイミングを決めます。
- アプリケーションの依存関係とアップデートを管理するワークフローを決めます。

手順

1. Contrast .NET Core エージェントを同じインストール場所にダウンロードしますが、Contrast リポジトリを使用します。
   - Saas 版の Contrast をご利用の場合：.NET Core エージェントは公開 NuGet リポジトリの最新リリースと同期されています。
   - オンプレミス版の Contrast をご利用の場合：お使いの Contrast インスタンスより新しいバージョンの Contrast エージェントを使用することは推奨されません。Contrast Web インターフェイスからダウンロードできるものと同じバージョンの.NET Core エージェントのバージョンを使用してください。
2. 次の API 情報を準備します。

```
CONTRAST_URL=<TeamServer URL e.g. https://app.contrastsecurity.com >
ORG_ID=<YOUR TEAMSERVER ORGANIZATION ID>
AUTH_TOKEN=<YOUR TEAMSERVER AUTHENTICATION TOKEN>
API_KEY=<YOUR TEAMSERVER API KEY>
```

3. 次のいずれかのスクリプトを使用して、Contrast .NET Core エージェントをダウンロードします。アプリケーションの起動スクリプトや自動化されたデプロイメントパイプライン、cron ジョブなどにこのスクリプトを追加すると、エージェンを自動的にアップデートできます。
   - Bash スクリプト

```
CONTRAST_URL=https://app.contrastsecurity.com
ORG_ID=xxxx
```

```
AUTH_TOKEN=xxxx
API_KEY=xxxx
curl -X GET $CONTRAST_URL/Contrast/api/ng/$ORG_ID/agents/default/
DOTNET_CORE /-o ./Contrast.NET.Core.zip -H 'Authorization: \
$AUTH_TOKEN' -H 'API-Key: $API_KEY' /-H 'Accept: application/json' -OJ
```

- Powershell

```
$ContrastUrl = "https://app.contrastsecurity.com/Contrast"
$UserId = ""
$ServiceKey = ""
$ApiKey = ""
$OrganizationId = ""
$InstallPath = ".\dotnet-core"

# Needed if the OS defaults to Tls1.1.
[Net.ServicePointManager]::SecurityProtocol = \
[Net.SecurityProtocolType]::Tls12

New-Item -ItemType Directory $InstallPath

Invoke-WebRequest `
    -Uri "$ContrastUrl/api/ng/$OrganizationId/agents/default/
DOTNET_CORE" `
    -Headers @{
        "API-Key"       = $ApiKey
        "Authorization" = \
[Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes("$
{UserId}:${ServiceKey}"))
    } `
    -OutFile "$InstallPath\Contrast.zip"

Invoke-WebRequest -Uri `
    "$ContrastUrl/api/ng/$OrganizationId/agents/external/default/
DOTNET_CORE" `
    -Headers @{
        "Accept"        = "text/yaml"
        "API-Key"       = $ApiKey
        "Authorization" = \
[Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes("$
{UserId}:${ServiceKey}"))
    } `
    -OutFile "$InstallPath\contrast_security.yaml"

Expand-Archive "$InstallPath\Contrast.zip" -DestinationPath \
$InstallPath
Remove-Item "$InstallPath\Contrast.zip"
```

4. ダウンロードしたファイルを解凍して、現在の Contrast エージェントの配置場所に保存します。配置する場所がわからない場合は、お使いのシステムの以下のコマンドで環境変数を調べてください。
   - Windows(64 ビット)

```
echo %CORECLR_PROFILER_PATH_64%CORECLR_PROFILER_PATH_64
```

   - Windows(32 ビット)

```
CORECLR_PROFILER_PATH_32
```

   - Linux(64 ビット)

```
CORECLR_PROFILER_PATH_64
```

- Powershell

```
printenv CORECLR_PROFILER_PATH_64
```

.NET Core エージェントの設定

全てのエージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。

ご利用の環境に合わせて、.NET Core エージェントを設定してください。

- Azure App Service (247ページ)
- 環境変数 (247ページ)
- YAML 設定ファイル (248ページ)
- インテグレーション (726ページ)

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

Azure App Service の.NET Core エージェントを設定

Azure App Service を使用している場合、次の方法で.NET Core エージェントを設定できます。

- **Azure Portal**： 環境変数 (247ページ)を使用して.NET Core エージェントを設定します。
  環境変数の構文を使用して、**構成**メニューの**アプリケーション設定**の画面で全ての設定を追加します。
- **Web.config ファイルの環境変数**：<aspNetCore> 要素の<environmentVariables>セクションに、環境変数の構文を使用してオーバライドを指定します。
- **YAML 設定ファイル**：アプリケーションのデプロイメントに含めるか、Kudu コンソールを使用して、このファイルを Azure Web アプリケーションにアップロードします。
  構成\アプリケーション設定 画面で、CONTRAST_CONFIG_PATH という新しいアプリケーション設定を追加して、このファイルを指す値を指定します。
  例えば、アプリケーションルートの *contrast_security.yaml* ファイルを使用するには、 **構成\アプリケーション設定**画面にアクセスし、CONTRAST_CONFIG_PATH というキーに *D:\Home\site\wwwroot\contrast_security.yaml* という値で、新しいアプリケーション設定を追加します。Azure App Service のアプリケーションファイルが、*D:\home\site\wwwroot* にデプロイされます。

関連項目

- Azure App Service で.NET Core エージェントをインストール (232ページ)

環境変数を使用して.NET Core エージェントを設定する

環境変数は、いくつかの方法で設定できます。

- IIS では、web.config ファイルを使用してアプリケーションの環境変数を設定できます。
- Azure App Service では、Web サイトの環境変数を設定するための UI が Azure プラットフォームに用意されています。
- 開発時には、launchSettings.json ファイルを使用して、起動されるアプリケーションの環境変数を設定できます。

ヒント

.NET Core エージェントの YAML テンプレート (248ページ)にあるプロパティは、いずれも環境変数に変換できます。

- エージェントのログレベル(`agent.logger.level`)を"TRACE"に変更するには、`CONTRAST__AGENT__LOGGER__LEVEL` というキーに"TRACE"という値を指定します。
- エージェントのサーバ名(`server.name`)を"MyServer"に変更するには、`CONTRAST__SERVER__NAME` というキーに"MyServer"という値を指定します。

最も一般的な設定のいくつかは次のとおりです。

環境変数	目的
CONTRAST__APPLICATION__NAME	Contrast サーバに報告されるアプリケーション名を指定
CONTRAST__APPLICATION__GROUP	このアプリケーションを関連付けるアクセスグループを指定(アクセスグループ (822ページ)は作成済みであること)。
CONTRAST__APPLICATION__SESSION_METADATA	Contrast でセッションの新規作成時に使用されるメタデータを指定。エージェントによって検出された脆弱性は、この新しいセッションメタデータに関連付けられます。
CONTRAST__SERVER__NAME	Contrast に報告されるサーバ名を指定
CONTRAST__SERVER__ENVIRONMENT	アプリケーションを実行する環境を指定(Development、QA、Production)

その他の利用可能なプロパティの説明は、.NET Core エージェントの YAML テンプレート (248ページ)を参照してください。

.NET Core の YAML 設定ファイルのテンプレート

YAML 設定ファイルを使用して.NET Core エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、YAML 設定の説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します。

- **Windows**：`C:/ProgramData/contrast/dotnet-core/contrast_security.yaml`
- **Unix**：`/etc/contrast/dotnet-core/contrast_security.yaml`

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
```

```
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the version of the TLS protocol the agent uses to communicate with \
the
  # Contrast UI. The .NET agent default behavior is \
(SecurityProtocolType.Tls
  # | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls12).
  # tls_versions: tls|tls11|tls12

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Determine the location from which the agent loads a client
    # certificate. Value options include `File` or `Store`.
    # certificate_location: NEEDS_TO_BE_SET

    # Set the absolute path to the client certificate's
    # .CER file for communication with Contrast UI. The
    # `certificate_location` property must be set to `File`.
    # cer_file: NEEDS_TO_BE_SET
```

```
    # Specify the name of certificate store to open. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `AuthRoot`, `CertificateAuthority`,
    # `My`, `Root`, `TrustedPeople`, or `TrustedPublisher`.
    # store_name: NEEDS_TO_BE_SET

    # Specify the location of the certificate store. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `CurrentUser` or `LocalMachine`.
    # store_location: NEEDS_TO_BE_SET

    # Specify the type of value the agent uses to find the certificate
    # in the collection of certificates from the certificate store.
    # The `certificate_location` property must be set to `Store`.
    # Value options include `FindByIssuerDistinguishedName`,
    # `FindByIssuerName`, `FindBySerialNumber`,
    # `FindBySubjectDistinguishedName`, `FindBySubjectKeyIdentifier`,
    # `FindBySubjectName`, or `FindByThumbprint`.
    # find_type: NEEDS_TO_BE_SET

    # Specify the value the agent uses in combination with
    # `find_type` to find a certification in the certificate store.
    #
    # Note - The agent will use the first certificate from
    # the certificate store that matches this search criteria.
    #
    # find_value: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
```

```
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

      # Set the connection type used for Syslog messages.
      # Value options are `UNENCRYPTED` and `ENCRYPTED`.
      # connection_type: UNENCRYPTED

  # \
================================================================================
  # agent.dotnet
  # The following properties apply to any .NET agent-wide configurations.
  # \
================================================================================
  # dotnet:

    # Set a list of application pool names that the agent does not
```

```
# instrument or analyze. Names must be formatted as a comma-separated
# list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
# app_pool_denylist: NEEDS_TO_BE_SET

# Set a list of application pool names that the agent instruments or
# analyzes. If set, other application pools are ignored. Allowlist takes
# precedence over denylist. Names must be formatted as a comma-separated
# list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
# app_pool_allowlist: NEEDS_TO_BE_SET

# Set a list of application names that the agent does not
# analyze. (The applications are still instrumented).
# Names must be formatted as a comma-separated list.
# New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# application_denylist: NEEDS_TO_BE_SET

# Set a list of application names that the agent analyzes.
# If set, other applications are not analyzed, but are
# still instrumented. Allowlist takes precedence over
# denylist. Names must be formatted as a comma-separated
# list. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# application_allowlist: NEEDS_TO_BE_SET

# Enable a profiler chaining feature to allow Contrast to
# work alongside other tools that use the CLR Profiling
# API. Defaults to `true`. New after .NET Framework 19.1.3
# (Installed Only) and .NET Core 1.9.3 (Installed Only).
# enable_chaining: true

# Indicate that the agent should allow CLR optimizations
# of JIT-compiled methods. Defaults to `true`. New
# after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_instrumentation_optimizations: true

# Indicate that the agent should allow the CLR to inline
# methods that are not instrumented by Contrast. Defaults to
# `true`. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_jit_inlining: true

# Indicate that the agent should allow the CLR to perform
# transparency checks under full trust. Defaults to `false`.
# New after .NET Framework 19.1.3 and .NET Core 1.0.0.
# enable_transparency_checks: false

# Set to display ASCII art to std::out on agent startup. Defaults
# to `true`. New after .NET Framework 20.6.3 and .NET Core 1.0.0.
# enable_cat: true

# Sets the maximum amount of time a Protect regular expression
# is allowed to run before being cancelled. Set to -1 to never
# cancel regular expression execution. Defaults to `20_000`.
# New after .NET Framework 20.4.3 and .NET Core 1.5.0.
# protect_searchers_single_pattern_deadline_ms: 20_000

# Sets the maximum amount of time a 'Probe Analysis' Protect
```

```
    # regular expression is allowed to run before being cancelled. Set
    # to -1 to never cancel regular expression execution. Defaults to
    # `5_000`. New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # protect_searchers_probe_analysis_single_pattern_deadline_ms: 5_000

    # Sets the maximum amount of time a Protect rule is
    # allowed to run before being cancelled. Set to -1 to never
    # cancel Protect rule execution. Defaults to `60_000`.
    # New after .NET Framework 20.4.3 and .NET Core 1.5.0.
    # protect_searchers_total_rule_deadline_ms: 60_000

    # Sets the maximum amount of time a 'Probe Analysis' Protect
    # rule is allowed to run before being cancelled. Set to -1 to
    # never cancel Protect rule execution. Defaults to `10_000`.
    # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # protect_searchers_probe_analysis_total_rule_deadline_ms: 10_000

    # Sets the maximum duration of time agent log files should be kept
    # since last write before being deleted by the agent. Defaults to
    # `604_800_000`. New after .NET Framework 20.6.1 and .NET Core 1.5.5.
    # log_cleanup_maximum_age_ms: 604_800_000

    # Suppresses gathering process-level metrics (process level metrics are
    # gathered by default), used to identify performance problems. Metric
    # counters may further decrease the stability of already unstable
    # systems and can be disabled (set to true) if issues occur. Defaults
    # to `false`. New after .NET Framework 20.6.6 and .NET Core 1.5.10.
    # suppress_metric_counters: false

    # Enable file based application watching. Set to false if
    # file watching is causing locking issues. Defaults to `true`.
    # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # enable_file_based_app_watching: true

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Specifies the cloud provider from which the agent should gather metadata
  # (such as resource identifiers). Options are `AWS`, `Azure`, or `GCP`.
```

```
  #
  # gather_metadata_via: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000
```

```
  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:
```

```
      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-chaining
    # Use the following properties to configure how the
    # sql injection semantic analysis chaining rule works.
    # \
================================================================================
    # sql-injection-semantic-chaining:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-dangerous-functions
    # Use the following properties to configure how the sql
    # injection semantic analysis dangerous functions rule works.
    # \
================================================================================
    # sql-injection-semantic-dangerous-functions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-suspicious-unions
    # Use the following properties to configure how the sql
    # injection semantic analysis suspicious unions rule works.
    # \
================================================================================
    # sql-injection-semantic-suspicious-unions:

      # Set the mode of the rule. Value options
```

```
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-tautologies
    # Use the following properties to configure how the sql
    # injection semantic analysis tautologies rule works.
    # \
================================================================================
    # sql-injection-semantic-tautologies:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
```

```
    # \
================================================================================
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
```

```
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
================================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:
```

```
      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET
```

```
  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
```

```
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Override the reported server path. New after
  # .NET Framework v21.3.1 and .NET Core v1.8.0.
  # path: NEEDS_TO_BE_SET
```

```
# \
==========================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
==========================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
==========================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
==========================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
```

```
  user_name: NEEDS_TO_BE_SET

  # Set the version of the TLS protocol the agent uses to communicate with \
the
  # Contrast UI. The .NET agent default behavior is \
(SecurityProtocolType.Tls
  # | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls12).
  # tls_versions: tls|tls11|tls12

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Determine the location from which the agent loads a client
    # certificate. Value options include `File` or `Store`.
    # certificate_location: NEEDS_TO_BE_SET

    # Set the absolute path to the client certificate's
    # .CER file for communication with Contrast UI. The
    # `certificate_location` property must be set to `File`.
    # cer_file: NEEDS_TO_BE_SET

    # Specify the name of certificate store to open. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `AuthRoot`, `CertificateAuthority`,
    # `My`, `Root`, `TrustedPeople`, or `TrustedPublisher`.
    # store_name: NEEDS_TO_BE_SET

    # Specify the location of the certificate store. The
    # `certificate_location` property must be set to `Store`.
    # Value options include `CurrentUser` or `LocalMachine`.
    # store_location: NEEDS_TO_BE_SET

    # Specify the type of value the agent uses to find the certificate
    # in the collection of certificates from the certificate store.
    # The `certificate_location` property must be set to `Store`.
    # Value options include `FindByIssuerDistinguishedName`,
    # `FindByIssuerName`, `FindBySerialNumber`,
    # `FindBySubjectDistinguishedName`, `FindBySubjectKeyIdentifier`,
    # `FindBySubjectName`, or `FindByThumbprint`.
    # find_type: NEEDS_TO_BE_SET

    # Specify the value the agent uses in combination with
    # `find_type` to find a certification in the certificate store.
    #
    # Note - The agent will use the first certificate from
```

```
    # the certificate store that matches this search criteria.
    #
    # find_value: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

    # Set the proxy authentication type. Value
    # options are `NTLM`, `Digest`, and `Basic`.
    # auth_type: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
```

```
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # roll_size: 100

    # Set the number of backup files to keep. Set to `0` to disable.
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT
```

```
    # Set the log level of Blocked attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked: NOTICE

    # Set the log level of Blocked At Perimeter
    # attacks. Value options are `ALERT`, `CRITICAL`,
    # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked_perimeter: NOTICE

    # Set the log level of Probed attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_probed: WARNING

    # Set the log level of Suspicious attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_suspicious: WARNING

    # Set the connection type used for Syslog messages.
    # Value options are `UNENCRYPTED` and `ENCRYPTED`.
    # connection_type: UNENCRYPTED

  # \
================================================================================
  # agent.dotnet
  # The following properties apply to any .NET agent-wide configurations.
  # \
================================================================================
  # dotnet:

    # Set a list of application pool names that the agent does not
    # instrument or analyze. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_denylist: NEEDS_TO_BE_SET

    # Set a list of application pool names that the agent instruments or
    # analyzes. If set, other application pools are ignored. Allowlist takes
    # precedence over denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 4.0.2.
    # app_pool_allowlist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent does not
    # analyze. (The applications are still instrumented).
    # Names must be formatted as a comma-separated list.
    # New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_denylist: NEEDS_TO_BE_SET

    # Set a list of application names that the agent analyzes.
    # If set, other applications are not analyzed, but are
    # still instrumented. Allowlist takes precedence over
    # denylist. Names must be formatted as a comma-separated
    # list. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
    # application_allowlist: NEEDS_TO_BE_SET

    # Enable a profiler chaining feature to allow Contrast to
```

```
  # work alongside other tools that use the CLR Profiling
  # API. Defaults to `true`. New after .NET Framework 19.1.3
  # (Installed Only) and .NET Core 1.9.3 (Installed Only).
  # enable_chaining: true

  # Indicate that the agent should allow CLR optimizations
  # of JIT-compiled methods. Defaults to `true`. New
  # after .NET Framework 19.1.3 and .NET Core 1.0.0.
  # enable_instrumentation_optimizations: true

  # Indicate that the agent should allow the CLR to inline
  # methods that are not instrumented by Contrast. Defaults to
  # `true`. New after .NET Framework 19.1.3 and .NET Core 1.0.0.
  # enable_jit_inlining: true

  # Indicate that the agent should allow the CLR to perform
  # transparency checks under full trust. Defaults to `false`.
  # New after .NET Framework 19.1.3 and .NET Core 1.0.0.
  # enable_transparency_checks: false

  # Set to display ASCII art to std::out on agent startup. Defaults
  # to `true`. New after .NET Framework 20.6.3 and .NET Core 1.0.0.
  # enable_cat: true

  # Sets the maximum amount of time a Protect regular expression
  # is allowed to run before being cancelled. Set to -1 to never
  # cancel regular expression execution. Defaults to `20_000`.
  # New after .NET Framework 20.4.3 and .NET Core 1.5.0.
  # protect_searchers_single_pattern_deadline_ms: 20_000

  # Sets the maximum amount of time a 'Probe Analysis' Protect
  # regular expression is allowed to run before being cancelled. Set
  # to -1 to never cancel regular expression execution. Defaults to
  # `5_000`. New after .NET Framework 20.7.3 and .NET Core 1.5.11.
  # protect_searchers_probe_analysis_single_pattern_deadline_ms: 5_000

  # Sets the maximum amount of time a Protect rule is
  # allowed to run before being cancelled. Set to -1 to never
  # cancel Protect rule execution. Defaults to `60_000`.
  # New after .NET Framework 20.4.3 and .NET Core 1.5.0.
  # protect_searchers_total_rule_deadline_ms: 60_000

  # Sets the maximum amount of time a 'Probe Analysis' Protect
  # rule is allowed to run before being cancelled. Set to -1 to
  # never cancel Protect rule execution. Defaults to `10_000`.
  # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
  # protect_searchers_probe_analysis_total_rule_deadline_ms: 10_000

  # Sets the maximum duration of time agent log files should be kept
  # since last write before being deleted by the agent. Defaults to
  # `604_800_000`. New after .NET Framework 20.6.1 and .NET Core 1.5.5.
  # log_cleanup_maximum_age_ms: 604_800_000

  # Suppresses gathering process-level metrics (process level metrics are
  # gathered by default), used to identify performance problems. Metric
```

```
    # counters may further decrease the stability of already unstable
    # systems and can be disabled (set to true) if issues occur. Defaults
    # to `false`. New after .NET Framework 20.6.6 and .NET Core 1.5.10.
    # suppress_metric_counters: false

    # Enable file based application watching. Set to false if
    # file watching is causing locking issues. Defaults to `true`.
    # New after .NET Framework 20.7.3 and .NET Core 1.5.11.
    # enable_file_based_app_watching: true

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Specifies the cloud provider from which the agent should gather metadata
  # (such as resource identifiers). Options are `AWS`, `Azure`, or `GCP`.
  #
  # gather_metadata_via: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Control the values captured by Assess vulnerability events. `Full`
  # captures most values by calling ToString on objects, which can
  # provide more info but causes increased memory usage. `Minimal`
  # has better performance as it only captures String type objects
  # as strings and uses type name for other object type values.
  # event_detail: minimal
```

```
  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
```

```
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-chaining
    # Use the following properties to configure how the
    # sql injection semantic analysis chaining rule works.
```

```
    # \
================================================================================
    # sql-injection-semantic-chaining:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-dangerous-functions
    # Use the following properties to configure how the sql
    # injection semantic analysis dangerous functions rule works.
    # \
================================================================================
    # sql-injection-semantic-dangerous-functions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-suspicious-unions
    # Use the following properties to configure how the sql
    # injection semantic analysis suspicious unions rule works.
    # \
================================================================================
    # sql-injection-semantic-suspicious-unions:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.sql-injection-semantic-tautologies
    # Use the following properties to configure how the sql
    # injection semantic analysis tautologies rule works.
    # \
================================================================================
    # sql-injection-semantic-tautologies:

      # Set the mode of the rule. Value options
      # are `monitor`, `block` or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:
```

```
      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Tell the agent to detect when commands come directly
      # from input. The agent blocks if blocking is enabled.
      # detect_phased_commands: true

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
    # \
================================================================================
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
```

```
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
```

```
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
================================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
```

```
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
```

```
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Override the reported server path. New after
  # .NET Framework v21.3.1 and .NET Core v1.8.0.
  # path: NEEDS_TO_BE_SET
```

.NET エージェントエクスプローラ

重要

.NET Core エージェント 4.0.0 および.NET Framework エージェント 51.0.0 より、Contrast トレイアプリケーションを.NET エージェントエクスプローラに置き換えました。

.NET エージェントエクスプローラは、.NET Core エージェントおよび.NET Framework エージェントの稼働状況に関する概要情報を表示するアプリケーションです。このアプリケーションを使用すると、エージェントが想定通りに動作しているかを確認できます。特にエージェントを最初にインストールした後にご利用ください。

エージェントをインストールすると、このアプリケーションもインストールされます。両方の種類のエージェントをインストールした場合は、エージェントエクスプローラのインスタンスは 1 つだけインストールされます。

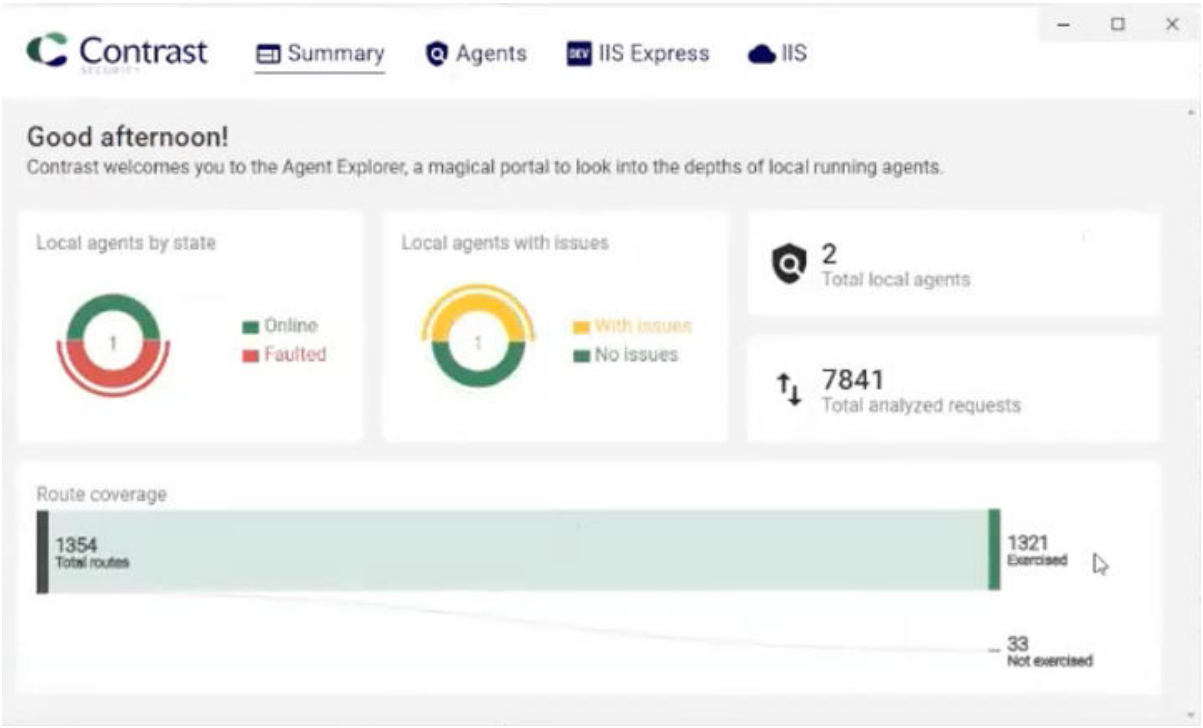
エージェントエクスプローラへのアクセス

.NET Core エージェントか.NET Framework エージェントをインストールした後、エージェントエクスプローラのアイコン( )がトレイに表示されます。アイコンを右クリックすると、アプリケーションが開きます。

エージェントエクスプローラでの情報

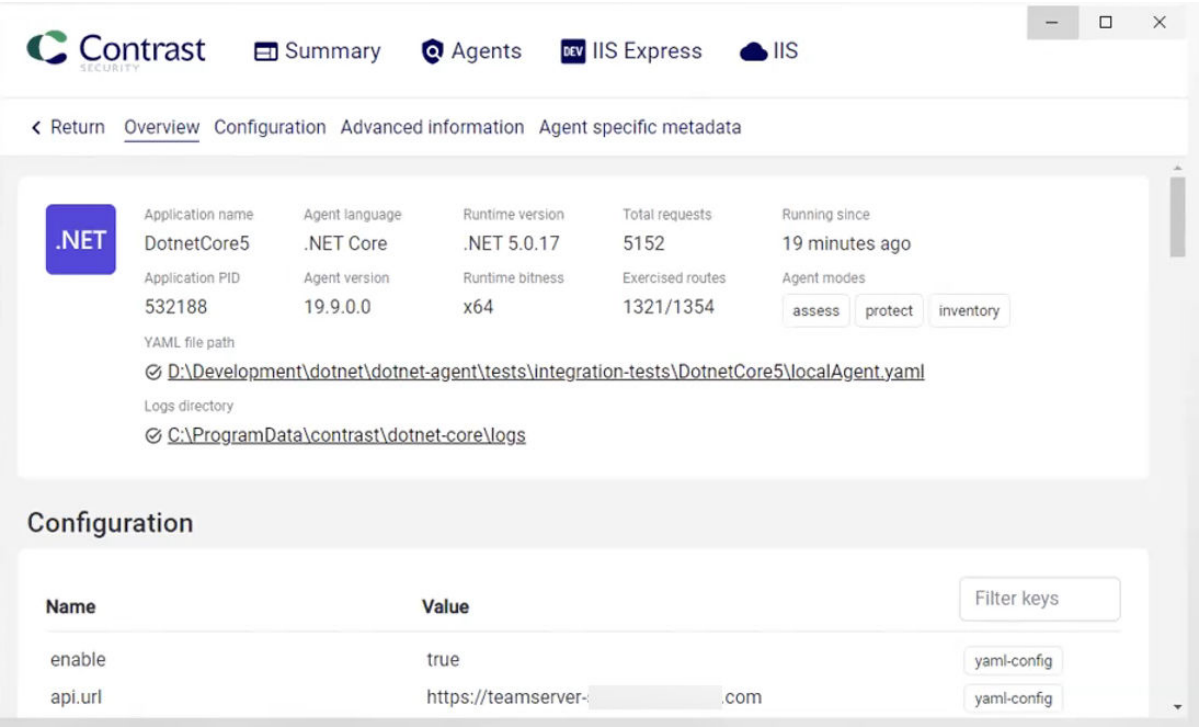
エージェントエクスプローラには、以下の情報が表示されます。

- **Summary(サマリー)**



このダッシュボードには、エージェントのステージ、問題のあるエージェントの有無、ルートカバレッジなど、エージェントに関する概要レベルの情報が表示されます。

- **Agents(エージェント)**



このタブには、.NET Core エージェントおよび.NET Framework エージェントの稼働状況に関する情報が表示されます。Configuration(設定)セクションには、特定の問題が発生していることをエージェントエクスプローラが検知した場合に、メッセージが表示されます。
Overview(概要)セクションにあるリンクから、エージェントの設定ファイル(YAML ファイル)に直接アクセスすることができます。下にスクロールすると、エージェントの設定内容、詳細情報、セッションメタデータの情報などが表示されます。

- **IIS Express**
  このタブには、IIS Express で実行されている Web アプリケーションの情報が表示されます。
- **IIS**
  このタブには、IIS サーバで実行されている Web アプリケーションの情報が表示されます。

.NET Core エージェントのプロファイラチェーン

プロファイラチェーンを使用して、.NET Core APM プロファイラと併せて.NET Core エージェントを実行できます。

Contrast .NET Core エージェントは、ランタイム、デプロイタイプ、OS の特定の組み合わせを前提として、以下のプロファイリングツールとの互換性をテストおよび検証済みです。

プロファイリングツール	検証済バージョン	.NET Core ランタイム	サードパーティプロファイラのデプロイタイプ	OS
AppDynamics	21.8.1	6.0	インストーラ使用、NuGet パッケージ	Windows
Dynatrace OneAgent	1.253.245	6.0	インストーラ使用	Windows、Linux
New Relic	8.23.107	6.0	NuGet パッケージ	Windows、Linux
リバーベッド SteelCentral Aternity	12.9.0	6.0	インストーラ使用	Windows
Datadog	2.35.0	6.0	インストーラ使用、NuGet パッケージ	Windows、Linux

注記
その他のプロファイリングツールでも、CoreCLR プロファイリング API の規則に準拠しており、プロファイリング環境に関する前提条件がない場合は、エージェントはそのプロファイリングツールと互換性がある可能性があります。

チェーンはデフォルトで有効になっており、`agent.dotnet.enable_chaining` を `false` に設定することで無効にできます。

```
agent:
   dotnet:
     enable_chaining: false
```

自動(Windows、IIS)

IIS 用.NET Core エージェントのインストーラを使用する場合､ホストされている全ての.NET Core アプリケーションに対して、インストーラが自動でチェーンを構成します。

1. サードパーティのエージェントを最初にインストールし(推奨)、それから Contrast .NET Core エージェントをインストールしてください。
2. IIS ワーカを再起動します(デフォルトでは、これはエージェントのインストーラによって自動的に行われます)。再起動すると、Contrast .NET Core エージェントは、IIS に登録されている他のプロファイリングツールの存在を自動的に検出し、Contrast .NET Core エージェントとサードパーティの両方のプロファイラを読み込むよう環境を構成します。

自動(Linux)

Linux では、`LD_PRELOAD` 環境変数を設定することで自動チェーンを設定することができます。

```
LD_PRELOAD=<path to the extracted Contrast files>/runtimes/linux-x64/native/
ContrastChainLoader.so
```

例えば、Contrast エージェントを`/contrast` に解凍した場合、以下のようにするとチェーンが自動的に設定されます。

```
LD_PRELOAD=/contrast/runtimes/linux-x64/native/ContrastChainLoader.so \
dotnet ./HelloWorld
```

APM プロファイラによっては、既に `LD_PRELOAD` が設定されているものもあります(例、 Dynatrace)。この場合、Contrast のモジュールが最初にロードされ、他の `LD_PRELOAD` はコロン`:`で区切られていることを確認してください。例：

```
LD_PRELOAD=/contrast/runtimes/linux-x64/native/ContrastChainLoader.so:/
<path to Dynatrace>/liboneagentproc.so dotnet ./HelloWorld
```

注記
Kubernetes の環境で実行する場合、Contrast エージェントオペレータ (514ページ)は、自動的にチェーンを設定するため、既存の Kubernete ワークロードに対してエージェントを追加するのに推奨する方法です。

手動

以下のような環境では、チェーンを手動で設定する必要があります。

- IIS 以外でホストされている
- (インストールされたエージェントではなく)サードパーティエージェントの Nuget パッケージを使用する環境

注記
Dynatrace エージェントとのチェーンは、上記の自動オプションを使用した場合のみにサポートされます。

1. プロファイリングツールの CLR 環境変数を CONTRAST_CCC_CORECLR バージョンに置き換えてください。以下の環境変数名はいずれも置き換える必要があります。

変更すべき変数名	変更後
CORECLR_PROFILER	CONTRAST_CCC_CORECLR_PROFILER
CORECLR_PROFILER_PATH	CONTRAST_CCC_CORECLR_PROFILER_PATH
CORECLR_PROFILER_PATH_32	CONTRAST_CCC_CORECLR_PROFILER_PATH_32
CORECLR_PROFILER_PATH_64	CONTRAST_CCC_CORECLR_PROFILER_PATH_64

2. そして、エージェントを手動 (227ページ)で追加します。

.NET Framework および.NET Core のテレメトリ

.NET Framework エージェントおよび.NET Core エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集され、その後も定期的(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切 (964ページ)であると考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、安全に Contrast に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	データ	
.NET Framework 2020.8.3 以降 .NET Core 1.5.15 以降	エージェントのバージョン	
	オペレーティングシステムとバージョン	
	エージェントがコンテナ内で実行されているかどうか	
	エージェントが Azure App Services で実行されているかどうか	
	ハッシュ化されたメディアアクセス制御(MAC)アドレス：暗号化(SHA256)された、匿名の一意なマシン ID	
	カーネルのバージョン	
	プロセスの実行時間	
	Assess が有効であるかどうか	
	Protect が有効であるかどうか	
	.NET Framework 2020.8.3 以降	.NET Framework ランタイムのバージョン
	.NET Core 1.5.15 以降	.NET Core ランタイムのバージョン

エージェントのバージョン	データ
.NET Framework 20.9.1 以降 .NET Core 1.5.17 以降	Contrast インスタンスが SaaS 版かオンプレミス版であるか
.NET Framework 20.9.3 以降 .NET Core 1.5.19 以降	CLR Instrumentation Engine (CIE)の使用状況 アプリケーションのフレームワーク 連携しているプロファイリングツール
.NET Framework 20.10.1 以降 .NET Core 1.5.20 以降	プロセスのホスティングモード CIE ネイティブプロファイラフックの使用状況
.NET Framework 20.10.2 以降 .NET Core 1.5.21 以降	デフォルト以外の値が指定されている設定名 無効にされている Assess ルール名
.NET Framework 20.12.2 以降 .NET Core 1.7.2 以降	エージェントのプロファイラコンポーネントが初期化されるまでに経過した時間 エージェントから Contrast Web インターフェイスへの最初のリクエストまでに経過した時間 エージェントのプロファイラコンポーネントが初期化されるまでに経過した時間 エージェントの初期化から最初のリクエストが終了するまでに経過した時間
.NET Framework 21.1.1 以降	IIS でホストされているアプリケーションに関する測定値： • 合計アプリケーション数 • 解析対象となるアプリケーション数(アプリケーションの許可/拒否リストで設定) • CLR4 のアプリケーションプールでホストされているアプリケーション数 • CLR2 のアプリケーションプールでホストされているアプリケーション数 IIS のアプリケーションプールに関する測定値： • 合計数 • エージェントが接続されている数 • CLR4 の数 • CLR2 の数 単独のアプリケーションプールでのアプリケーションの最小数 単独のアプリケーションプールでのアプリケーションの最大数 全てのアプリケーションプールにあるアプリケーション数の中央値
.NET Framework 21.1.2 以降 .NET Core 1.7.5 以降	Protect の各ルールのモード(例、監視やブロック)
.NET Framework 21.4.2 以降 .NET Core 1.8.4 以降	エージェントのセンサーコード内でスローされキャッチされた例外。ログメッセージ、例外タイプ、例外のメッセージ、`System` メソッドおよび `Contrast` メソッドのスタックトレースフレームなど。
.NET Framework21.7.1 以降 .NET Core1.9.7 以降	• プロセスアーキテクチャ(x86/x64) OS アーキテクチャ(x86/x64) Azure App Service における以下の環境変数の値： • `WEBSITE_PHYSICAL_MEMORY_MB` • `WEBSITE_PLATFORM_VERSION` • `WEBSITE_SKU`
.NET Framework21.9.2 以降 .NET Core2.0.1 以降	YAML 設定ファイルを読み込んだ場所(環境変数で指定したパス、デフォルトの場所、アプリケーションディレクトリなど)に関する説明

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、*telemetry.dotnet.contrastsecurity.com* に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

Azure Functions のサポート

バージョン

ランタイムバージョン	言語のバージョン	サポート対象	サポート対象外
1.x	.NET Framework 4.8	• Windows アプリケーション：Azure の.NET Framework 拡張機能でサポート	• Docker Linux イメージ • Linux アプリケーション
3.x	.NET 5	• Windows アプリケーション：ローカル、および.NET Core 拡張機能によって Azure でサポート • Docker Linux イメージ：ローカルおよび Azure でサポート	• Linux アプリケーション
4.x	.NET 6	• Windows アプリケーション：ローカル、および.NET Core 拡張機能によって Azure でサポート • Docker Linux イメージ：ローカルおよび Azure でサポート	• Linux アプリケーション

注記

すべてのバージョンで、Azure Functions アプリケーションを分離プロセスで実行している場合は、.NET エージェントのサポート対象外です。

サポート対象のトリガー

- HTTP
- Service Bus

設定

Azure Functions では、3 つのデプロイシナリオがサポートされています。Windows アプリケーション、Linux アプリケーション、Docker Linux イメージの 3 つです。この 3 つのうち、.NET エージェントが対応しているのは、Windows アプリケーションと Docker Linux イメージのみです。Linux アプリケーションのデプロイは、.NET エージェントではサポートされません。

Windows アプリケーション

Windows アプリケーションのデプロイは、Azure Functions のバージョン 1、3、4 に対して完全にサポートされます。ローカルでは、アプリケーションは Contrast .NET Core エージェントの NuGet パッケージを参照します。Azure 上には、Contrast .NET Core の拡張機能をインストールする必要があります。ツール(Visual Studio や Core Tools など)や、CI/CD パイプライン、または Azure Functions のポータルページのエディタを使用して、Azure Functions のアプリケーションをデプロイすると、サイト拡張機能は必要なアプリケーション設定を自動的に**設定しません**。アプリケーション設定を手動で指定する必要があります。

そのためには、以下のアプリケーション設定(設定 > 構成 > アプリケーション設定)を行ってください。

```
CORECLR_ENABLE_PROFILING=1
CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
CORECLR_PROFILER_PATH_32=C:\home\SiteExtensions\Contrast.NetCore.Azure.SiteExtension\ContrastNetCoreAppService-<version>\runtimes\win-x86\native\ContrastProfiler.dll
CORECLR_PROFILER_PATH_64=C:\home\SiteExtensions\Contrast.NetCore.Azure.SiteExtension\ContrastNetCoreAppService-<version>\runtimes\win-x64\native\ContrastProfiler.dll
```

上記の<version>の箇所には、"0.0.0.0"という形式でエージェントのバージョンを指定します。例えば、エージェントのバージョンが「2.1.8」であれば、パスは次のようになりま

す： `C:\home\SiteExtensions\Contrast.NetCore.Azure.SiteExtension\ContrastNetCoreAppService-2.1.8.0\runtimes\win-x64\native\ContrastProfiler.dll`

Contrast サーバへの接続情報は、アプリケーション設定を使用するか、アプリケーション設定から参照される設定ファイルで指定します。

アプリケーション設定 ：

```
CONTRAST__API__USER_NAME=my_username
CONTRAST__API__SERVICE_KEY=my_service_key
CONTRAST__API__API_KEY=my_api_key
CONTRAST__API__URL=my_api_url
```

設定ファイルで Contrast サーバの接続情報を指定する場合は、次のアプリケーション設定を指定する必要があります：`CONTRAST_CONFIG_PATH=C:\home\site\wwwroot\contrast_security.yaml`

Docker Linux イメージ

カスタム Linux イメージのデプロイは、Azure Functions のバージョン 3 および 4 でサポートされます。カスタム Linux イメージが必要で、イメージにはアプリケーションと Contrast .NET Core エージェントの NuGet パッケージが含まれている必要があります。Linux アプリケーションは、カスタム Linux イメージから実行されない場合、.NET エージェントではサポート対象外であることに注意してください。

エージェントを組み込むために、以下のアプリケーション設定を行う必要があります。 これらの設定は、イメージ自体(環境変数として)に含めるか、またはアプリケーション設定(設定 > 構成 > アプリケーション設定)として設定します。

```
CORECLR_ENABLE_PROFILING=1
CORECLR_PROFILER={8B2CE134-0948-48CA-A4B2-80DDAD9F5791}
CORECLR_PROFILER_PATH=/home/site/wwwroot/contrast/runtimes/linux-x64/native/
ContrastProfiler.so
CORECLR_PROFILER_PATH_64=/home/site/wwwroot/contrast/runtimes/linux-x64/
native/ContrastProfiler.so
CONTRAST_CORECLR_INSTALL_DIRECTORY=/home/site/wwwroot/bin/contrast/
```

Contrast サーバへの接続情報は、アプリケーション設定/環境変数を使用するか、アプリケーション設定/環境変数から参照される設定ファイルで指定します。

アプリケーション設定/環境変数：

```
CONTRAST__API__USER_NAME=my_username
CONTRAST__API__SERVICE_KEY=my_service_key
CONTRAST__API__API_KEY=my_api_key
CONTRAST__API__URL=my_api_url
```

設定ファイルで Contrast サーバの接続情報を指定する場合は、次のアプリケーション設定/環境変数を指定する必要があります： `CONTRAST_CONFIG_PATH=/home/site/wwwroot/contrast_security.yaml`

Node.js エージェント

Contrast Node.js エージェントは、トランスコンパイルなどの確立された技術を駆使して、Node.js の Web アプリケーションの動作を解析し、実行前にアプリケーションに Contrast のセンサーを追加します。

Contrast Node.js エージェントは、セマンティックバージョニング(major.minor.patch)に従います。エージェントは、こちらのサポート対象テクノロジ (285ページ)とシステム要件 (288ページ)で最適に動作します。

Node.js エージェントは、Babel コンパイラを使用してアプリケーションの起動前にアプリケーションコードを変換します。起動後、エージェントはサポート対象のフレームワークやモジュール (285ページ)に必要な関数をパッチします。

Node.js エージェントをインストール (289ページ)すると、アプリケーションの動作を監視するために、2 つの主要なソースコード変換が行われます。

- **AST 変換**は、エージェントによって作成されたコード本体の抽象的な構文ツリーを操作し、この構文ツリーに基づいて新しいソースコードを作成するプロセスです。エージェントはこのプロセスによって、関数フックが機能しない状況を処理します。例えば、変換によって Contrast は JavaScript に演算子のオーバーロードを追加できるため、信頼されないデータの流れを適切に追跡できるようになります。
- **関数フック**は、`child_process.exec` など特定の関数の実行を引き継ぎ、その引数や戻り値に関するデータを収集し、エージェントの解析部分にこのデータを送信します。その結果、エージェントは特定の関数について報告できるようになります。

Contrast サービス

Contrast サービス (494ページ)は、Node.js エージェント内にパッケージ化された実行可能ファイルで、別のプロセスで実行されます。エージェントのバージョン 4.x.x では、Contrast サービスはエージェントと一緒に自動的に起動します。

このサービスによって、Node.js エージェントと Contrast サーバ間の通信が可能になります。エージェントと同様に、環境変数 (66ページ)や YAML 設定ファイル (63ページ)で設定 (495ページ)できます。Contrast サービスは、エージェントと Contrast 間の HTTP 通信に、デフォルトで 30555 番ポートを使用します。

エージェントと Contrast 間のポートおよび通信プロトコルは設定できます。利用可能なプロトコルは、HTTP、Linux ソケット(ファイル記述子)、gRPC です。サービスは、エージェントと 1 対 1 でデプロイすることも、複数のコンテナをホストする単一のサーバ上のエージェントを 1 つのグループとして共有することもできます。

Node.js のサポート対象テクノロジ

このページは、注記で特に記載されていない限り、npmjs.com(npm 公式 Web サイト)で入手可能な最新バージョンのサポート対象テクノロジや機能を反映しています。

注記

Contrast Node.js エージェントは、npmjs.com で非推奨(deprecated)とタグ付けされたモジュールでは機能しない場合があります。非推奨のモジュールは、セキュリティ上のリスクが高く、エージェントの機能に悪影響を及ぼす可能性があります。

また、`webpack`、`parcel`、`esbuild` などのバンドラーを使用してサーバサイドの JavaScript コードをパッケージ化したり圧縮したりするアプリケーションもサポートしません。

言語のバージョン

• JavaScript ECMAScript 5 • JavaScript ECMAScript 6 • ECMAScript モジュール(ESM) • TypeScript	**注記** Contrast は、「アクティブ LTS」または「メンテナンス LTS」ステータスの Node.js の偶数番号バージョンをサポートします。 Node.js LTS バージョンは、JavaScript ECMAScript5 と 6 の機能をサポートします。 Contrast Node.js エージェントは、ESM を使用するアプリケーションで限定的なサポートを提供します。 TypeScript がサポートされるのは、エージェントがアプリケーションのコンパイル済みエントリポイントを指すように設定されている場合のみです。
Node.js LTS(長期サポート)版	
現在、アクティブおよびメンテナンス LTS ステータスの全てのバージョン： • 12*、14* • 16* (エージェントバージョン 4.5.0 以降のみ) • 18 (エージェントバージョン 4.25.0 以降)	**注記** 常にアクティブまたはメンテナンスステータスの Node.js の LTS バージョンを使用してください。 *12 LTS、14 LTS、16 LTS で、Contrast エージェントは機能しますが、それぞれ 2022 年 4 月末、2023 年 4 月末、2023 年 9 月 11 日に EOL(サポート期間終了)になっています。これらの EOL バージョンは、パッチが適用されなくなったため、深刻なセキュリティリスクがあります。また、エージェントのバージョン 5.x.x には、Node.js のバージョン 16.13.0 以上が必要であることにご注意ください。 Node.js エージェントは、実験的(Stability: 1)な実装に分類される Node.js の機能のサポートを保証しません。また、ネイティブな `net` モジュールにエージェントは組み込まれません。この表にあるサポート対象のアプリケーションフレームワークを使用して構築された `HTTP(S)` アプリケーションサーバにのみ機能します。 `HTTP/2` は、Node.js core の `HTTP/2` または `spdy` ライブラリが使用されている場合に、Node.js エージェントのサポート対象となります。 Contrast Node.js エージェントで `HTTP/2` を使用するアプリケーションでは、`assess.enable_lazy_tracking: false` を使用するようにエージェントを設定する必要があります。 Node.js のバージョンステータスは、Node.js の LTS リリーススケジュールに記載されています。
• 20 (エージェントバージョン 4.33.0 以降、エージェントバージョン 5.0.0-ベータ.2 以降のみ)	エージェントは、`--experimental-permission` でアクセスを制限して、アプリケーションを実行する機能をサポートしていません。アクセスを制限するとネイティブモジュールが非アクティブになり、権限を減らしてエージェントを組み込むと、すぐにクラッシュすることになります。
パッケージ管理システム(npm)	
`npm` バージョン： • 6.13.7 以降 • 7.11.0 以降 • 8.5.5 以降	Node.js エージェントが、Contrast Web インターフェイスに正確にライブラリを報告するためには、これらの npm バージョンのいずれかにアクセスする必要があります。バージョン 7 よりも、バージョン 6 または 8 を推奨します。
アプリケーションフレームワーク	
• Express 4 • hapi 16*、17*、18*、19*、20 • Fastify 3 • Koa 2.3 以降 • Kraken 2.2.0、2.3.0 • LoopBack 3*、4 • Restify 8 • Sails 1.2.3 以降	**注記** *これらのライブラリは保守管理者によって非推奨(deprecated)とされており、セキュリティ上のリスクの可能性があります。
データベースドライバとオブジェクト関係マッピング(ORM)	

• DynamoDB(Assess のみ) AWS SDK for JavaScript : 2.x、3.x • MongoDB 2.2.36*、3.3.0 以降、4.x(データベースバージョン 3.6、4.x、5.x に対応) • MySQL2 2.0.0 以降(MySQL データベースバージョン 5.6.51、5.7.x、8.0.x に対応) • Mongoose 5.x、6.x • MSSQL 6.4.0 以降 • Postgres driver 7.5.0 以降、8.x • RethinkDB ドライバ バージョン 2.4.0 以降 • Sequelize 5.x、6.x • SQLite3 driver 4.x(データベースバージョン 3.26.0 以降に対応)	**注記** *このバージョンではエージェントはまだ機能しますが、保守管理者によって非推奨(deprecated)とされており、セキュリティ上のリスクの可能性があります。
検証モジュール	
• Joi 17 以降 • Validator 13 以降 • Class-validator 0.13.0 以降	
テンプレートエンジン	
• Handlebars 4 • Pug 3 • EJS 2.6.2、3.0.1 • Mustache 4.x 以降	
その他のテクノロジ	
• Express-session 1.15.6、1.16.0 以降	
テストスイート	
Node Test Benches(Node テストベンチ)	Node.js エージェントに変更が加えられると、Contrast は一連の自動化テストを実行し、サポート対象のすべての Node バージョン内でサポート対象テクノロジでの検出結果を確認します。Node Test Benches には、Contrast のサポート対象のフレームワークでエージェントを実行するテストが含まれています。エージェントにサードパーティのライブラリサポートが追加されると、モノレポ(Monorepo)内の各フレームワークが更新されます。

Node.js エージェント 5.0.0 ベータ 仕様一覧

重要
このエージェントはベータ版です。ベータステータスでは、オプションが変更されたり、予期しない動作をする可能性があることを意味します。このベータ版を利用することで、お客様は Contrast ベータ版利用規約 (964ページ)に同意することになります。

このページは、備考で特に記載されていない限り、npmjs.com(npm 公式 Web サイト)で入手可能な最新バージョンのサポート対象テクノロジや機能を反映しています。

テクノロジ	サポート対象バージョン	備考
システム	• Node.js LTS バージョン 16、18、20* • プロセッサのサポート - Apple M1/M2、Intel/AMD(AMD64) • オペレーティングシステムのサポート - Windows Server、 Windows 10/11、MacOS、Linux (Debian、CentOS など) • PM2 • Node.js エージェントのシステム要件 (288ページ)	• * Node.js 20.5.0 以降のサポートを追加しました。
言語のバージョン	• Node.js のサポート対象テクノロジ (285ページ)	

テクノロジ	サポート対象バージョン	備考
NPM バージョン	• 8.5.5 以降	
アプリケーションフレームワーク	• Express 4 • Fastify 3、4 • Koa 2.3 以降	
データベースドライバとオブジェクト関係マッピング(ORM)	• MarsDB 現在はメンテナンスされていないが、脆弱な JuiceShop アプリケーションには必要。 • Mongoose 6.x、7.x • MongoDB 2.2.36、3.3.0 以降、4.x、5.x(データベースバージョン 4.x、5.x、6.x に対応) • MySQL2 2.0.0 以降(MySQL データベースバージョン 5.6.51、5.7.x、8.0.x に対応) • MSSQL 6.4.0 以降 • Postgres ドライバ 7.5.0 以降、8.x • Sequelize 5.x(こちらは保守管理者によって非推奨)、6.x • SQLite3 ドライバ 4.x(データベースバージョン 3.26.0 以降に対応)これは主に JuiceShop やデモアプリのためのものであり、SQLite は「本番用」のデータベースではありません。	• MongoDB 2.2.36 は、脆弱な NodeGoat アプリケーションで必要とされる場合のみ、サポートされています。 • SQLite と MarsDB は本番環境で使用するのではなく、脆弱な JuiceShop アプリケーションの実行やテストを可能にするためにのみサポートされています。
検証パッケージ/ライブラリ	• Class-validator 0.13.0 以降 • Joi 17 以降 • Validator 13 以降	
テンプレートエンジン	• Pug 3	
その他のパッケージ/ライブラリ	• Express-session 1.15.6、1.16.0 以降	

追加情報：

- このバージョンで、JuiceShop と NodeGoat のデモとテストができます。これらのアプリケーションの全ての依存関係は完全にサポートされています。
- 今後のベータ版で予定されている機能：
  - ソース：http/2
  - フレームワーク/パッケージ/ライブラリ：
    - hapi 19、20、21 バージョン 19.x 未満はサポートされません。19.x は保守管理者によって非推奨(deprecated)とされています。
    - DynamoDB(Assess のみ) AWS SDK for JavaScript：2.x、3.x
  - ルール：
    - Contrast Assess : XML 外部エンティティ処理(XXE)
  - 設定オプション：
    - `server.type` の設定
  - プロパゲータ：
    - ベータリリースでは対応していないプロパゲータを追加実装

Node.js エージェントのシステム要件

重要

Node.js エージェントは、M1/M2 チップを搭載した Mac での実行を制限付きでサポートするようになりました。制限としては、Apple M1/M2(ARM64)で `Alpine` ベースの Docker コンテナの実行は、まだ Node.js エージェントのサポート対象ではありません。`Slim` ベースの Docker イメージの実行はサポート対象です。

このページは、注記で特に記載されていない限り、npmjs.com(npm 公式 Web サイト)で入手可能な最新バージョンのシステム要件や機能を反映しています。

Node.js エージェントをインストールする前に、以下の要件を満たしていることを確認してください。

- 検査対象となるデプロイ済みアプリケーションには `package.json` ファイルがあり、その Web アプリケーションのテクノロジは Contrast のサポート対象であること。
- エージェントが Contrast サーバとネットワークで接続されていること。

Node.js エージェントを使用すると、受信情報の処理や解析が増加するため、アプリケーションで使用可能な CPU とメモリを増やす必要があります。Node.js エージェントを使用すると、アプリケーション単体で使用するよりも多くのリソースが必要となります。CPU の負荷も増加しますが、これはアプリケーションのアーキテクチャや既存の CPU 使用状況に大きく影響されます。

Assess を使用する場合、Contrast Node.js エージェントを使用しない場合と比較して、各コンテナで使用可能なメモリを 2 倍にする必要があります。

オペレーティングシステム

- CentOS/RHEL 7.9、8 以降
- Ubuntu 14.04 LTS、16.04 LTS、18.04 LTS、20.04 LTS、22.04 LTS
- Debian 9、10、11
- Windows
- macOS

プロセスマネージャ

- PM2：4.5.0 以降、5.1.0 以降
  - Contrast Node.js エージェントは、フォーク(fork)モードおよびクラスター(cluster)モードの両方の実行をサポートしています。
  - Contrast Node.js エージェントを実行する際に使用する起動コマンドに、インストール済の`@contrast/agent` モジュールへの完全なパスを指定する必要があります。例：

```
node -r /Users/michael/Dev/my-app/node_modules/@contrast/agent app.js
```

コンテナ

- Distroless コンテナ
  - Contrast Node.js エージェントが正しく機能するためには、`npm` と`/bin/sh` へアクセスする必要があります。これらのプログラム/パッケージを含まない Distroless コンテナイメージで実行するアプリケーションにインストールすると、エージェントの機能が低下し、検出結果やライブラリが報告されない場合があります。

CPU

- AMD 64、x86_64、およびその互換
- Apple M1/M2 ARM64 (限定サポート)

Node.js エージェントのインストール

エージェントのインストール方法はいくつかあり、ご利用の環境によって異なりますが、一般的には次のような流れになります。

1. Node.js エージェントを npm から取得します。
2. エージェントの認証キー (61ページ)を設定します。
3. エージェントを有効にしてアプリケーションを実行するために、`package.json` ファイルにコマンドを追加します。
4. エージェントを有効にしてアプリケーションを実行します。
5. 通常通りにアプリケーションを疎通し、Contrast でアプリケーションが認識されていることを確認します。

具体的な手順については、アプリケーションのデプロイ方法に応じて、以下の各手順に従ってください。

- 手動でのインストール (290ページ)
- コンテナでのインストール (290ページ)
- IBM Cloud でのインストール (296ページ)

Node.js エージェントを手動でインストール

エージェントを手動でインストールまたは更新するには：

1. アプリケーションのルートディレクトリで以下のコマンドを実行して、npm からエージェントの最新バージョンをインストールします。

```
npm install @contrast/agent
```

または、yarn を使用する場合、次のコマンドを実行してエージェントをインストールします。

```
yarn add @contrast/agent
```

> **注意**
>
> バージョン 4.x のエージェントの Protect 機能が、サポート終了(EOS)になりました。フル機能を備えたバージョン 5.x の Node.js エージェントが、2023 年後半にリリースされるまで、Node.js アプリケーションを保護する必要のあるユーザ向けに、*Protect のみ*の新しいエージェントがリリースされています。
>
> アプリケーションのルートディレクトリから次のコマンドを実行することで、npm から *Protect のみ*のエージェントの最新バージョンをインストールできます。
>
> ```
> npm install @contrast/protect-agent
> ```
>
> または、yarn を使用する場合、次のコマンドを実行してエージェントをインストールします。
>
> ```
> yarn add @contrast/protect-agent
> ```

2. 環境変数 (66ページ)を使用して、エージェントの認証キー (61ページ)を指定します。 Node.js アプリケーションが実行時に環境変数にアクセスできるようにしてください。
   または、こちらのテンプレート (305ページ)を使用して YAML 設定 (63ページ)ファイル (contrast_security.yaml)を作成し、認証キーを指定することもできます。
   `contrast_security.yaml` は、アプリケーションのルートディレクトリに作成してください。
3. アプリケーションの *package.json* ファイルの"scripts"セクションに次のコマンドを追加します。

```
"scripts": {
    "contrast": "node -r @contrast/agent <app-main>.js",
    "start": ...,
    "test": ...
}
```

4. エージェントを有効にしてアプリケーションを実行します。

```
npm run contrast
```

> **ヒント**
>
> 上記の npm スクリプトを変更して、別の設定ファイルの場所など実行時の他の設定 (304ページ)を指定することもできます。

5. 手動または自動テストを行ってアプリケーションを操作し、エージェントがインストールされた状態でアプリケーションが正しく機能することを確認してください。
6. アプリケーションサーバが Contrast に認識され、アプリケーションが Contrast に報告されていることを確認します。

コンテナを使用して Node.js エージェントをインストール

Node.js エージェントのコンテナへのインストールは、基本的に標準のインストール手順と同じですが、インストールがコンテナ内で行われる点と、ベストプラクティスに従い環境変数を使用して Contrast の認証情報を設定する必要がある点が異なります。

Node.js エージェントをコンテナにインストールする場合、環境変数を使用するのが最も安全な方法です。コンテナは QA や本番システムから移行されることが多いため、コンテナ定義に認証情報をハードコーディングしないことをお勧めします。

インストールを行う前に

本項では、Docker を例として、コンテナ化されたアプリケーションに Node.js エージェントをインストールするための一般的な手順について説明します。

コンテナや関連ソフトウェアの仕組みを基本的に理解している必要があります。実際の手順は、ご利用の環境に合わせて調整してください。

エージェントをインストール

以下のいずれかの方法で、Nodejs エージェントをインストールします。

- **アプリケーションの開発時にエージェントを追加** (推奨)
  この方法の場合、アプリケーションの `package.json` でエージェントを取り込みます。
  次のコマンドを使用して、パイプラインやコンテナイメージにエージェントを追加します。

```
npm install @contrast/agent
```

- **Dockerfile にエージェントを追加**
  アプリケーションのイメージを別々に管理したい場合(Contrast エージェントを使用する場合と使用しない場合)は、コンテナのビルド時にエージェントを追加します。
  次のコマンドを使用して、既存の Dockerfile、またはアプリケーションのイメージをベースイメージとして使用する新規の Dockerfile にエージェントを追加します。

```
npm install @contrast/agent
```

注意

バージョン 4.x のエージェントの Protect 機能が、サポート終了(EOS)になりました。フル機能を備えたバージョン 5.x の Node.js エージェントが、2023 年後半にリリースされるまで、Node.js アプリケーションを保護する必要のあるユーザ向けに、*Protect のみ*の新しいエージェントがリリースされています。

アプリケーションのルートディレクトリから次のコマンドを実行することで、npm から *Protect のみ*のエージェントの最新バージョンをインストールできます。

```
npm install @contrast/protect-agent
```

または、yarn を使用する場合、次のコマンドを実行してエージェントをインストールします。

```
yarn add @contrast/protect-agent
```

エージェントを設定

Node.js エージェントを Docker などのコンテナにデプロイしたアプリケーションで設定する場合は、以下の手順に従ってください(そうでない場合は、通常の Node.js エージェントの設定 (304ページ)を参照してください)。Node.j エージェントの設定には、設定した値が有効になる優先順位 (62ページ)があります。

- 環境変数を使用して、アプリケーション固有の設定値を指定します。これらは、Dockerfile 内で ENV 命令を定義するか、Docker run コマンドの実行時に -e オプションで指定することができます。アプリケーション固有の値を設定するためによく使用される環境変数の一覧 (305ページ)をご覧ください。また、Contrast エージェント設定エディタ (64ページ)で変数の一覧を参照することもできます。
  例えば、次のコマンドを使用して、コンテナを構築することができます。

```
docker build -t my-app-image
```

  そして、コンテナの実行時には次のコマンドを使用します。

```
docker run -p 3000:3000 --name my-app-instance \
-e "CONTRAST__API__URL=your-ts-url" \
```

```
-e "CONTRAST__API__API_KEY=your-api-key" \
-e "CONTRAST__API__SERVICE_KEY=your-service-key" \
-e "CONTRAST__API__USER_NAME=your-user-name" \
my-app-image
```

クラウドプロバイダの使用時に環境変数を設定するプロセスでは、一般的には、シークレットマネージャを使用し、それらのシークレットの値を環境変数にリンクすることになります。

実行して確認

1. Node.js エージェントのリライタ CLI(現在、バージョン 4.x のエージェントのみで利用可能)を使用する場合は、RUN 命令を追加して、`contrast-transpile` 実行ファイルを呼び出すコマンドを指定します。そして、アプリケーションのエントリポイントを指定します。

   ```
   RUN npx -p @contrast/agent --no contrast-transpile index.js
   ```

   npx コマンドが Contrast Security からのものであり、サプライチェーン攻撃を試みる悪意のある人物からではないことを確認するために、`-p @contrast/agent --no` オプションを使用することが重要です。
   `-p` は、`--package` の省略形で、`@contrast/agent` パッケージにのみコマンドを使用するよう **npx** に指定します。
   `--no` は、 非推奨となった`--no-install` オプションの新しいオプション名で、 コマンドのバイナリが見つからない場合に npm からインストールを行わないよう **npx** に指定します。
   npx コマンドを実行する前に、Contrast エージェントと **npx** バイナリが正常にインストールされていることを想定しています。
2. アプリケーションを起動する際に、Contrast エージェントを先にロードしておく必要があります。通常は、これは Dockerfile の CMD 文で行いますが、`package.json` に定義した npm スクリプトも使用できます。
   例えば、通常アプリケーションを次のように起動する場合：

   ```
   CMD [“node”, “app”]
   ```

   次のコマンドを使用すると、Contrast を有効にしてアプリケーションを実行できます。

   ```
   CMD [“node”, “-r”, “@contrast/agent”, “app”]
   ```
3. エージェントが起動すると、YAML 設定ファイルに記述されたエージェントの認証キー (61ページ)を使用して Contrast サーバに接続します。

   ヒント
   エージェントの認証情報を保護するために、Docker secret を利用して、デプロイ時に環境変数として渡すこともできます。例：

   ```
   docker run -e CONTRAST__API_ -e \
   CONTRAST__API__API_KEY=<value> -e \
   CONTRAST__API__SERVICE_KEY=<value> -e \
   CONTRAST__API__USER_NAME=<value> -e \
   CONTRAST__SERVER__ENVIRONMENT=<value> image_with_contrast
   ```
4. コンテナのログのアクティビティをチェックして、Contrast が実行されていることを確認します。例えば、ログアクティビティは以下のようになります。

   ```
   @contrast/agent 2.16.8-------------------------------------
   2020-07-20T19:05:14.407Z INFO contrast-service: BUILD \
   {"progname": "Contrast Service", "version": "2.8.1", "buildTime": ""}
   2020-07-20T19:05:14.407Z INFO Building timer for orphan request cleanup \
   {"progname": "Contrast Service", "cleanupMs": 5000}
   ```

```
2020-07-20T19:05:14.408Z INFO Building timer for orphan app cleanup \
{"progname": "Contrast Service", "time": 5000}
2020-07-20T19:05:14.450Z INFO Creating New Application Server \
{"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "serverName": "bc1bd6e5cd3a", "clientId": "1", "pid":
 1}
2020-07-20T19:05:14.450Z WARN Failed to initialize secure client, \
falling back to insecure client {"progname": "Contrast Service"}
2020-07-20T19:05:15.473Z INFO setting new server features for \
context{"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "serverName": "bc1bd6e5cd3a"}
2020-07-20T19:05:15.474Z ERROR Error setting up CEF syslog \
{"progname": "Contrast Service", "err": "open /juice-shop/security.log: \
permission denied"}
2020-07-20T19:05:15.475Z INFO starting event scanner \
{"progname": "Contrast Service", "report": {}}
2020-07-20T19:05:15.486Z INFO Creating new application \
{"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "serverName": "bc1bd6e5cd3a", "appName": "juiceshop-
guide", "language": "Node", "clientId": "1", "pid": 1}
2020-07-20T19:05:15.486Z INFO AppCreate: creating and initializing new \
application {"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "server_name": "bc1bd6e5cd3a", "app_name": "juiceshop
-guide", "app_lang": "Node", "client_id": "1", "pid": 1}
2020-07-20T19:05:15.921Z INFO setting new application settings \
{"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "serverName": "bc1bd6e5cd3a", "appName": "juiceshop-
guide", "language": "Node"}
2020-07-20T19:05:15.922Z INFO Setting session id on app context: \
{"progname": "Contrast Service", "uuid": "96299b72-f867-4354-
b9c9-1eb23511cb8a", "clientid": "1", "appname": "juiceshop-
guide", "applang": "Node", "apppath": "/juice-shop/
package.json", "sessionid": "cd0b271e66974162bf5fcca8b32e37b1"}
Entering main at /juice-shop/appinfo: All dependencies in ./
package.json are satisfied (OK)...
```

注記

また、Docker イメージの作成時にエージェントをインストール (293ページ)したり、distroless の Node.js コンテナ (295ページ)を使用してコンテナにインストールすることも可能です。

関連項目

Contrast サポートポータル：Kubernetes での Node.js エージェント

Contrast サポートポータル：AWS Fargate と Contrast エージェント

Docker イメージの作成時にエージェントをインストール

Node.js アプリケーションに Contrast エージェントをインストールする別の方法として、ソースコードリポジトリの *package.json* ファイルを変更する代わりに、`npm install` コマンドを Docker イメージ作成の一部として実行できます。

Docker ファイルのみを修正して、Contrast エージェントを使用してセキュリティ検査を実行できるようにしたい場合に適しています。

例：

```
FROM node:18 as installer
COPY . /juice-shop
WORKDIR /juice-shop
RUN npm i -g typescript ts-node
RUN npm install --omit=dev --unsafe-perm
RUN npm install @contrast/agent@4.x
RUN npm dedupe

# Neeed to explicitly set Assess mode
ENV CONTRAST__APPLICATION__NAME=juice-assess-docker-slim
ENV CONTRAST__ASSESS__ENABLE=true
ENV CONTRAST__AGENT__LOGGER__STDOUT=true
ENV CONTRAST__AGENT__LOGGER__PATH=/dev/null
ENV DEBUG="contrast:*"

ENV CONTRAST__AGENT__NODE__REWRITE_CACHE__PATH="/juice-shop/rewrite_cache"

RUN npx contrast-transpile build/app.js

RUN rm -rf frontend/node_modules
RUN rm -rf frontend/.angular
RUN rm -rf frontend/src/assets
RUN mkdir logs
RUN chgrp -R 0 ftp/ frontend/dist/ logs/ data/ i18n/
RUN chmod -R g=u ftp/ frontend/dist/ logs/ data/ i18n/
#RUN rm data/chatbot/botDefaultTrainingData.json || true
#RUN rm ftp/legal.md || true
#RUN rm i18n/*.json || true

FROM node:18-slim
ARG BUILD_DATE
ARG VCS_REF

WORKDIR /juice-shop
COPY --from=installer /juice-shop .

EXPOSE 3000
# The following environment variables were added
ENV CONTRAST__APPLICATION__NAME=juice-assess-docker-slim
ENV CONTRAST__AGENT__SERVICE__GRPC=true
ENV CONTRAST__AGENT__LOGGER__STDOUT=true
ENV CONTRAST__AGENT__LOGGER__PATH=/dev/null
ENV DEBUG="contrast:*"

ENV CONTRAST__AGENT__NODE__REWRITE_CACHE__PATH="/juice-shop/rewrite_cache"

# This explicitly turns on Assess mode
ENV CONTRAST__ASSESS__ENABLE=true
ENV CONTRAST__ASSESS__ENABLE_LAZY_TRACKING=false
ENV CONTRAST__AGENT__NODE__APP_ROOT=/juice-shop
```

```
CMD ["node", "-r", "@contrast/agent", "build/app.js"]
```

Distroless コンテナ

distroless の Node.js コンテナを使用している場合、コンテナイメージに **npm** や **shell** はインストールされていません。必須モジュールとしてエージェントを実行するために、NODE_OPTIONS 環境変数を指定する必要があります。

ただし、NODE_OPTIONS を使用すると、すべての **node** または **npm** コマンドでエージェントが実行され、意図しない実行によって起動時間が長くなる可能性があるため、注意が必要です。

例：

```
FROM node:18 as installer
COPY . /juice-shop
WORKDIR /juice-shop
RUN npm i -g typescript ts-node
RUN npm install --omit=dev --unsafe-perm
RUN npm install @contrast/agent@4.x

RUN npm dedupe

RUN rm -rf frontend/node_modules
RUN rm -rf frontend/.angular
RUN rm -rf frontend/src/assets
RUN mkdir logs
RUN chown -R 65532 logs
RUN chgrp -R 0 ftp/ frontend/dist/ logs/ data/ i18n/
RUN chmod -R g=u ftp/ frontend/dist/ logs/ data/ i18n/
RUN rm data/chatbot/botDefaultTrainingData.json || true
RUN rm ftp/legal.md || true
RUN rm i18n/*.json || true

FROM gcr.io/distroless/nodejs:18
ARG BUILD_DATE
ARG VCS_REF
LABEL maintainer="Bjoern Kimminich <bjoern.kimminich@owasp.org>" \
    org.opencontainers.image.title="OWASP Juice Shop" \
    org.opencontainers.image.description="Probably the most modern and \
sophisticated insecure web application" \
    org.opencontainers.image.authors="Bjoern Kimminich \
<bjoern.kimminich@owasp.org>" \
    org.opencontainers.image.vendor="Open Web Application Security \
Project" \
    org.opencontainers.image.documentation="https://help.owasp-juice.shop" \
    org.opencontainers.image.licenses="MIT" \
    org.opencontainers.image.version="14.5.1" \
    org.opencontainers.image.url="https://owasp-juice.shop" \
    org.opencontainers.image.source="https://github.com/juice-shop/juice-
shop" \
    org.opencontainers.image.revision=$VCS_REF \
    org.opencontainers.image.created=$BUILD_DATE
WORKDIR /juice-shop
COPY --from=installer --chown=65532:0 /juice-shop .
USER 65532
EXPOSE 3000
```

```
ENV NODE_OPTIONS "-r @contrast/agent"
CMD ["/juice-shop/build/app.js"]
```

Node.js エージェントを IBM Cloud でインストール

1. Node.js LTS(長期サポート)の最新版をインストールします。
2. エージェントを npm からインストールする場合は、アプリケーションのルートディレクトリで次のコマンドを実行します。

   ```
   npm install @contrast/agent
   ```

   または、yarn を使用する場合、次のコマンドを実行してエージェントをインストールします。

   ```
   yarn add @contrast/agent
   ```
3. YAML 設定ファイルを使用して Node.js エージェントを設定 (304ページ)し、エージェントの認証キー (61ページ)およびアプリケーション固有の設定を指定します。
   以下は、YAML 設定ファイル(*contrast_security.yaml*)のサンプルです。<URL>、<UserName>、<APIKey>、<ServiceKey>の箇所をご利用の認証情報に置き換え、<ServerName>にはアプリケーションがレポートする IBM Cloud のサーバ名を指定できます(このようにサーバ名を指定すると、Contrast Web インターフェイスで表示した際にサーバを特定できます)。

   ```
   contrast:
     url: <URL>
     user_name: <UserName>
     api_key: <APIKey>
     service_key: <ServiceKey>
   server:
     name: <ServerName>
   ```
4. アプリケーションのルートディレクトリに *contrast* という名前のフォルダを作成し、*node-contrast-.tgz* ファイルと *contrast_security.yaml* ファイルを *contrast* フォルダに移動します。
5. アプリケーションの *package.json* ファイルの"scripts": セクションに次のコマンドを追加します。

   ```
   "ibmcloud-with-contrast": "npm install @contrast/agent && node -r \
   @contrast/agent index.js -c /home/vcap/app/contrast/
   contrast_security.yaml",
   ```
6. IBM Cloud ではデフォルトで起動スクリプトが実行されるため、前述の手順で指定した ibmcloud-with-contrast の行で開始するよう起動コマンドを変更する必要があります。次のコマンドを使用して、エージェントを実行します。

   ```
   "start":"npm run bluemix-with-contrast"
   ```

   そして、package.json の"scripts"セクションは以下のようになるはずです。

   ```
   "scripts": {
   "bluemix-with-contrast": "npm install @contrast/agent && node -r \
   @contrast/agent index.js -c /home/vcap/app/contrast/
   contrast_security.yaml",
   "start":"npm run bluemix-with-contrast”
   },
   ```
7. 次のコマンドを使用して、アプリケーションを IBM Cloud にプッシュします。

   ```
   cf push <application-name> -t 180
   ```
8. 次のコマンドを使用して、エージェントを実行します。

   ```
   npm start contrast
   ```

9. 手動または自動テストを行なってアプリケーションを操作し、エージェントがインストールされた状態でアプリケーションが正しく機能することを確認してください。
10. アプリケーションサーバが Contrast に認識され、アプリケーションが Contrast に報告されていることを確認します。

VMware Tanzu で Node.js エージェントをインストール

VMare Tanzu(旧 Pivotal Cloud Foundry)では、デフォルトの Node.js ビルドパックを利用して、アプリケーションで Contrast と様々なインテグレーションができます。

ユーザー提供サービスを作成し、そのサービスをアプリケーションにバインドすることによって、独自のビルドパックをインテグレーションとして利用することができます。サービスブローカーを使用すれば、複数のサービスプランを定義し、アプリケーションにバインドできるサービスインスタンスを複数作成できます。

Contrast は Contrast サービスブローカータイル (298ページ)を提供しており、 BOSH 展開と Contrast サービスブローカー (300ページ)の設定を自動化します。

重要

Contrast と VMware Tanzu のインテグレーションでは、Node.js エージェントはダウンロードされず、アプリケーション起動も変更されません。Node.js エージェントをダウンロードして、手動でインストール (290ページ)する必要があります。

エージェントの設定は、インテグレーションで提供される Contrast サービスブローカータイルで行うこともできますし、ユーザー提供サービスで自動設定を使用することもできます。

ビルドパック

VMware Tanzu の環境に Node.js エージェントをインストールするには、以下のいずれかのビルドパックをアプリケーションで使用する必要があります。

- **タイル利用の場合：** NodeJS ビルドパックバージョン 1.6.52 以降
- **ユーザー提供サービス利用の場合：** NodeJS ビルドパックバージョン 1.6.56 以降

Contrast フレームワークのサポートが含まれていないビルドパックを使用する場合、フレームワークのサポートを追加できます。追加する場合は、フォークしたビルドバックで適切な変更が必要です。ビルドパックのオフライン版を使用している場合、アプリケーションで使用されているエージェントのバージョンは上書きされません。ビルドバック内に依存関係がバンドルされています。

Contrast フレームワークのサポート : Contrast エージェントのフレームワークにより、最新のエージェントが自動でダウンロードされ、設定ファイルが作成されます。ビルドパックの detect スクリプト(検出スクリプト)は、タグを標準出力に出力します。

設定

detect スクリプトは、1 つのバインドされた Contrast サービスが存在するかを確認します。Contrast サービスが存在すると判断される基準は、VCAP_SERVICES ペイロードに `contrast-security` を部分文字列として含むサービス名、ラベル、またはタグがある場合です。

ユーザー提供サービスを使用して Contrast をバインドする場合は、`contrast-security` を含む名前かタグが必要です。また、認証情報のペイロードに基本の YAML オプション (63ページ)も含める必要があります。

以下は、ユーザー提供サービスを作成し、アプリケーションにバインドする例です。

```
cf create-user-provided-service contrast-security-service -
p "teamserver_url, username, api_key, service_key"
cf bind-service spring-music contrast-security-service
cf restage spring-music
```

注記
`teamserver_url` の箇所には、プロトコルとホスト名のみを指定してください。`/Contrast/`や/Contrast/api は含めないでください。

関連項目
VMware Tanzu の Contrast サービスブローカータイルを追加 (298ページ)

VMware Tanzu の Contrast サービスブローカーを追加 (300ページ)(タイルなしでサービスブローカーを使用)

Node.js の Contrast サービスブローカータイルを追加
サービスブローカーを使用すると、Apps Manager やコマンドラインから VMware Tanzu(旧 Pivotal Cloud Foundry)のアプリケーションをサービスにバインドして、簡単にサービスを使用できるようになります。Contrast サービスブローカーは、VMware Tanzu 上の Node.js アプリケーションとしてデプロイすることができ、1 つ以上の Contrast アカウントを使用できます。ブローカーによって、Contrast サービスが VMware Tanzu マーケットプレイスに公開されるので、サービスのインスタンスを直接作成して、アプリケーションにバインドすることができます。

タイルを追加すると、**contrast-security-service-broker-org** という 1 つの組織が作成されます。この組織を使用して、Contrast サービスブローカーのアプリケーションをデプロイします。これには 512MB のメモリが必要です。

開始する前に
Contrast サービスブローカータイルを追加する前に、以下が必要です。

- Pivotal Apps Manager、Ops Manager
- 有効な Contrast のアカウント
- デフォルトの Node.js ビルドパック(Contrast を使用する全てのアプリケーションに必要)カスタムのビルドパックを使用している場合は、Contrast フレームワークのサポートと設定をビルドパックにコピーしておく必要があります。

手順
Node.js の Contrast サービスブローカータイルを追加するには：

1. VMware Tanzu Network から Contrast サービスブローカータイルをダウンロードします。
2. Ops Manager で **Import a Product**(プロダクトをインポート)ボタンを選択して、ダウンロードした *contrast-security-service-broker-#.#.#.pivotal* タイルを選択します。

   注記
   ダウンロードしたファイルの拡張子が ZIP の場合は、ファイル名を *contrast-security-service-broker-#.#.#.pivotal* に変更してください。

3. サービスプランを追加するには、Contrast サービスブローカータイルの **Service Plans**(サービスプラン)を選択し、**Add**(追加)ボタンをクリックします。タイルをデプロイするには、設定がいくつか必要です。デフォルトでは、サービスブローカーにはサービスプランがありません。Contrast サービスブローカータイルをデプロイする前に、少なくとも 1 つ追加する必要があります。

4. サービスプランで以下の設定パラメータを入力します。
   - **TeamServer**：Contrast サーバへの URL
   - **TeamServer Service Key**：組織の設定にあるサービスキー (61ページ)
   - **TeamServer API Key**：組織の設定にある API キー (61ページ)
   - **Organization UUID**：アプリケーションが存在する組織の組織 ID (61ページ)
   - **Username**：Contrast ユーザ名
   - **Plan Name**：Apps Manager で表示されるプラン名
   - **Proxy Host**：サービスブローカーが Contrast と通信するプロキシのホスト名
   - **Proxy Port**：プロキシのポート
   - **Proxy Username**：認証が必要な場合のプロキシのユーザ名
   - **Plan Password**： プロキシのパスワード
5. サービスプランを定義したら、**Save**(保存)を選択します。アプリケーションを別の組織に入れたい場合は、必要となる他のプランを定義してください。
6. ダッシュボードで **Apply Changes**(変更を適用)を選択します。完了までに時間がかかる場合があります。
7. ここで、Cloud Foundry CLI を使用してアプリケーションをバインドします。
8. プロジェクトをローカルディレクトリにクローンします。例：`Node/pcf/node-hello-world`
9. アプリケーションの `package.json` を更新して、Contrast Node.js エージェントを依存関係として追加します。例：`"@contrast/agent": "^4"`
10. `package.json` で"start"スクリプトを更新して、Contrast エージェントをアプリケーションに組み込みます。例：`"start": "node -r @contrast/agent app.js"`
11. アプリケーションを VMwareTanzu にプッシュします。例えば、`Node/pcf` ディレクトリから以下のように実行します。

```
cf push myAppNodeBroker -p node-hello-world \
    -b 'https://github.com/cloudfoundry/nodejs-buildpack.git'   \
    -t 180
```

12. そして、以下のコマンドを実行します。

```
cf service-access
```

出力例：

```
Getting service access as admin...
broker: contrast-security-service-broker
   service             plan     access   orgs
   contrast-security   apptwo   all
```

13. そして、以下のコマンドを実行します。

```
cf create-service contrast-security apptwo contrast
```

出力例：

```
Creating service instance contrast in org system / space apps as admin...
OK
```

14. そして、以下のコマンドを実行します。

```
cf services
```

出力例(現在バインドされているアプリケーションがないことに注意)：

```
Getting services in org system / space apps as admin...

name       service             plan     bound apps   last operation     \
broker                             upgrade available
contrast   contrast-security   apptwo                create succeeded   \
contrast-security-service-broker
```

15. 以下のコマンドを実行すると、サンプルアプリケーションがサービスにバインドされます。

```
cf bind-service myAppNodeBroker contrast
```

出力例：

```
Binding service contrast to app myAppNodeBroker in org system / space \
apps as admin...
OK
```

16. 以下を再度実行すると、アプリケーションがサービスにバインドされたことを確認できます。

```
cf services
```

出力例：

```
Getting services in org system / space apps as admin...

name       service              plan     bound apps        last \
operation     broker                                upgrade available
contrast   contrast-security   apptwo   myAppNodeBroker   create \
succeeded   contrast-security-service-broker
```

17. 以下のコマンドを実行して、サービスにバインドされたアプリケーションを再ステージングします。

```
cf restage myAppNodeBroker
```

18. Contrast に移動して、アプリケーションを表示します。

関連項目

Contrast サービスブローカーを追加 (300ページ)

Node.js の Contrast サービスブローカーを追加

Contrast サービスブローカーを使用することで、VMware Tanzu(旧 Pivotal Cloud Foundry)のアプリケーションにサービスを簡単にバインドでき、Contrast Node.js エージェントを使用できます。

手順

VMware Tanzu を設定するには：

1. サポートにお問合せください。
2. サービスブローカーのソースコードを入手したら、サービスブローカーアプリケーションをデプロイします。

```
cf push contrast-security-service-broker
```

これでサービスブローカーが PCF に表示されるようになります。

3. CONTRAST_SERVICE_PLANS 環境変数を使用して、プランを設定します(デフォルトでは、サービスブローカーにはプランがありません)。
Pivotal Ops Manager を使用して、環境変数を設定することもできます。IBM Cloud を使用している場合、アプリケーションのコンソールページで **Runtime** を選択したら **Environment Variables** を選択し、値を設定します。
例：コマンドラインから値を設定するには、以下の例を参考にしてください。

```
cf set-env contrast-security-service-broker CONTRAST_SERVICE_PLANS
    " {
           "ServicePlan1": {
                "name":"ServicePlan1",
                "teamserver_url":"https://yourteamserverurl.com",
                "username":"your_username",
```

```
                "org_uuid":"00000000-1111-2222-3333-000000000000",
                "api_key":"your_api_key",
                "service_key":"your_service_key"
            },
            "AnotherServicePlan":{
                "name":"AnotherServicePlan",
                "teamserver_url":"https://yourteamserverurl.com",
                "username":"your_username",
                "org_uuid":"00000000-1111-2222-3333-000000000001",
                "api_key":"your_api_key",
                "service_key":"some_other_service_key"
                }
            } "
```

IBM Cloud でエージェントを実行する場合、CONTRAST_SERVICE_PLANS 環境変数の値は一重引用符で囲んでください。例：

```
    cf set-env contrast-security-service-broker CONTRAST_SERVICE_PLANS
    " {
            'ServicePlan1': {
                'name':'ServicePlan1',
                'teamserver_url':'https://yourteamserverurl.com',
                'username':'your_username',
                'org_uuid':'00000000-1111-2222-3333-000000000000',
                'api_key':'your_api_key',
                'service_key':'your_service_key'
            },
            'AnotherServicePlan':{
                'name':'AnotherServicePlan',
                'teamserver_url':'https://yourteamserverurl.com',
                'username':'your_username',
                'org_uuid':'00000000-1111-2222-3333-000000000000',
                'api_key':'your_api_key',
                'service_key':'some_other_service_key'
                }
            } "
```

4. アプリケーションを再ステージングするために、以下のコマンドを実行します。

```
cf restage contrast-security-service-broker
```

5. ユーザ名とパスワードの環境変数を設定します。

```
cf set-env contrast-security-service-broker SECURITY_USER_NAME \
aSecureUsername
cf set-env contrast-security-service-broker SECURITY_USER_PASSWORD \
aSecurePassword
```

6. サービスブローカーのインスタンスを作成します。サービスプランを少なくとも 1 つ定義してください。ユーザ名とパスワードは、前の手順で設定したのと同じものを使用する必要があります。

```
cf create-service-broker contrast-security-service-broker USER_NAME \
PASSWORD
<URL of your application>
```

IBM Cloud では、コマンドの最後に--space-scoped を追加してください。例：

```
cf create-service-broker contrast-security-service-broker USER_NAME \
PASSWORD
<URL of your application> --space-scoped
```

7. 全てのサービスブローカーは最初はプライベートです。そのためパブリックに変更します。

```
cf enable-service-access contrast-security-service-broker
```

8. サービスブローカーが動作するようになったので、サービスのインスタンスを作成し、アプリケーションにバインドします。サービスのインスタンスを作成するには、以下のコマンドを実行します。

```
cf create-service contrast-security-service-broker ServicePlan1 \
<name_of_service>
```

9. サービスブローカーをアプリケーションにバインドするために、以下のコマンドを実行します。

```
cf bind-service <app_name> <name_of_service>
```

アプリケーションでエージェントが起動されるのを確認できるはずです。Contrast Web インターフェイスにアプリケーションが表示されるはずです。

関連項目

Contrast サービスブローカータイルを追加 (298ページ)

Node.js エージェントのアップデート

Contrast の Node.js エージェントを自動的に更新するのに最も信頼性が高く効果的な方法は、Node.js の npm パッケージマネージャを使用し、利用可能な最新バージョンをダウンロードしてインストールすることです。

npm は通常、Node.js アプリケーションのすべての依存関係を管理するため、すでにビルド環境の一部として利用可能になっているはずです。Node.js エージェントを更新する頻度と更新の取得先は、組織の設定と Contrast の実装(SaaS 版またはオンプレミス版)によって異なります。

エージェントのアップデートは、手動または自動のどちらでも行うことができます。

開始する前に

設定を開始する前に、以下が必要です。

- DevOps の実践と Node の npm パッケージマネージャにある程度の知識があること。
- Contrast エージェントの npm リポジトリへアクセスできること。
- Node.js アプリケーションが Contrast エージェントなしで想定どおりに機能すること。
- 事前に Contrast の Node.js エージェントが正常にインストールされていること。
- 変更管理ポリシーと使用する環境に基づいて、エージェントをアップデートする方法とタイミングを決めていること。

重要

Contrast サポートからのアドバイスがない限り、Contrast インスタンスで利用できるバージョンよりも先行したバージョンの Node.js エージェントを使用しないでください。

手順

1. Node.js エージェントを npm のパブリック(またはプライベート)リポジトリからインストールします。Contrast の利用状況に応じて、いずれかまたは両方のソースを使用して最新の Node.js エージェントを取得します。

- **SaaS 版 :** npm からエージェントの最新バージョンを入手します。 エージェントを使用する前に検証を必要とする場合、承認されたバージョンのみがあるプライベートの npm リポジトリを使用することもできます。
- **オンプレミス(EOP)版：**M オンプレミス版を利用しているお客様の多くが、Contrast で新しいソフトウェアがリリースされても、コアソフトウェアやエージェントをすぐには更新しません。通常、パブリックリポジトリ(npm など)は、新しいバージョンのエージェントを提供しますが、新しいバージョンのエージェントは、古いバージョンの Contrast で動作するように設計やテストされていません。オンプレミス版をご利用の場合は、インストールしたオンプレミス版の Contrast と一致するエージェントのバージョンのみを保存するプライベート npm リポジトリから、エージェントの更新を取得してください。

2. エージェントをインストールし、スクリプトにより最適な方法で自動アップデートを行います。
   - **`package.json` を使う：**このファイルには、npm(パブリックまたはプライベート)からのアーティファクトを使用して、Node.js アプリケーションをビルドするたびに自動的に解決する依存関係を指定します。ここに Contrast の Node.js エージェントを含めることで、アプリケーションの新しいビルドを常に最新バージョンのエージェントに合わせることが簡単にできます。例 :

```
{
  "name": "sample_application",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "nodemon",
    "contrast": "node -r @contrast/agent index.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1",
    "@contrast/agent": "latest",
  },
  "devDependencies": {
    "nodemon": "^1.19.2"
  }
}
```

   そして、アプリケーションをビルドする際には常に`$ npm update` コマンドを使用します。これにより、Node.js エージェントが npm から Node.js アプリケーションに自動的にダウンロードされ、追加または更新されます。
   - **コマンドラインによる手動インストールおよび更新：**組織によっては、`package.json` ファイルが環境間で一貫している必要がある場合や、すべての環境に Contrast の Node.js エージェントをインストールする予定がない場合があります。このような場合、手動でエージェントをインストールします。Node.js のビルドプロセスの一部として手動でエージェントを更新できます。
     次のコマンドを使用して、Contrast の Node.js エージェントを手動で取得し、npm(パブリックまたはプライベート)から Node.js アプリケーションに追加または更新します。

```
$ npm install @contrast/agent
```

3. いずれの方法でインストールしても、インストール後に以下のような出力が表示されます。

```
$ npm install @contrast/agent

> grpc@1.24.4 install /Users/<aUserName>/Documents/test-apps/juice-shop/
node_modules/grpc> node-pre-gyp install --fallback-to-build --
library=static_library
```

```
node-pre-gyp WARN Using request for node-pre-gyp https download[grpc] \
Success: "/Users/<aUserName>/Documents/test-apps/juice-shop/node_modules/
grpc/src/node/extension_binary/node-v72-darwin-x64-unknown/
grpc_node.node" is installed via remotenpm WARN jest-config@26.6.1 \
requires a peer of ts-node@>=9.0.0 but none is installed. You must \
install peer dependencies yourself.npm WARN jsdom@16.4.0 requires a \
peer of canvas@^2.5.0 but none is installed. You must install peer \
dependencies yourself.npm WARN ws@7.3.1 requires a peer of \
bufferutil@^4.0.1 but none is installed. You must install peer \
dependencies yourself.npm WARN ws@7.3.1 requires a peer of utf-8-
validate@^5.0.2 but none is installed. You must install peer \
dependencies yourself.

+ @contrast/agent@3.4.0added 19 packages from 43 contributors, updated \
5 packages and audited 1995 packages in 14.904sfound 19 vulnerabilities \
(5 low, 7 moderate, 4 high, 3 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
```

4. インストール/アップデートが成功したかどうかを確認するには、次のコマンドを実行して、出力結果を確認してください。

```
$ npm list | grep contrast
 @contrast/agent@3.4.0
  @contrast/distringuish-prebuilt@2.0.0
  @contrast/escodegen@1.16.0
  @contrast/esprima@4.1.1
  @contrast/estraverse@5.1.0
  @contrast/flat@4.2.0
  @contrast/fn-inspect@2.3.0
  @contrast/heapdump@1.0.0
  @contrast/protobuf-api@2.2.3
  @contrast/require-hook@1.1.2
  @contrast/synchronous-source-maps@1.1.0
```

関連項目

- Node.js のサポート対象テクノロジ (285ページ)
- Node.js エージェントのインストール (289ページ)

Node.js エージェントの設定

全てのエージェントには基本の設定 (46ページ)があり、設定値には優先順位 (62ページ)があります。

Node.js エージェントを設定する方法はいくつかありますが、一般的には以下のようになります。

- 組織やコンテナ内の全てのアプリケーションに共通する設定値(例えば、ログのリダイレクトやプロキシの設定など)を指定するには、YAML 設定ファイルを使用します。こちらのテンプレート (305ページ)には、Node.js エージェントで有効な設定オプションが全てあります。YAML 設定については、こちらの説明 (63ページ)を参照してください。
- エージェントの認証キーやアプリケーション固有の設定値を指定するには、環境変数 (305ページ)を使用します。環境変数については、こちらの説明 (66ページ)を参照してください。

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。 エディタでは、必要に応じて適切な環境変数も表示されます。

環境変数

アプリケーション固有の設定値(サーバの環境、アプリケーション名、エージェントログの設定など)には、環境変数を使用します。また、Node.js エージェントで有効なその他のプロパティの設定にも環境変数を使用できます。

有効なプロパティの全リストは Node.js エージェントの YAML テンプレート (305ページ)にありますが、 ここではよく使用される環境変数をいくつか紹介します。

環境変数	説明
CONTRAST__API__SERVICE_KEY	Contrast との通信に必要なサービスキーを設定します。
CONTRAST__API__API_KEY	Contrast との通信に必要な API キーを設定します。
CONTRAST__API__USER_NAME	Contrast との通信に必要なユーザ名を設定します。
CONTRAST__API__URL	Contrast Web インターフェイスの URL を設定します。
CONTRAST__APPLICATION__NAME	Contrast に報告されるアプリケーション名を設定します。
CONTRAST_CONFIG_PATH	設定すると、YAML 設定ファイルのデフォルトの場所を上書きします。(他の環境変数とは異なり、この環境変数は YAML プロパティとして設定することはできず、アンダースコアは 1 つだけです。)
CONTRAST__SERVER__PATH	Contrast に報告されるサーバのパスを上書きします。
CONTRAST__SERVER__NAME	コンテナ化されたアプリケーションで多数のサーバレコードが生成される場合に、一貫したサーバ名を指定します。マイクロサービス名やアプリ名になることが考えられます。
CONTRAST__AGENT__DIAGNOSTICS__ENABLE	診断やトラブルシューティングの情報を追跡するために、起動時に設定ファイルとシステムファイルを作成します。デフォルトは、`true` です。
CONTRAST__AGENT__LOGGER__APPEND	`false` に設定すると、毎日ログファイルを追加してローテーションする代わりに、起動時に新しいログファイルを作成します。デフォルトは、`true` です。
CONTRAST__AGENT__LOGGER__LEVEL	ログのレベル : `FATAL`ERROR`WARN`INFO`DEBUG`TRACE` のいずれかを設定します。デフォルトは、`ERROR` です。
CONTRAST__AGENT__LOGGER__PATH	Contrast のデバッグログを保存するパスを指定します。デフォルトは、*node-contrast.log* です。
CONTRAST__AGENT__LOGGER__STDOUT	`false` に設定すると、`(stdout)`へ出力されなくなります。デフォルトは、`true` です。

注記

Node.js エージェントでは、環境変数 DEBUG に手動の設定が必要です。環境変数 DEBUG に Contrast の名前空間を含むように設定(DEBUG=contrast:*)しない限り、INFO レベルのメッセージはコンソールには記録されません。名前空間を操作して、特定のパスの表示/非表示を切り替えることもできます。これは、ファイルシステムにアクセスできない環境(Docker や ECS など)で有用です。

Node.js のログをリダイレクトしたい場合は、npm のサイトの他の例を参考にするか、サポートまでお問い合わせください。

Node.js エージェントの YAML テンプレート

YAML 設定ファイルを使用して Node.js エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、こちらの説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します : `/etc/contrast/contrast_security.yaml`

```
# \
================================================================================
==
```

```
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the default request timeout.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
```

```
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

    # If the Key file requires a password, it can be set here or in
    # the matching ENV value (`CONTRAST__CERTIFICATE__KEY_PASSWORD`).
    # key_password: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # Set to limit the length of Error stack traces to a specified number.
  # stack_trace_limit: 10

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
```

```
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # Set the directory in which to write diagnostic files.
    # Defaults to the application's current working directory.
    # report_path: ./

  # \
================================================================================
  # agent.effective_config
  # None
  # \
================================================================================
  # effective_config:

    # \
================================================================================
    # agent.effective_config.reporting
    # None
    # \
================================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
```

```
    # level: INFO

    # Set to `false` for the agent to always create a
    # new log file instead of appending and rolling.
    # append: true

    # Set to `false` to suppress log output to `stdout`.
    # stdout: true

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # This feature is only available in agent version >=4.0.0
    # roll_size: 100M

    # Set the number of backup files to keep. Set to `0` to disable.
    # This feature is only available in agent version >=4.0.0
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Set to `true` to log output to `stdout` as well as the configured \
file.
    # stdout: false

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
```

```
    # ip: NEEDS_TO_BE_SET

    # Set the port of the Syslog server to
    # which the agent should send messages.
    # port: NEEDS_TO_BE_SET

    # Set the facility code of the messages the agent sends to Syslog.
    # facility: 19

    # Set the log level of Exploited attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_exploited: ALERT

    # Set the log level of Blocked attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked: NOTICE

    # Set the log level of Blocked At Perimeter
    # attacks. Value options are `ALERT`, `CRITICAL`,
    # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_blocked_perimeter: NOTICE

    # Set the log level of Probed attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_probed: WARNING

    # Set the log level of Suspicious attacks. Value options are `ALERT`,
    # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
    # severity_suspicious: WARNING

  # \
================================================================================
  # agent.service
  # The following properties are used by the Contrast Service.
  # \
================================================================================
  # service:

    # Set to `false` to disallow the service to be started, and
    # effectively disable the agent, if read by the service. If the
    # agent reads this property, it disallows service auto-start.
    # enable: true

    # Set to `true` to enable listening for gRPC connections.
    # The `socket`, `host` and `port` fields will be used for
    # configuring the gRPC server in place of the legacy RPC server.
    # grpc: false

    # If this property is defined, the service is
    # listening on a Unix socket at the defined path.
    # socket: /tmp/service.sock

    # ********************** REQUIRED **********************
    # Set the the hostname or IP address of the Contrast
    # service to which the Contrast agent should report.
```

```
    host: localhost

    # ********************** REQUIRED **********************
    # Set the the port of the Contrast service
    # to which the Contrast agent should report.
    port: 30555

    # \
================================================================================
    # agent.service.teamserver_retry
    # The following properties are used by the Teamserver HTTP client
    # to configure failed request retrying in the Contrast service.
    # \
================================================================================
    # teamserver_retry:

      # Enable retrying HTTP requests to the Teamserver endpoint.
      # enable: true

      # How long to wait between retries in milliseconds.
      # interval_ms: 5000

      # How many times to retry HTTP requests to Teamserver before giving \
up.
      # max_attempts: 3

    # \
================================================================================
    # agent.service.logger
    # The following properties are used by the logger in the
    # Contrast service. If the properties are not defined, the
    # service uses the logging values from the Contrast UI.
    # \
================================================================================
    # logger:

      # Set the location to which the Contrast service saves log output.
      # If no log file exists at this location, the service creates one.
      #
      # Example - `/opt/Contrast/contrast_service.log` will
      # create a log in the `/opt/Contrast` directory.
      #
      # path: ./contrast_service.log

      # Set the the log output level. Options are `OFF`, `FATAL`,
      # `ERROR`, `WARN`, `INFO`, `DEBUG`, `TRACE`, and `ALL`.
      # level: ERROR

      # Override the name of the process used in logs.
      # progname: Contrast Service

      # Set to `true` to send log output to `stdout`.
      # stdout: false

  # \
```

```
# ============================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
  # \
# ============================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

  # \
# ============================================================================
  # agent.node
  # The following properties apply to any Node configurations.
  # \
# ============================================================================
  # node:

    # Set the directory containing the application's `package.json` file.
    # app_root: NEEDS_TO_BE_SET

    # \
# ============================================================================
    # agent.node.rewrite_cache
    # Use the following properties to set up rewrite caching in the agent.
    # \
# ============================================================================
    # rewrite_cache:

      # Set to `true` to enable rewrite caching.
      # enable: false

      # Set the location of the rewrite cache source.
      # path: NEEDS_TO_BE_SET

    # \
# ============================================================================
    # agent.node.library_usage
```

```
    # Configuration for Node.js library usage reporting
    # \
================================================================================
    # library_usage:

      # Set to `true` to have Contrast report
      # libraries that npm has marked extraneous.
      # read_extraneous_libraries: false

      # \
================================================================================
      # agent.node.library_usage.reporting
      # Use the following properties to set
      # up enhanced library usage reporting.
      # \
================================================================================
      # reporting:

        # Set to `false` to disable enhanced library usage features, i.e.
        # scanning for composition of dependencies, reporting library usage.
        # enable: true

        # Set the interval (in milliseconds) for
        # collecting code events for library usage.
        # interval_ms: 1

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:
```

```
  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # Set to `true` to trust incoming strings when they pass custom
  # validators (Mongoose, Joi, validator, fastify-static).
  # This feature is only available in agent version 4.10.0 and later
  # trust_custom_validators: false

  # Enables the serve-static module as a path-traversal
  # sanitizer. Express uses serve-static in a safe way
  # but manual setup of serve-static can be vulnerable.
  #
  # Even with Express there is a possibility for "traversing-down" the \
served
  # folder or user misconfiguration if not configured with an absolute path
  #
  # This feature is only available in agent version 4.31.0 and later
  # enable_sanitizer_serve_static: false

  # When set to `true`, string tracking will occur lazily as user-controlled
  # values are accessed by application code. When `false`, tracking will
  # occur at the time of input parsing and will be limited to 250 values.
  # This feature is only available in agent version 4.18.0 and later
  # enable_lazy_tracking: true

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

# \
================================================================================
==
# protect
```

```
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # Support for this option is limited to Node agent versions >= 5
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
```

```
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
    # \
================================================================================
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
```

```
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
```

```
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
================================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
```

```
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.ssjs-injection
    # Use the following properties to configure
    # how the SSJS Injection rule works.
    # \
================================================================================
    # ssjs-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.nosql-injection
    # Use the following properties to configure
    # how the NOSQL Injection rule works.
    # \
================================================================================
    # nosql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.nosql-injection-mongo
    # Use the following properties to configure
    # how the NOSQL Injection rule works.
```

```
    # \
================================================================================
    # nosql-injection-mongo:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Pass arguments to the underlying application.
  # args: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
```

```
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET
```

```
  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
==========================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
==========================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
==========================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
==========================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # Set the default request timeout.
  # timeout_ms: NEEDS_TO_BE_SET

  # \
```

```
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

    # If the Key file requires a password, it can be set here or in
    # the matching ENV value (`CONTRAST__CERTIFICATE__KEY_PASSWORD`).
    # key_password: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
```

```
# agent:

  # Set to limit the length of Error stack traces to a specified number.
  # stack_trace_limit: 10

  # \
================================================================================
  # agent.diagnostics
  # Use the properties in this section to specify the information the agent
  # should collect and report in order to diagnose problems in the agent.
  #
  # \
================================================================================
  # diagnostics:

    # Set to `false` to disable agent diagnostics
    # enable: true

    # Set the directory in which to write diagnostic files.
    # Defaults to the application's current working directory.
    # report_path: ./

  # \
================================================================================
  # agent.effective_config
  # None
  # \
================================================================================
  # effective_config:

    # \
================================================================================
    # agent.effective_config.reporting
    # None
    # \
================================================================================
    # reporting:

      # Defaults to `true`. Controls whether configuration
      # setting reports are sent to the Contrast web interface.
      # enable: true

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
```

```
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `false` for the agent to always create a
    # new log file instead of appending and rolling.
    # append: true

    # Set to `false` to suppress log output to `stdout`.
    # stdout: true

    # Set the roll size for log files in megabytes. The agent will
    # attempt to prevent the log file from being larger than this size.
    # This feature is only available in agent version >=4.0.0
    # roll_size: 100M

    # Set the number of backup files to keep. Set to `0` to disable.
    # This feature is only available in agent version >=4.0.0
    # backups: 10

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Set to `true` to log output to `stdout` as well as the configured \
file.
    # stdout: false

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
```

```
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.service
  # The following properties are used by the Contrast Service.
  # \
================================================================================
  # service:

    # Set to `false` to disallow the service to be started, and
    # effectively disable the agent, if read by the service. If the
    # agent reads this property, it disallows service auto-start.
    # enable: true

    # Set to `true` to enable listening for gRPC connections.
    # The `socket`, `host` and `port` fields will be used for
```

```
    # configuring the gRPC server in place of the legacy RPC server.
    # grpc: false

    # If this property is defined, the service is
    # listening on a Unix socket at the defined path.
    # socket: /tmp/service.sock

    # ********************** REQUIRED **********************
    # Set the the hostname or IP address of the Contrast
    # service to which the Contrast agent should report.
    host: localhost

    # ********************** REQUIRED **********************
    # Set the the port of the Contrast service
    # to which the Contrast agent should report.
    port: 30555

    # \
================================================================================
    # agent.service.teamserver_retry
    # The following properties are used by the Teamserver HTTP client
    # to configure failed request retrying in the Contrast service.
    # \
================================================================================
    # teamserver_retry:

      # Enable retrying HTTP requests to the Teamserver endpoint.
      # enable: true

      # How long to wait between retries in milliseconds.
      # interval_ms: 5000

      # How many times to retry HTTP requests to Teamserver before giving \
up.
      # max_attempts: 3

    # \
================================================================================
    # agent.service.logger
    # The following properties are used by the logger in the
    # Contrast service. If the properties are not defined, the
    # service uses the logging values from the Contrast UI.
    # \
================================================================================
    # logger:

      # Set the location to which the Contrast service saves log output.
      # If no log file exists at this location, the service creates one.
      #
      # Example - `/opt/Contrast/contrast_service.log` will
      # create a log in the `/opt/Contrast` directory.
      #
      # path: ./contrast_service.log

      # Set the the log output level. Options are `OFF`, `FATAL`,
```

```
      # `ERROR`, `WARN`, `INFO`, `DEBUG`, `TRACE`, and `ALL`.
      # level: ERROR

      # Override the name of the process used in logs.
      # progname: Contrast Service

      # Set to `true` to send log output to `stdout`.
      # stdout: false

  # \
================================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
  # \
================================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

  # \
================================================================================
  # agent.node
  # The following properties apply to any Node configurations.
  # \
================================================================================
  # node:

    # Set the directory containing the application's `package.json` file.
    # app_root: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.node.rewrite_cache
    # Use the following properties to set up rewrite caching in the agent.
    # \
================================================================================
    # rewrite_cache:
```

```
      # Set to `true` to enable rewrite caching.
      # enable: false

      # Set the location of the rewrite cache source.
      # path: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.node.library_usage
    # Configuration for Node.js library usage reporting
    # \
================================================================================
    # library_usage:

      # Set to `true` to have Contrast report
      # libraries that npm has marked extraneous.
      # read_extraneous_libraries: false

      # \
================================================================================
      # agent.node.library_usage.reporting
      # Use the following properties to set
      # up enhanced library usage reporting.
      # \
================================================================================
      # reporting:

        # Set to `false` to disable enhanced library usage features, i.e.
        # scanning for composition of dependencies, reporting library usage.
        # enable: true

        # Set the interval (in milliseconds) for
        # collecting code events for library usage.
        # interval_ms: 1

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # Set to `true` to trust incoming strings when they pass custom
  # validators (Mongoose, Joi, validator, fastify-static).
  # This feature is only available in agent version 4.10.0 and later
  # trust_custom_validators: false

  # Enables the serve-static module as a path-traversal
  # sanitizer. Express uses serve-static in a safe way
  # but manual setup of serve-static can be vulnerable.
  #
  # Even with Express there is a possibility for "traversing-down" the \
served
  # folder or user misconfiguration if not configured with an absolute path
  #
  # This feature is only available in agent version 4.31.0 and later
  # enable_sanitizer_serve_static: false

  # When set to `true`, string tracking will occur lazily as user-controlled
  # values are accessed by application code. When `false`, tracking will
  # occur at the time of input parsing and will be limited to 250 values.
  # This feature is only available in agent version 4.18.0 and later
  # enable_lazy_tracking: true

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
```

```
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # Support for this option is limited to Node agent versions >= 5
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:
```

```
      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-chained-commands
    # Use the following properties to configure how the
    # 'command injection - chained commands' rule works
    # \
================================================================================
    # cmd-injection-semantic-chained-commands:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-semantic-dangerous-paths
    # Use the following properties to configure how the
    # 'command injection - dangerous paths' rule works
```

```
    # \
================================================================================
    # cmd-injection-semantic-dangerous-paths:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection-command-backdoors
    # Use the following properties to configure how the
    # 'command injection - command backdoors' rule works
    # \
================================================================================
    # cmd-injection-command-backdoors:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal-semantic-file-security-bypass
    # Use the following properties to configure how the
    # 'path traversal - file security bypass' rule works
    # \
================================================================================
    # path-traversal-semantic-file-security-bypass:

      # Set the mode of the rule. Value options
      # are `monitor`, `block`, or `off`.
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
```

```
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.unsafe-file-upload
    # Use the following properties to configure
    # how the unsafe file upload rule works.
    # \
================================================================================
    # unsafe-file-upload:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:
```

```
      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.untrusted-deserialization
    # Use the following properties to configure
    # how the untrusted deserialization rule works.
    # \
================================================================================
    # untrusted-deserialization:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.ssjs-injection
    # Use the following properties to configure
    # how the SSJS Injection rule works.
    # \
================================================================================
    # ssjs-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.nosql-injection
    # Use the following properties to configure
    # how the NOSQL Injection rule works.
    # \
================================================================================
    # nosql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
```

```
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.nosql-injection-mongo
    # Use the following properties to configure
    # how the NOSQL Injection rule works.
    # \
================================================================================
    # nosql-injection-mongo:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET
```

```
  # Pass arguments to the underlying application.
  # args: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
```

```
# by the Contrast UI. This allows the user to configure the
# environment dynamically at startup rather than manually
# updating the Server in the Contrast UI themselves afterwards.
#
# Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
# For example, `PRODUCTION` registers this Server as
# running in a `PRODUCTION` environment, regardless of the
# organization's default environment in the Contrast UI.
#
# environment: NEEDS_TO_BE_SET

# Apply a list of labels to the server. Labels
# must be formatted as a comma-delimited list.
# Example - `label1,label2,label3`
#
# tags: NEEDS_TO_BE_SET
```

コンテナの起動時間を短縮する

Contrast エージェントをインストールしたアプリケーションを起動すると、アプリケーションおよびアプリケーションがロードする依存関係のあるコードの両方にコード変換が適用されます。このコード変換が行われるため、Contrast エージェントを使用したアプリケーションの起動時間が長くなります。

Node.js エージェントのバージョン 4.x 以降、アプリケーションを起動する前にプリコンパイルできるコマンドラインユーティリティを提供するようになりました。Contrast エージェントを有効にしてプリコンパイルされたアプリケーションを起動すると、変換済みファイルがディスクからロードされて起動時間が大幅に改善されます。

リライタ CLI を使用する

1. Node.js エージェントの設定ファイル(`contrast_security.yaml`)で以下を行います。
   - リライタのキャッシュを有効にし、キャッシュのパスを指定します。
   - ログレベルを指定します。

   > **注記**
   > CLI 変数または環境変数 (305ページ)を指定してリライタを起動して、YAML 設定をオーバーライドすることができます。例えば、標準出力(stdout)やディスクへのログ出力を防ぐ場合に、次の様にオーバーライドします。
   >
   > ```
   > npx -p @contrast/agent --no contrast-transpile index.js --
   > agent.logger.stdout false --agent.logger.path /dev/null;
   > ```
   >
   > **npx** コマンドが Contrast Security からのものであり、サプライチェーン攻撃を試みる悪意のある人物からではないことを確認するために、`-p @contrast/agent --no` オプションを使用することが重要です。
   >
   > - `-p` は、`--package` の省略形で、`@contrast/agent` パッケージにのみコマンドを使用するよう **npx** に指定します。
   > - `--no` は、 非推奨となった`--no-install` オプションの新しいオプション名で、コマンドのバイナリが見つからない場合に npm からインストールを行わないよう **npx** に指定します。
   >
   > **npx** コマンドを実行する前に、Contrast エージェントと **npx** バイナリが正常にインストールされていることを想定しています。

   - また、Protect か Assess を明示的に有効にする必要があります。

例：

```
agent:
  logger:
    level: info
    path: ./node-contrast
  node:
    rewrite_cache:
      enable: true
      path: ./rewrite_cache
assess:
  enable: true
protect:
  enable: false
```

2. アプリケーションのエントリポイント(例えば、`index.js`)を指定して、実行ファイルを起動します。例：

```
npx -p @contrast/agent --no contrast-transpile index.js --
agent.logger.level trace;
...
trace: 2021-07-12T18:31:15.128Z 2934603 contrast:rewrite - successfully \
rewrote code for /path/to/app/index.js
trace: 2021-07-12T18:31:15.186Z 2934603 contrast:rewrite - successfully \
rewrote code for /path/to/app/node_modules/koa/lib/application.js
trace: 2021-07-12T18:31:15.251Z 2934603 contrast:rewrite - successfully \
rewrote code for /path/to/app/node_modules/koa-router/lib/router.js
trace: 2021-07-12T18:31:15.314Z 2934603 contrast:rewrite - successfully \
rewrote code for /path/to/app/node_modules/@koa/router/lib/router.js
...
info: 2021-07-12T18:31:41.608Z 2946030 contrast:cli-rewriter - \
rewriting complete [26.625s]
```

注記
現在、Node.js エージェントのリライタ CLI は、Assess を有効にして実行するアプリケーションのプリコンパイルのみをサポートしています。Protect と Assess のどちらが有効であるかによって、エージェントがソースコードを変換する方法は異なります。Protect では、Assess よりも必要なソース変換が少なく、同じ起動遅延が発生することはありません。

3. リライタ処理が完了したら、通常と同じように Contrast を有効にしてアプリケーションを起動します。(例、`node -r @contrast/agent index.js`)

ESM で Node.js エージェントを使用する

Contrast Node.js エージェントは、 Node.js のサーバサイドアプリケーションで ECMAScript モジュール(ESM)を使用する場合に限定的なサポートを提供します。ESM は JavaScript クライアントサイドのコードをパッケージ化するための標準仕様で、現在 Node.js は ESM をサポートしています。

ESM のサーバサイドアプリケーションを使用する場合、Contrast Node.js エージェントを実行する方法は 2 つあります。

- 実行中のコードが ESM であり、ESM として実行される必要があることを明示的にするために、MJS ファイル拡張子を使用します。ファイル拡張子によるモジュールシステムの決定についての詳細は、Node.js のドキュメントを参照してください。

MJS 拡張子を使用すると、Node.js は ESM が記述されていることを認識し、JavaScript は ESM として解析されます。CJS についても同様で、Node.js は CJS ファイル拡張子を CommonJS として実行する必要があることを認識し、 JavaScript は CommonJS として解析されます。

- もしくは、次のように package.json に `"type": "module"`を指定することで、CommonJS ではなく ESM として Node.js アプリケーションを実行できます。

```
"main": "index.js",
"type": "module",
```

これにより、Node.js は `package.json` 下の JS ファイルを ESM として解析するようになります。それ以外の場合、デフォルトでは(または、`"type": "commonjs"`指定している場合)、Node.js は JS ファイルを CommonJS として解析します。

注記

Contrast Node.js エージェントは、ESM の条件付きエクスポート(Conditional Exports)をサポートしていません。

ECMAScript でのエージェントによる検査は試験的であり、Contrast でサポートするには少なくとも Node.js のバージョン 14.15.0 が必要です。

ESM(または ESM と CJS の組み合わせ)を使用するアプリケーションに Contrast エージェントを組み込む場合は、次のようにアプリケーションを起動します。

```
  Usage: node --experimental-loader @contrast/agent/esm.mjs
app-main.mjs [agent arguments] -- [app arguments]
```

トランスパイラ、コンパイラ、ソースマップおよび Node.js エージェント

Node.js エージェントは、TypeScript など JavaScript にコンパイルされる言語で書かれたアプリケーションをサポートします。Node.js エージェントは、JavaScript のみに組み込まれますが、アプリケーションを JavaScript にコンパイルするようトランスパイラを設定した場合、TypeScript も使用できます。

注記

結果として生成される JavaScript とソースが一致しない場合があります。そのため、報告されるメタデータ(やなど)は、元のソースではなくコンパイルした結果を参照します。

TypeScript など一部の言語では、ランタイム前にコードのプリコンパイルが必要なものがあります。このような場合、Node.js エージェントは、アプリケーションのコンパイル済エントリポイントを指す必要があります。

これを行うには、-r オプションを使用して Node.js エージェントを設定します。

```
scripts: {
    "test": "...",
    "start": "...",
    "contrast": "node -r @contrast/agent /path/to/transpiled/entrypoint.js"
}
```

ソースマップ

ソースマップを使用すると、TypeScript のソースとトランスパイルされた JavaScript 間で対応する行番号を確認できます。

ソースマップを使用するには、YAML 設定で Babel 変換(enable_babel)と変換キャッシュ(rewrite_cache)を有効にする必要があります。

```
agent:
  node:
    rewrite_cache:
      enable: true
      path: ./cache
    enable_babel: true
```

有効にすると、エージェントはロードされているソースと同じディレクトリにあるソースマップ(map ファイル)を検索します(例：`/home/app/index.js` ファイルがロードされると、`/home/app/index.js.map` が検索されます)。

ソースマップがまだない場合は、ソースマップを生成するために必要なフラグを使用して再コンパイルする必要があります。これはトランスパイラによって異なりますので、お使いのトランスパイラのオプションを確認してください。例えば、TypeScript の場合は、`--source-map` フラグを TypeScript コンパイラ(tsc)に追加するか、`tsconfig.json` の"`compilerOptions`"セクションに"`sourceMap`": `true` エントリを追加します。

Node.js のテレメトリ

Node.js エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集されます。現在は、起動時のみデータが送信されます。将来的には、エージェントを組み込んだアプリケーションが実行され、疎通されているときに、エージェントからエラーや測定値が送信される予定です。

弊社では、お客様のプライバシーは非常に大切である (964ページ)と考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、Contrast に安全に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	データ
`@contrast/agent` 4.12.0 以降	エージェントのバージョン
	オペレーティングシステムとバージョン
	Node.js のバージョン
	アプリケーションがコンテナで実行されているか(Y/N)

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、http://telemetry.nodejs.contrastsecurity.com に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

PHP エージェント

Contrast PHP エージェントは、PHP の Web アプリケーションを実行時に解析し、ライブラリの使用状況や脆弱性を検出します。PHP エージェントは、PHP の拡張モジュールとして実装されます。

注記
PHP エージェントは、現在、Assess と SCA のみに対応しています。

次のステップ：

- PHP エージェントをインストール (342ページ)
- PHP エージェントのシステム要件を確認 (342ページ)
- PHP エージェントのサポート対象テクノロジを確認 (342ページ)

PHP エージェントのサポート対象テクノロジ

このエージェントでは、以下のテクノロジをサポートしています。

テクノロジ	サポート対象バージョン	備考
言語のバージョン	• 7.4、8.0、8.1、8.2	エージェントは、`mbstring` および `curl` 拡張モジュールに依存します。 NTS 版の PHP のみをサポート。
オペレーティングシステム	• Debian • RHEL/CentOS	
アプリケーションフレームワーク	• Laravel • Symfony • Drupal 8、9	
プロセッサのアーキテクチャ	• AMD64	
サーバ	• Apache	その他、`mod_php` や `php-fpm` を使用しているサーバでも動作する可能性がありますが、現在はサポートされていません。
パッケージ管理	• Composer	SCA でサポートされています。

PHP エージェントのシステム要件

PHP エージェントをインストールする前に、以下のシステム要件を満たす必要があります。

要件	バージョン	備考
ランタイムシステム	• 64 ビット Linux	

PHP エージェントのインストール

PHP エージェントの基本的なインストールは、次のようになります。

1. エージェントパッケージをインストールします。
2. `contrast_security.yaml` をダウンロードし、適切なパスに置きます。
3. Contrast エージェント拡張機能を有効にするよう PHP インタプリタを設定します。
4. アプリケーションの疎通やテストを行います。
5. Contrast でアプリケーションが認識されていることを確認します。

具体的なインストール手順については、ご利用の環境に合わせて以下より選択してください。

- Debian で PHP エージェントをインストール (342ページ)
- RPM で PHP エージェントをインストール (344ページ)

Debian で PHP エージェントをインストール

手順

PHP エージェントをインストールするには：

1. https://pkg.contrastsecurity.com からエージェントパッケージを取得します。以下のコマンドで、Contrast のパッケージリポジトリをシステムに登録します。

```
curl \
  https://pkg.contrastsecurity.com/api/gpg/key/public | sudo apt-key \
add -

echo "deb https://pkg.contrastsecurity.com/debian-public/ $(sed -rne 's/
^VERSION_CODENAME=(.*)$/\1/p' /etc/*ease) contrast" \
  | sudo tee /etc/apt/sources.list.d/contrast.list

echo "deb https://pkg.contrastsecurity.com/debian-public/ all contrast" \
  | sudo tee -a /etc/apt/sources.list.d/contrast.list
```

2. 完了したら、以下のコマンドでエージェントをインストールします。

```
sudo apt-get update && sudo apt-get install contrast-php-agent
```

> **注記**
> PHP がこの拡張モジュールを使用するように設定されると、 インタープリタが実行されるたびにこの拡張モジュールが使用されるようになります。このステップは、アプリケーション自体を実行する直前まで遅らせる必要があります。これにより、artisan や他の PHP コマンドの実行時に予期せぬ動作が発生するのを防ぐことができます。

3. PHP の設定ファイルである php.ini を探します。ほとんどの場合、/usr/local/etc/php/内にあります。
   php-config コマンドを使用できる場合は、php-config --ini-path を使用して設定ファイルのパスを検索できます。このパスに設定ファイルがまだ存在しない場合は、作成する必要があります。
4. PHP 設定ファイルを編集します。

```
echo "extension=/usr/local/lib/contrast/php/contrast.so" >> `php-
config --ini-path`/php.ini
```

5. PHP エージェントの YAML テンプレート (345ページ)か環境変数を使用して、PHP エージェントを設定 (345ページ)します。
6. 通常の方法でアプリケーションを開始します。
7. アプリケーションの疎通やテストを行います。
8. PHP エージェントが実行されているかを確認するには、Contrast Web インターフェイスを確認するか、(設定によっては)PHP エージェントのログ出力を確認するなどしてください。

注記

- コマンドラインにフラグを追加することで、PHP CLI を使ってエージェントを使用することが可能です。

```
php -d extension=/usr/local/lib/contrast/php/contrast.so
```

- 現在、ライブラリ解析とルート検出は、アプリケーションへの最初のリクエストで実行されます。そのため、最初のリクエストは、それ以降のリクエストに比べてかなり遅くなることが予想されます。
- このエージェントは、サードパーティの PHP 拡張モジュールとはテストされていません。他のサードパーティの拡張モジュール(xdebug、APM などを含む)が使用された場合のエージェントの動作は不確定です。
- プリロードを有効にすると、エージェントが正しく動作しない場合があります。 このエージェントを使用する場合は、プリロードを無効にすることをお勧めします。
- デフォルトでは、エージェントは PHP アプリケーションを実行する際のサーバの作業ディレクトリが、アプリケーションのソースツリーのトップレベルのディレクトリと同じであると想定していま

す。エージェントは、このパスを使用して、ライブラリ解析とルート検出を行います。パスが異なる場合は、application.path の設定を使用して、アプリケーションのトップレベルのディレクトリを指定する必要があります。

RPM(Red Hat Package Manager)で PHP エージェントをインストール

手順

PHP エージェントをインストールするには：

1. `https://pkg.contrastsecurity.com` からエージェントパッケージを取得します。以下のスクリプトをシェルで実行し、お使いの RPM ベースのシステムに Contrast のパッケージリポジトリを設定します。`sudo` の権限が必要になる場合があります。

```
tee /etc/yum.repos.d/contrast.repo <<-"EOF"
[contrast]
name=Contrast centos-$releasever repo
baseurl=https://pkg.contrastsecurity.com/rpm-public/redhat-$releasever/
gpgcheck=0
enabled=1
EOF
```

2. 完了したら、以下のコマンドでエージェントをインストールします。

```
sudo yum install contrast-php-agent
```

> **注記**
> PHP がこの拡張モジュールを使用するように設定されると、インタープリタが実行されるたびにこの拡張モジュールが使用されるようになります。このステップは、アプリケーション自体を実行する直前まで遅らせる必要があります。これにより、artisan や他の PHP コマンドの実行時に予期せぬ動作が発生するのを防ぐことができます。

3. PHP の設定ファイルである `php.ini` を探します。ほとんどの場合、`/usr/local/etc/php/`内にあります。
   `php-config` コマンドを使用できる場合は `php-config --ini-path` を使用して設定ファイルのパスを検索できます。このパスに設定ファイルがまだ存在しない場合は、作成する必要があります。
4. PHP 設定ファイルを編集します。

```
echo "extension=/usr/local/lib/contrast/php/contrast.so" >> `php-
config --ini-path`/php.ini
```

5. PHP エージェントの YAML テンプレート (345ページ)か環境変数を使用して、PHP エージェントを設定 (345ページ)します。
6. 通常の方法でアプリケーションを開始します。
7. アプリケーションの疎通やテストを行います。
8. PHP エージェントが実行されているかを確認するには、Contrast Web インターフェイスを確認するか、(設定によっては)PHP エージェントの口グ出力を確認するなどしてください。

注記

- コマンドラインにフラグを追加することで、PHP CLI を使ってエージェントを使用することが可能です。

```
php -d extension=/usr/local/lib/contrast/php/contrast.so
```

- 現在、ライブラリ解析とルート検出は、アプリケーションへの最初のリクエストで実行されます。そのため、最初のリクエストは、それ以降のリクエストに比べてかなり遅くなることが予想されます。

- このエージェントは、サードパーティの PHP 拡張モジュールとはテストされていません。他のサードパーティの拡張モジュール(xdebug、APM などを含む)が使用された場合のエージェントの動作は不確定です。
- プリロードを有効にすると、エージェントが正しく動作しない場合があります。 このエージェントを使用する場合は、プリロードを無効にすることをお勧めします。
- デフォルトでは、エージェントは PHP アプリケーションを実行する際のサーバの作業ディレクトリが、アプリケーションのソースツリーのトップレベルのディレクトリと同じであると想定しています。エージェントは、このパスを使用して、ライブラリ解析とルート検出を行います。パスが異なる場合は、application.path の設定を使用して、アプリケーションのトップレベルのディレクトリを指定する必要があります。

PHP エージェントの設定

全てのエージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。

アプリケーションの実行時に、YAML 設定ファイルまたは環境変数を使用してエージェントを設定できます。

- 独自に YAML 設定ファイルを作成するか、PHP エージェント用に有効な全てのプロパティがある YAML テンプレート (345ページ)を利用することができます。
- もしくは、環境変数 (66ページ)を使用してビルドを設定できます。

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

PHP エージェントの YAML テンプレート

YAML 設定ファイルを使用して PHP エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、YAML 設定の説明 (63ページ)を参照してください)。

YAML 設定ファイルを、アプリケーションの作業ディレクトリかデフォルトの場所(`/etc/contrast/contrast_security.yaml`)に配置します。

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
```

```
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET
```

```
    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
```

```
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

  # \
================================================================================
  # agent.go
  # The following properties apply to any Go agent-wide configurations.
  # \
================================================================================
  # go:

    # \
================================================================================
    # agent.go.preview
    # Enable opt-in Go agent features.
    # \
================================================================================
    # preview:

      # Enable Assess gRPC sources.
      # grpc: false

    # \
================================================================================
    # agent.go.profile
    # Enable Go agent self-profiling features.
    # \
================================================================================
    # profile:

      # Enable CPU profiling for running application.
      # cpu: false

      # Enable memory profiling for running application.
      # mem: false
```

```
# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
```

```
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET
```

```
  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
==========================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
==========================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
==========================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
==========================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
```

```
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET
```

```
    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
```

```
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

  # \
================================================================================
  # agent.go
  # The following properties apply to any Go agent-wide configurations.
  # \
================================================================================
  # go:

    # \
================================================================================
    # agent.go.preview
    # Enable opt-in Go agent features.
    # \
================================================================================
    # preview:

      # Enable Assess gRPC sources.
      # grpc: false

    # \
================================================================================
    # agent.go.profile
    # Enable Go agent self-profiling features.
    # \
================================================================================
    # profile:

      # Enable CPU profiling for running application.
      # cpu: false

      # Enable memory profiling for running application.
      # mem: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true
```

```
# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
```

```
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET
```

```
  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
```

```
# tags: NEEDS_TO_BE_SET
```

Python エージェント

Python エージェントは、Python アプリケーションの IAST(Interactive Application Security Testing)および RASP(Runtime Application Self-Protection)を可能にします。Python エージェントは、最も一般的な Python Web アプリケーションフレームワークに対応しており、 WSGI や ASGI に準拠するその他のアプリケーションやフレームワークとも互換性を保てるようにしています。

Contrast Assess(IAST)では、アプリケーションを通常どおり実行している間にエージェントが脆弱なデータフローパスやその他の問題を特定します。エージェントはこれらの検出結果を Contrast 内で組織に報告します。そのため、アプリケーションを実際の環境にデプロイする前に脆弱性の対策を取ることができます。

Contrast Protect(RASP)では、Python エージェントは HTTP リクエストを検査して、問題を引き起こす可能性のある入力ベクトルを特定します。リクエスト中に、エージェントはデータベースクエリやファイル書込みのほか、リクエストに起因するその他の有害なアクションを検査します。リクエストの最後に、エージェントはレンダリングされた出力を検査して、攻撃が成功したかどうかを確認し、成功した攻撃がユーザに転送されないようにブロックすることができます。そのうえ、パフォーマンス向上など多くの利点が得られるよう、エージェントは全ての解析を内部でネイティブに処理します。

注記
Python エージェントは、Contrast Assess、Contrast Protect、Contrast SCA に対応しています。

次のステップ：

- Python エージェントをインストールする (360ページ)
- Python エージェントのサポート対象テクノロジを確認する (360ページ)

Python エージェントのシステム要件

Python エージェントをインストールする前に、以下のシステム要件を満たす必要があります。

- 検査対象のアプリケーションがデプロイされており、その Web アプリケーションのテクノロジは Contrast のサポート対象であること。
- アプリケーションは再起動可能であること。
- Web サーバが Contrast とネットワークで接続されていること。
- Web サーバが PyPI とネットワークで接続されているか、エージェントが手動でインストールされていること。
- サーバが以下の表に示す最小要件を満たしていること。

要件	バージョン	備考
オペレーティングシステム	• 64 ビット OS X • 64 ビット Linux	エージェントのバージョン 2.3.0 以降で、パッケージのインストール時に C 拡張のコンパイルが必要になります。この処理は自動的に行われますが、ターゲット環境に特定のソフトウェアがインストールされている必要があります。 • 必須：`gcc`、`make`、`automake`、`autoconf`。パッケージ名はプラットフォームによって異なる場合があります。お使いのプラットフォームのバージョンの `build-essential` のインストールやシステムヘッダのインストールが必要な場合があります。Alpine でエージェントを実行する場合は、`libtool` が必要です。

要件	バージョン	備考
Python パッケージ	• protobuf：3.12 以降 • psutil：5.7 以降 • pip：6 以降	

Python エージェントのサポート対象テクノロジ

このエージェントでは、以下のテクノロジをサポートしています。

テクノロジ	サポート対象バージョン	備考
言語のバージョン	• 3.12.x：最初にサポート対象となったエージェントは 5.27.0 • 3.11.x：最初にサポート対象となったエージェントは 5.19.0 • 3.10.x：最初にサポート対象となったエージェントは 5.2.0 • 3.9.x：最初にサポート対象となったエージェントは 4.2.0 • 3.8.x：最初にサポート対象となったエージェントは 2.8.0	Contrast は、**バグフィックス**および **セキュリティ**ステータスの Python LTS(長期サポート)バージョンをサポートします。Python バージョンのサポートは、ワーキンググループが LTS 期間をシフトするのに合わせて行われます。 **サポート対象外：** • 3.6.x、3.5x、2.7.x：サポートする最後のバージョンは 4.14.3 • 3.7.x：サポートする最後のバージョンは 5.27.0
アプリケーションフレームワーク	• Aiohttp：3.7 以降 • Bottle： 0.12 以降 • Django： 3.2 以降 • Django Rest Framework：3.12 以降 • Falcon (ASGI)： 3.0、3.1 • Falcon (WSGI)： 3.0 以降 • FastAPI： 0.71.x 以降 • Flask： 1.1 以降 • Pyramid： 1.10 以降 • Quart：0.15 以降	Python エージェントは、WSGI 互換を意図して作られています。ガイドラインに従う限り、他の WSGI アプリケーションと互換性がある可能性があります。 また、Python エージェントは Django、FastAPI、Quart などの ASGI インターフェイスを提供するフレームワークとも互換性があります。 FastAPI および Starlette(FastAPI が依存している)は比較的新しいライブラリで、継続的な開発変更が行われているため、Contrast が対応できなくなる可能性があります。FastAPI は、開発スピードが速いとされています。 Contrast は記載のバージョンまでサポートを続け、新しいサポート対象がリリースされたときはドキュメントを更新します。
プロセッサのアーキテクチャ	エージェントのテストは x86_64 で実施	他のアーキテクチャで機能する可能性もありますが、正式にはサポートされていません。
Web サーバ	• Gunicorn 0.16.1 –21.2.x • uWSGI 2.0.14 - 2.0.x • Uvicorn 0.14.x - 0.23.x	
データベース	• Mongo (`pymongo`) • MySQL (`PyMySQL`、`mysql-connector`) • PostgreSQL (`psycopg2`) • SQLite3 (`sqlite3`、`pysqlite2`)	
オブジェクト関係マッピングデータベース(ORM)	• Flask-SQLAlchemy • SQLAlchemy	

Python エージェントのインストール

Python エージェントは、Python の標準パッケージとしてインストールされます。

旧バージョン(バージョン 5.19.0 より前)の Python エージェントでは、検出結果を報告するために Contrast サービスを使用します。デフォルトでは、アプリケーションの起動時にサービスが自動的に開始します。エージェントは、個別に実行するスタンドアローンの Contrast サービス (494ページ)と通信するように設定することもできます。

バージョン 5.19.0 以降の Python エージェントでは、Contrast サービスを使用しません。

Python エージェントは、PyPI を使用してインストール (360ページ)するか、Python エージェントのアップデート (361ページ)を使用します。

PyPI を使用した Python エージェントのインストール

PyPI を使用して Python エージェントをインストールするには：

1. pip を使用してエージェントをインストールします。

```
pip install contrast-agent
```

ヒント

`requirements.txt` ファイルがある場合は、このファイルに `contrast-agent` を追加し、`pip install -r requirements.txt` でインストールできます。

2. エージェントを設定します。 (362ページ)
3. エージェントを実行するシステムに `autoconf` がインストールされていることを確認します。
4. Contrast ランナー (414ページ)を使用してアプリケーションを起動し、疎通します。
5. アプリケーションサーバが Contrast に認識され、アプリケーションの検査結果が Contrast に報告されていることを確認します。

Python エージェントのアップデート

Contrast の Python エージェントを自動的に更新するのに信頼性が高く確実な方法は、Python の `pip` を使用して、利用可能な最新バージョンをダウンロードしてインストールすることです。pip は Python アプリケーションのすべての依存関係を管理するため、すでに利用可能でビルド環境の一部になっているはずです。Python エージェントを更新する頻度と更新の取得先は、組織の設定と Contrast の実装 (SaaS 版またはオンプレミス版)によって異なります。

主な手順：

1. Contrast の Python エージェントの入手元を決めます。
2. エージェントをインストールします。
3. スクリプトを使って自動アップデートを行います。

開始する前に

- Contrast エージェントの PyPI リポジトリへアクセスできること。
- Python アプリケーションが Contrast エージェントなしで想定どおりに機能すること。
- 事前に Contrast の Python エージェントが正常にインストールされていること。
- 変更管理ポリシーと使用する環境に基づいて、エージェントをアップデートする方法とタイミングを決めていること。

自動アップデートのスクリプトを使用

1. Contrast の Python エージェントの入手元を決めます。
   - PyPI のパブリック(またはプライベート)リポジトリ
2. Contrast の Python エージェントを `requirements.txt` に依存関係として指定します。
   `requirements.txt` は、Python アプリケーションが PyPI(パブリックまたはプライベート)リポジトリからのアーティファクトでビルドされるたびに自動的に解決する依存関係を指定するファイルです。このファイルに Contrast の Python エージェントを指定することで、アプリケーションの新しいビルドを常に最新バージョンのエージェントに合わせることが簡単になります。Contrast エージェント(contrast-agent)のバージョンは指定しないで下さい。そうすれば最新のバージョンが取得されます。
3. `requirements.txt` を更新した後、アプリケーションをビルドする際に次のコマンドを使用します。これにより、Contrast の Python エージェントが PyPI から自動的にダウンロードされ、Python アプリケーションに追加または更新されます。

```
$ pip install -U -r requirements.txt
```

手動でアップデート

組織によっては、requirements.txt ファイルを環境間で一貫させる必要がある場合や、すべての環境に Contrast の Python エージェントをインストールする予定がない場合があります。このような場合、手

動でエージェントをインストールします。Python のビルドプロセスの一環として手動でエージェントを更新できます。

1. Contrast の Python エージェントの入手元を決めます。
   - PyPI のパブリック(またはプライベート)リポジトリ
2. 次のコマンドを使用して、Contrast の Python エージェントを手動で取得し、PyPI(パブリックまたはプライベート)から Python アプリケーションに追加または更新します。

```
$ pip install -U contrast-agent
```

関連項目

- Python エージェントのサポート対象テクノロジ (360ページ)
- Python エージェントのインストール (360ページ)

Python エージェントの設定

全てのエージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。

エージェントのバージョン 5.24.0 以降は、サポート対象のほとんどのフレームワークで Python エージェントを使用する方法として、Contrast ランナー (414ページ)が推奨されます。手動でのミドルウェアの設定 (384ページ)は、下位互換性のために引き続きサポートされ、特定のフレームワークやアプリケーションでは今後も必要な場合があります。

Contrast サービスの設定(5.19.0 より前のバージョンのみ)

旧バージョン(バージョン 5.19.0 より前)の場合、Python エージェントは起動時に実行ファイルを起動し、この実行ファイルにも設定ファイルへのアクセスが必要です。サービスは通常、Python エージェントのプロセスによって起動されるため、サービスはエージェントと同じ設定ファイルにアクセスすることができます。ただし、サービスが単独で起動された場合には、設定ファイルの優先順位 (62ページ)が同じになるよう試みます。

つまり、(通常の場合のように)サービスの作業ディレクトリがアプリケーションのベースディレクトリでもある場合、サービスはアプリケーションの設定ファイルを共有できるということです。エージェントとサービスの両方で、 `/etc/contrast/contrast_security.yaml` パスを使用するということです。

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

Python エージェントの YAML テンプレート

YAML 設定ファイルを使用して Python エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、こちらの説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します：`/etc/contrast/contrast_security.yaml`

注記

YAML 設定ファイルの `agent.service` セクションは、旧バージョンの Python エージェント(バージョン 5.19.0 より前)のみに適用されます。

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
```

```
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:
```

```
    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead
    # of the file system. Overriden by `stdout` configuration.
    # stderr: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
```

```
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING
```

```
  # \
================================================================================
  # agent.python
  # The following properties apply to any Python agent-wide configurations.
  # \
================================================================================
  # python:

    # Allow the agent to dump `cProfile` data to file for each request.
    # enable_profiler: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
```

```
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
```

```
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
```

```
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
```

```
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET
```

```
  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
==========================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
==========================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
==========================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
==========================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
```

```
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
```

```
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead
    # of the file system. Overriden by `stdout` configuration.
    # stderr: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:
```

```
    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
```

```
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.python
  # The following properties apply to any Python agent-wide configurations.
  # \
================================================================================
  # python:

    # Allow the agent to dump `cProfile` data to file for each request.
    # enable_profiler: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
```

```
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
```

```
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
```

```
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
```

```
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
```

```
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
```

```
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

Python エージェントの設定の検証

Python エージェントには、エージェントの設定を検証し、テストするためのスクリプトが含まれています。

スクリプトを実行する手順

1. `pip` を使用して、contrast-agent のバージョン 5.1.0 以降をインストールします。
2. YAML ファイル (362ページ)や環境変数 (66ページ)、またはその両方を使用して、Contrast エージェントの設定を準備します。
3. `contrast-validate-config` というコマンドを実行します。 このスクリプトは、現在の設定を検証し、Contrast と通信できるかをテストします。

注記

Response: 400 Bad Request のメッセージが表示されても、Contrast への接続は成功しています。

以下の例の場合、Contrast Web インターフェイスにまだアプリケーションのサーバが作成されていないことを示しています。

```
[contrast-validate-config] Sending test request to Contrast UI
[contrast-validate-config] Response: 400 Bad Request
[contrast-validate-config] {
 "success" : false,
 "messages" : [ "Invalid agent request" ]
}
[contrast-validate-config] 400 status code indicates success \
for this endpoint
[contrast-validate-config] Connection to the Contrast UI \
successful
```

そして、以下の例では、既に Contrast Web インターフェイスにはサーバが作成されていることを示しています。

```
[contrast-validate-config] Sending test request to Contrast UI
[contrast-validate-config] Response: 304 Not Modified
[contrast-validate-config] 304 status code indicates success \
for this endpoint
[contrast-validate-config] Connection to the Contrast UI \
successful
```

関連項目

- Python エージェントの設定 (362ページ)

ミドルウェアの設定

警告

手動によるミドルウェアの設定は、現在では推奨されていません。代わりに、`contrast-python-run` コマンドを使ってアプリケーションを実行する必要があります。このコマンドによって、フレームワーク固有のエージェントの組み込みが自動的に検出されて有効になります。詳細は、Contrast ランナー (414ページ)を参照してください。

手動によるミドルウェアの設定が必要なまれなケースについては、以下の手順を参照してください。

ミドルウェアは、Web アプリケーションの一部を構成するソフトウェアコンポーネントであり、リクエストを受け取って必要な処理を行いレスポンスを返すことができます。

Python エージェントは、Contrast がサポートする全てのフレームワークのミドルウェアとして実装されます。Python エージェントを使用するには、アプリケーションで使用しているフレームワークのミドルウェアを設定する必要があります。

AIOHTTP

AIOHTTP ミドルウェア は、aiohttp の `web.Application` コンストラクタに引数として渡されるクラスベースのミドルウェア です。次の例は、Contrast ミドルウェアクラスを使用する AIOHTTP アプリケーションの例です。

```
from aiohttp import web
from contrast.aiohttp import ContrastMiddleware


routes = web.RouteTableDef()

@routes.get("/")
def index(request):
    raise web.HTTPFound("/hello")


middlewares = [ContrastMiddleware(app_name="app name")]
app = web.Application(middlewares=middlewares)
app.add_routes(routes)

web.run_app(app)
```

Bottle

Contrast の Bottle ミドルウェアは WSGI ミドルウェアで、Bottle アプリケーションのインスタンスをラッピングすることによって動作します。

Bottle で Python エージェントを設定するには：

1. アプリケーションのコードベースで、Bottle アプリケーションのオブジェクトを検索します。これは、`bottle.Bottle` のインスタンスになります。
   以下はサンプルの Bottle アプリケーションで、app はお使いの Bottle アプリケーションのオブジェクトになります。

   ```
   from bottle import Bottle, run

   app = Bottle(__name__)

   @app.route("/")
   def index():
       return "hello world"

   <InstallContrastHere>

   run(app)
   ```
2. Bottle アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、`<InstallContrastHere>`を次のように置き換えます。

   ```
   from contrast.bottle import ContrastMiddleware
   app = ContrastMiddleware(app)
   ```

重要
まれに、WSGI アプリケーションオブジェクトに加えて、元の `bottle.Bottle` アプリケーションインスタンスを `ContrastMiddleware` に直接渡すことが必要になる場合があります。アプリケーションが起動しない場合は、元の Bottle アプリケーションを検索し、Contrast のミドルウェアに渡します。例えば、以下のようになります。

```
app = ContrastMiddleware(
    app,
    original_bottle_app,  # instance of bottle.Bottle
)
```

Django

新設定手順：Contrast Python エージェントのバージョン 4.6.0 以降はこの手順を利用します

Contrast の Django ミドルウェアは WSGI ミドルウェアであり、Django スタイルのミドルウェアではありません。

1. WSGI アプリケーションのオブジェクトを検索します。`WSGI_APPLICATION` という Django の設定オプションでプロジェクトの WSGI アプリを指定しますが、通常これは `wsgi.py` にあります。`WSGI_APPLICATION` をまだ設定していない場合は、設定する必要があります。
   以下はサンプルの `wsgi.py` で、`application` はお使いの WSGI アプリケーションのオブジェクトになります。

   ```
   from django.core.wsgi import get_wsgi_application
   application = get_wsgi_application()

   <InstallContrastHere>
   ```
2. WSGI アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、`<InstallContrastHere>`を次のように置き換えます。

   ```
   from contrast.django import ContrastMiddleware
   application = ContrastMiddleware(application)
   ```

旧設定手順：Contrast Python エージェントのバージョン 5.0.0 以降でこの手順は利用できません

Django ミドルウェアの設定は、`settings.py` ファイルで行います。

1. **`settings.py` ファイルを検索してください。**このファイルは、すべてのアプリケーションで同じ場所にあるわけではありませんが、通常はアプリケーションソースツリーの最上部などにあります。一般的な場所は次のとおりです。
   - `/settings.py`
   - `config/settings.py`
   - `app/settings.py`

   注記
   ソースツリーで *settings.py* を検索する場合は、Python の仮想環境に該当するディレクトリは必ず除外してください。

   アプリケーションによって、複数の *settings.py* ファイルが存在し、さまざまなアプリケーション構成(例えば、本番環境やテスト環境など)に対応している場合があります。そのような場合には、Contrast エージェントのミドルウェアを使用する全ての設定に追加してください。

2. **Contrast エージェントモジュールをリストに追加してください。**
   Contrast ミドルウェアを可能な限りリストの最初の方に追加してください。ただし、状況によって、アプリケーションを動作させるために順序の変更が必要になる場合があります。
   - **Django 1.10 以降**：配列になっている `MIDDLEWARE` という設定変数を検索します。リストに Contrast エージェントモジュールを追加します。

     ```
     MIDDLEWARE = [
       'contrast.agent.middlewares.django_middleware.DjangoMiddleware',
       # OTHER MIDDLEWARE,
     ]
     ```
   - **Django 1.6 から 1.9 :** *settings.py* で `MIDDLEWARE_CLASSES` という設定変数を検索し、リストに Contrast エージェントモジュールを追加します。

     ```
     MIDDLEWARE_CLASSES = [
      \
      'contrast.agent.middlewares.legacy_django_middleware.DjangoMiddleware'
     ,
       # OTHER MIDDLEWARE
     ]
     ```

   ミドルウェアの組み込みの詳細については、Django のドキュメントを参照してください。

Falcon(ASGI)

Falcon(ASGI)ミドルウェアは、Falcon(ASGI)アプリケーションのインスタンスをラッピングすることによって動作する ASGI ミドルウェアです。

以下の例は、Falcon(ASGI)アプリケーションを Contrast ミドルウェアのクラスでラップするサンプルです。

注記
Falcon(ASGI)アプリケーションで Contrast 以外のミドルウェアを使用している場合、特定の機能を動作させるために、Contrast を最初のミドルウェアとして初期化する必要があります。

```
import falcon.asgi
from contrast.falcon_asgi import ContrastMiddleware


class Home(object):
    async def on_get(self, req, resp):
        resp.status = falcon.HTTP_200
        resp.body = "Hello World"

def create():
    # This is where the example app is declared. Look for something \
similar in your
    # application since this instance needs to be wrapped by Contrast \
middleware
    _app = falcon.asgi.App()

    # Add routes to your app
    home = Home()
```

```
    _app.add_route("/home", home)

    # This line wraps the application instance with the Contrast middleware
    # NOTE: Contrast should be the first middleware if others are used
    _app = ContrastMiddleware(_app)
    return _app


app = create()
```

Falcon(WSGI)

Contrast の Falcon ミドルウェアは WSGI ミドルウェアであり、Falcon スタイルのミドルウェアではありません。

1. Falcon アプリケーションのオブジェクトを検索します。これは、Falcon のバージョンによって、`falcon.API` または `falcon.App` のインスタンスとなります。
   以下はサンプルの Falcon アプリケーションで、app はお使いの Falcon アプリケーションのオブジェクトになります。

   ```
   import falcon
   from views import Home

   app = falcon.API()

   home = Home()
   app.add_route('/home', home)

   <InstallContrastHere>
   ```

2. Falcon アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、`<InstallContrastHere>`を次のように置き換えます。

   ```
   from contrast.falcon import ContrastMiddleware
   app = ContrastMiddleware(app)
   ```

   重要
   まれに、WSGI アプリケーションオブジェクトに加えて、元の `falcon.App` や `falcon.API` アプリケーションインスタンスを `ContrastMiddleware` に直接渡すことが必要になる場合があります。アプリケーションが起動しない場合は、元の Falcon アプリケーションを検索し、Contrast のミドルウェアに渡します。例えば、以下のようになります。

   ```
   app = ContrastMiddleware(
     app,
     original_falcon_app,  # instance of falcon.App or \
   falcon.API
   )
   ```

   重要
   ルート登録が*完了後*、Falcon インスタンスをラップする必要があります。

FastAPI

FastAPI ミドルウェアは、`starlette.middleware.BaseHTTPMiddleware` に依存する、クラスベースの ASGI ミドルウェアです。Contrast がサポートする FastAPI のバージョンについては、Python エ

ージェントのサポート対象テクノロジ (360ページ)を確認してください。次の例は、Contrast ミドルウェアクラスを使用した FastAPI アプリケーションのサンプルです。

```
from fastapi import FastAPI
from contrast.fastapi import ContrastMiddleware

app = FastAPI()
app.add_middleware(ContrastMiddleware, original_app=app)

@app.get("/")
def read_root():
    return RedirectResponse("/home")
```

FastAPI アプリケーションで Contrast 以外のミドルウェアを使用している場合、特定の機能を動作させるために、Contrast を最初のミドルウェアとして初期化する必要があります。

```
from fastapi import FastAPI
from fastapi.middleware.httpsredirect import HTTPSRedirectMiddleware
from contrast.fastapi import ContrastMiddleware

app = FastAPI()

# ContrastMiddleware must be the first middleware
app.add_middleware(ContrastMiddleware, original_app=app)
app.add_middleware(HTTPSRedirectMiddleware)
```

一部の機能で、FastAPI の app を `original_app` キーワードの引数として渡す必要があるため、現時点では `add_middleware` メソッドでミドルウェアを追加する方法のみがサポートされます。ミドルウェアを直接 FastAPI クラスの初期化に渡すことでミドルウェアを初期化する方法は、Contrast では現在サポートされていません。

```
# Not currently supported.
app = FastAPI(middleware=[...])
```

警告

`add_middleware` を複数回呼ぶと、 前のミドルウェアが全て再初期化されます。

Flask

Contrast の Flask ミドルウェアは WSGI ミドルウェアで、Flask アプリケーションのインスタンスをラッピングすることによって動作します。

1. Flask アプリケーションのオブジェクトを検索します。これは、`flask.Flask` のインスタンスになります。
   以下はサンプルの Flask アプリケーションで、app はお使いの Flask アプリケーションのオブジェクトになります。

   ```
   import Flask

   app = Flask(__name__)

   <InstallContrastHere>
   ```

```
@app.route('/')
def index():
    return render_template('index.html')

app.run(...)
```

2. Flask アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、<InstallContrastHere>を次のように置き換えます。

```
from contrast.flask import ContrastMiddleware
app.wsgi_app = ContrastMiddleware(app)
```

> **注記**
> Contrast の Flask ミドルウェアは flask.Flask.wsgi_app ではなく、flask.Flask のインスタンスを引数として必要とします。

Pyramid

旧設定手順：Contrast Python エージェントのバージョン 5.0.0 以降でこの手順は利用できません

Pyramid では、ミドルウェアは「tween」と呼ばれています。

1. アプリケーションのコードベースで、Configurator オブジェクトを検索します。例えば、次のようなものです。

```
from pyramid.config import Configurator
config = Configurator()

<InstallContrastHere>
```

2. Contrast のミドルウェアを設定に追加します。上記の例では、<InstallContrastHere>を次のように置き換えます。

```
config.add_tween('contrast.agent.middlewares.pyramid_middleware.PyramidMiddleware')
```

tween の設定に関する詳細については、Pyramid のドキュメントを参照してください。

新設定手順：Contrast Python エージェントのバージョン 4.6.0 以降ではこの手順を利用します

Contrast の Pyramid ミドルウェアは WSGI ミドルウェアであり、Pyramid スタイルの「tween」ではありません。

1. WSGI アプリケーションのオブジェクトを検索します。これは、多くの場合 Configurator.make_wsgi_app()への呼び出しによって作成されます。
例えば、以下のようになります。

```
from pyramid.config import Configurator

config = Configurator()
app = config.make_wsgi_app()

<InstallContrastHere>
```

2. WSGI アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、<InstallContrastHere>を次のように置き換えます。

```
from contrast.pyramid import ContrastMiddleware
app = ContrastMiddleware(app)
```

重要
まれに、アプリケーションのレジストリ(Registry)オブジェクトもミドルウェアに渡す必要がある場合があります。レジストリは通常、Configurator インスタンスと Pyramid アプリケーションオブジェクトの両方の属性として利用できます。

アプリケーションが起動しない場合は、アプリケーションのレジストリを検索し、Contrast のミドルウェアに渡します。以下は、その例です。

```
app = ContrastMiddleware(app, config.registry)
```

Quart

Quart ミドルウェアは、Quart アプリケーションオブジェクトをラップして `asgi_app` 属性に割り当てる必要があります。以下の例は、Contrast ミドルウェアクラスを使用する Quart アプリケーションのサンプルです。

```
from quart import Quart, redirect
from contrast.quart import ContrastMiddleware


app = Quart(__name__)
app.asgi_app = ContrastMiddleware(app)

@app.route("/")
async def index():
    return redirect("/home")
```

WSGI

Contrast は、エージェントのコア機能を全て含む汎用の WSGI ミドルウェアを提供しています。ルート探索などのフレームワーク固有の機能は、汎用の WSGI ミドルウェアには実装されていません。

Contrast を使用して WSGI 準拠のアプリケーションを検査するには：

1. アプリケーションのコードベースで、WSGI アプリケーションのオブジェクトを検索します。例えば、次のように作成された WSGI アプリケーションがあるとします。

   ```
   from example.module import make_app
   wsgi_app = make_app()

   <InstallContrastHere>
   ```

2. WSGI アプリケーションのオブジェクトを Contrast のミドルウェアでラップします。上記の例では、`<InstallContrastHere>`を次のように置き換えます。

   ```
   from contrast.wsgi import ContrastMiddleware
   wsgi_app = ContrastMiddleware(wsgi_app)
   ```

WSGI ミドルウェアの詳細については、WSGI の仕様を参照してください。

Python Web サーバ

Python エージェントは、いくつかの一般的な Web サーバで検証済みです。一部のサーバでは、Contrast を適切に動作させるために特定の設定オプションが必要です。

Gunicorn

Gunicorn は、Django、Pyramid、Tornado などの一般的なフレームワークと一緒に組み合わせて使うことができます。

Gunicorn を起動するには次のコマンドを使用します：`gunicorn app.wsgi_app:application` (app はアプリケーションフォルダ名)

Gunicorn サーバで次のような設定ができます。

- **バインド：**
  `-b ADDRESS1`
  `gunicorn -b 127.0.0.1:8000`
- **タイムアウト：**
  `-t INT  --timeout INT`
  この秒数以上反応がないワーカは、強制終了されて、再起動されます。
  値は、正の数か `0` になります。 `0` を設定すると、全てのワーカのタイムアウトが完全に無効になり、タイムアウトは無限になります。
  デフォルトは 30 秒です。同期処理のワーカ(Sync ワーカ)への影響が確実な場合にのみ、この値を大きくします。非同期処理のワーカの場合には、ワーカプロセスは通信中となるため、1 つのリクエストを処理するために必要な時間の長さが影響することはありません。

Uvicorn

Uvicorn は非同期の Web サーバで、`pip install uvicorn[standard]`でインストールすることができます。

以下は、推奨される設定です。

- `--loop=uvloop`
- `--http=httptools`
  `uvloop` と `httptools` はどちらも Cython で書かれており、パフォーマンスが向上しますが、Windows や PyPy との互換性がありません。
- `--interface`
  いくつかの ASGI プロトコルに対してこのオプションを設定できます。WSGI を使用していて、`–ws=websockets` を使用している場合、websockets は使用されず、デフォルトの auto になります。
- `--ws`
  `--ws-max-size`、`--ws-ping-interval`、`--ws-ping-timeout` を設定しており、`–ws` が websockets で設定されていない場合、全て無視されます。
- Uvicorn のマネージャとして gunicorn を使用する場合は、以下を実行します。
  `gunicorn -k uvicorn.workers.UvicornWorker`
- UvicornWorker のデプロイでは、`uvloop` と `httptools` を使用します。PyPy で実行する場合には、代わりに pure-Python の実装を使用する必要があります。これを行うには、`UvicornH11Worker class.gunicorn -k uvicorn.workers.UvicornH11Worker` を使用します。

uWSGI の設定

uWSGI で Contrast を実行する場合は、次の設定オプションが必要です。コマンドラインまたは uWSGI 設定(.ini)ファイルで指定します。

- `--enable-threads`：スレッドの使用を有効にします。エージェントがバックグラウンドスレッドを開始できるように、このオプションを有効にする必要があります。
- `--single-interpreter`：初期化された Python プロセスで、Python エージェントが有効になるようにします。このオプションを指定すると、アプリケーションのリクエストを処理するプロセスと同じプロセスで Contrast が初期化されます。
- `--master` を指定する場合は、`--lazy-apps` も指定します。マスターモードで実行すると、uWSGI はマスタープロセスでアプリケーションを初期化しますが、リクエストを処理するワーカーにこのプロセスがフォークされます。正しく動作させるためには、各ワーカープロセスで個々に Contrast を初期化する必要がありますが、`--lazy-apps` で実現できます。

Python エージェントの YAML テンプレート

YAML 設定ファイルを使用して Python エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、こちらの説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します：`/etc/contrast/contrast_security.yaml`

注記

YAML 設定ファイルの `agent.service` セクションは、旧バージョンの Python エージェント(バージョン 5.19.0 より前)のみに適用されます。

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
```

```
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
```

```
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

    # Set to `true` to redirect all logs to `stderr` instead
    # of the file system. Overriden by `stdout` configuration.
    # stderr: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
```

```
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
```

```
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.python
  # The following properties apply to any Python agent-wide configurations.
  # \
================================================================================
  # python:

    # Allow the agent to dump `cProfile` data to file for each request.
    # enable_profiler: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
```

```
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
```

```
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
```

```
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:
```

```
      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
```

```
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
```

```
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==



# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
```

```
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
```

```
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false
```

```
    # Set to `true` to redirect all logs to `stderr` instead
    # of the file system. Overriden by `stdout` configuration.
    # stderr: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # Change the Contrast security logger from a file-sized based rolling
    # scheme to a date-based rolling scheme. At midnight server time,
    # the log from the previous day is renamed to *file_name.yyyy-MM-dd*.
    # Note - this scheme does not have a size limit; manual log
    # pruning will be required. This flag must be set to use the
    # backups and size flags. Value options are `true` or `false`.
    # roll_daily: NEEDS_TO_BE_SET

    # Specify the file size cap (in MB) of each log file.
    # roll_size: NEEDS_TO_BE_SET

    # Specify the number of backup logs that the agent will create before
    # Contrast cleans up the oldest file. A value of `0` means that no \
backups
    # are created, and the log is truncated when it reaches its size cap.
    #
    # Note - this property must be used with
    # `agent.security_logger.roll_daily=false`; otherwise,
    # Contrast continues to log daily and disregard this limit.
    #
    # backups: NEEDS_TO_BE_SET

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:
```

```
      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.python
  # The following properties apply to any Python agent-wide configurations.
  # \
================================================================================
  # python:

    # Allow the agent to dump `cProfile` data to file for each request.
    # enable_profiler: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:
```

```
  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10
```

```
    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
```

```
# ============================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
# ============================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
# ============================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
# ============================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
# ============================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
# ============================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
# ============================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off
```

```
      # Detect when custom code attempts to access sensitive
      # system files. The agent blocks if blocking is enabled.
      # detect_custom_code_accessing_system_files: true

      # Detect when users attempt to bypass filters by
      # using "::$DATA" channels or null bytes in file
      # names. The agent blocks if blocking is enabled.
      # detect_common_file_exploits: true

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
```

```
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
```

```
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

Python のテレメトリ

Python エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集され、その後も定期的に(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切 (964ページ)であると考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、Contrast に安全に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	データ
Python 4.14.0	エージェントのバージョン
	オペレーティングシステムとバージョン
	Python のバージョン
	アプリケーションのフレームワークとバージョン
	Web サーバとバージョン
	Contrast インスタンスが SaaS 版かオンプレミス版であるか

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、telemetry.python.contrastsecurity.com に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

Contrast ランナー

バージョン 5.24.0 より、Python エージェントは、Python アプリケーションを検査するための新しいコマンドラインインターフェイス(別名、ランナー)を提供するようになりました。原則として、ランナーを使う場合、ミドルウェアを手動で設定 (384ページ)する必要がなくなりました。代わりに、Contrast ランナーが自動的にフレームワーク固有のエージェント組込みをアプリケーションに適用します。

ランナーのコマンドは `contrast-python-run` は呼ばれ、 `contrast-agent` パッケージの一部として提供されます。`contrast-agent` パッケージを インストール (360ページ)したら、ほとんどの Python 環境で何も変更することなく、コマンドラインで使用できます。

ランナーの使用

ランナーを使用するには、アプリケーションを起動する元のコマンドの先頭に、`contrast-python-run` コマンドを追加します。

- 例えば、以下のコマンドで起動する **Django** アプリケーションの場合：

```
python manage.py runserver
```

  - このアプリケーションに Contrast を追加するには、以下のコマンドを実行します。

```
contrast-python-run -- python manage.py runserver
```

注記
ダッシュ 2 個の区切り文字「--」は、ランナーコマンドの引数と、元のコマンドの引数を分けるために使用します。

- 別の例として、以下のコマンドで起動する **Flask** アプリケーションの場合：

```
FLASK_APP=apps/app.py flask run --host=localhost --port=8080
```

- Contrast ランナーを使用すると以下のようになります(環境変数の設定はランナーコマンドの前に行うことに注意)。

```
FLASK_APP=apps/app.py contrast-python-run -- flask run --
host=localhost --port=8080
```

- `contrast-python-run` コマンドは、**uwsgi** や **gunicorn** などの Web サーバでデプロイしている場合にも使用できます。例：

```
contrast-python-run -- gunicorn apps/app:app --preload -b localhost:8080
```

- また、**mod_wsgi-express** を使って Apache にデプロイしている場合にも使用できます。例：

```
contrast-python-run -- mod_wsgi-express start-server app/wsgi.py --
user=www-data --group www-data
```

ランナーは、Contrast の設定の通常の優先順位 (62ページ)に従います。コマンドラインから直接ランナーで環境変数を使用することも可能です。

```
CONTRAST__AGENT__LOGGER__LEVEL=DEBUG contrast-python-run -- python \
manage.py runserver
```

Ruby エージェント

Ruby エージェントは、最も一般的な Web アプリケーションフレームワークとの互換性がある Rack ミドルウェアです。Ruby エージェントは、Rack との高度な互換性を実現し、Rack を基盤とするアプリケーションに IAST(Interactive Application Security Testing)および RASP(Runtime Application Self-Protection)機能を提供します。

Assess では、アプリケーションを通常どおり実行している間にエージェントが脆弱なデータフローパスやその他の問題を特定します。エージェントはこれらの検出結果を Contrast 内の組織に報告します。そのため、アプリケーションを実際の環境にデプロイする前に脆弱性に対処することができます。

Protect では、Ruby エージェントが HTTP リクエストを検査して、問題を引き起こす可能性のある入力ベクトルを特定します。リクエスト中に、データベースクエリとファイル書込みのほか、リクエストの結果が原因で損害を及ぼす可能性のあるその他のアクションをエージェントが検査します。リクエストの最後に、エージェントはレンダリングされた出力を検査し、攻撃が成功したかどうかを確認し、成功した攻撃がユーザに転送されないようにブロックすることができます。そして、攻撃の詳細が Contrast サーバに送信され、その後アラートが送信されると攻撃の詳細がインターフェイスに表示されます。

注記
Ruby エージェントは、Contrast Assess、Contrast Protect、Contrast SCA に対応しています。

次のステップ：

- Ruby エージェントをインストール (417ページ)する
- Ruby エージェントのサポート対象テクノロジを確認 (416ページ)する

Ruby エージェントのシステム要件

Ruby エージェントをインストールする前に、以下のシステム要件を満たす必要があります。

- 検査対象のアプリケーションがデプロイされており、その Web アプリケーションのテクノロジは Contrast のサポート対象であること。

- アプリケーションは再起動可能であること。
- Web サーバが Contrast とネットワークで接続されていること。
- Web サーバが RubyGems とネットワークで接続されているか、エージェントが手動でインストールされていること。
- サーバが以下の表に示す最小要件を満たしていること。

要件	バージョン	備考
オペレーティングシステム	• 64 ビット Linux (推奨) • 64 ビット OS X • 64 ビット Alpine	エージェントのバージョン 3.0.0 以降で、gem インストール時に C 拡張のコンパイルが必要になります。この処理は自動的に行われますが、ターゲット環境で以下のインストールが必要な場合があります。 • `gcc`、`make`、`automake`、`autoconf` (パッケージ名は異なる場合があります。また、お使いのプラットフォームのバージョンの `build-essential` のインストールが必要な場合があります。) • システムヘッダ エージェントは、Ruby のアプリケーション層で実行する際に C との依存関係がいくつかあるため、*g libc* または *musl libc* ベースのシステムでのコンパイルを許可するために`--platform ruby` フラグを付けてインストールしなければならない場合があります。
Ruby gems	• ougai : ~>1.8 (1.8 以降、2 より前) • parser : ~>2.6 (2.6 以降、3 より前) • protobuf : ~>3.10 (3.10 以降、4 より前) • rack : ~>2.0 (2.0 以降、3 より前)	

Ruby エージェントのサポート対象テクノロジ

このエージェントでは、以下のテクノロジをサポートしています。

テクノロジ	サポート対象バージョン	備考
言語のバージョン	• 3.2.x : 最初のサポート対象エージェントは 6.13.0 • 3.1.x : 最初のサポート対象エージェントは 6.0.0 • 3.0.x : 最初のサポート対象エージェントは 4.6.0 具体的なリリース日については、Ruby メンテナンスブランチのスケジュールを参照してください。	Contrast は、通常のメンテナンスおよびセキュリティメンテナンスステータスの Ruby LTS(長期サポート)バージョンをサポートします。Ruby バージョンのサポートは、ワーキンググループが LTS 期間をシフトするのに合わせて変更されます。 **サポート対象外 :** • 2.5.x : 最後のサポート対象エージェントは 4.14.1 • 2.4.x : 最後のサポート対象エージェントは 3.9.1 • 2.6.x : 最後のサポート対象エージェントは 5.3.0 • 2.7.x : 最後のサポート対象エージェントは 6.15.3

テクノロジ	サポート対象バージョン	備考
アプリケーションフレームワーク	• Rails 6.x - 7.x • Grape 1.5.x • Sinatra 3.x	正式にサポートされていない Rack ベースの Web フレームワークでもエージェントは動作しますが、サポート対象のフレームワークの場合と比べて具体性の低い検出結果となる可能性があります。アプリケーションコード内で発生した脆弱性が報告されるのではなく、Rack メソッドと直接のインターフェイスとなるフレームワークコード内の脆弱性が報告される可能性があります。 **サポート対象外：** • Rails 3.x：最後のサポート対象エージェントは 3.11.0 • Rails 4.x：最後のサポート対象エージェントは 3.11.0 • Rails 5.x：最後のサポート対象エージェントは 6.15.3
データベース	• MongoDB • Mysql2 • PG • SQLite3	Contrast は、古いバージョンや非推奨バージョンのサードパーティモジュールをサポートしません。
テスト環境	サポート対象のオペレーティングシステム、アプリケーションフレームワーク、Web サーバの組み合わせでテストを実施し、Ruby Mspec Suite も実行します。	エージェントに変更が加えられると、Contrast は一連の自動化テストを実行し、サポート対象のすべての Ruby バージョンについてサポート対象テクノロジでの検出結果を確認します。Contrast 独自のテストに加えて、エージェントを有効にした環境で Ruby Spec Suite を実行します。
Web サーバ	• Passenger 5.37、6.x • Puma 3.7 - 5.x • Thin 1.7.2 - 1.8.x • Unicorn 5.0.x - 6.x	

Ruby エージェントのインストール

`contrast-agent.gem` は、 アプリケーションの Gemfile に追加する標準の Ruby ライブラリです。

Ruby エージェントを RubyGems を使用して gem source として (417ページ)インストールする、もしくは Ruby エージェントをアップデート (418ページ)してください。

RubyGems から gem source として Ruby エージェントをインストール

RubyGems から Ruby エージェントをダウンロードするには：

1. 以下をアプリケーションの gemfile に追加します。

```
gem 'contrast-agent'
```

2. インストールを実行します。

```
bundle install
```

または更新を実行します。

```
bundle update contrast-agent
```

3. ミドルウェアを設定します (419ページ)(Rails、Sinatra または Grape)。
4. エージェントを設定します。 (419ページ)
5. エージェントを実行するシステムに `autoconf` がインストールされていることを確認します。
6. Contrast にアプリケーションが表示されることを確認します。

注記
Alpine にインストールする場合は、インストール前に `bundle config force_ruby_platform true` コマンドを実行する必要があります。

注記
特定の環境でのみ Contrast で実行したい場合は、Bundler のグループ機能を使用して実行できます。

Ruby エージェントのアップデート

Contrast の Ruby エージェントを自動的に更新するのに最も信頼性が高く効率的な方法は、`Ruby Bundler` を使用し、利用可能な最新バージョンをダウンロードしてインストールすることです。 Ruby の Bundler は通常、Ruby アプリケーションのすべての依存関係を管理するため、すでにビルド環境の一部として利用可能になっているはずです。Ruby エージェントを更新する頻度と更新の取得先は、組織の設定と Contrast の実装(SaaS 版またはオンプレミス版)によって異なります。

主な手順：

1. Contrast Ruby エージェントの入手元を決めます。
2. エージェントをインストールします。
3. スクリプトを使って自動アップデートを行います。

開始する前に

- Ruby の Bundler パッケージマネージャにある程度の知識があること。
- Contrast エージェントの RubyGems リポジトリへアクセスできること。
- Ruby アプリケーションが Contrast エージェントなしで想定どおりに機能すること。
- 事前に Contrast の Ruby エージェントが正常にインストールされていること。
- 変更管理ポリシーと使用する環境に基づいて、エージェントをアップデートする方法とタイミングを決めていること。

エージェントのインストールとスクリプトによる自動アップデート

1. Contrast の Ruby エージェントの入手元を決めます。
   - RubyGems のパブリック(またはプライベート)リポジトリ
2. Gemfile に Contrast の Ruby エージェントを含めると、アプリケーションの新しいビルドで常に最新バージョンのエージェントを使用するよう合わせることができます。Contrast エージェント

(contrast-agent)のバージョンは指定しないで下さい。そうすれば最新のバージョンが取得されます。
Gemfile は、Ruby アプリケーションが RubyGems(パブリックまたはプライベート)リポジトリからのアーティファクトでビルドされるたびに自動的に解決する依存関係を指定するファイルです。

3. Gemfile を更新した後、アプリケーションをビルドするときに次のコマンドを使用します。これにより、Contrast の Ruby エージェントが RubyGmes から自動的にダウンロードされ、Ruby アプリケーションに追加または更新されます。

```
$ bundle install
```

4. Gemfile に contrast-agent を追加した後は、次のように Bundler を使用してエージェントを更新できます。

```
bundle update contrast-agent
```

Gem を手動でインストールする

Ruby のビルドプロセスの一部として手動でエージェントを更新することができます。この場合、2 つの方法があります。 お客様の組織とワークフローに最適な方法を選択してください。

- **RubyGems** : 次のコマンドを使用して、RubyGems(パブリックまたはプライベート)から Contrast の Ruby エージェントを取得してアプリケーションにインストールします。

```
$ gem install contrast-agent
```

上記のコマンドはエージェントをローカルにインストールするのみのため、Bundler で更新を管理するには、Gemfile に次の行を追加します。

```
gem "contrast-agent"
```

次に、Bundler を使用してエージェントをインストールまたは更新するには、次のコマンドを実行します。

```
$ bundle install
```

- 上記のコマンドはエージェントをローカルにインストールするのみのため、Bundler で更新を管理するには、Gemfile に次の行を追加します。

```
gem "contrast-agent"
```

関連項目

- Ruby エージェントのサポート対象テクノロジ (416ページ)
- Ruby エージェントのインストール (417ページ)

Ruby エージェントの設定

全てのエージェントには基本の設定 (60ページ)があり、設定値には優先順位 (62ページ)があります。YAML 設定ファイルを使用して、エージェントを設定します。

ヒント

Contrast エージェント設定エディタ (64ページ)を使用すると、 YAML 設定ファイルの作成やアップロード、YAML の検証、推奨される設定値の表示などができます。

ご利用の環境に合わせて、次の各ガイドラインに従ってください。

- フレームワーク (441ページ)
  - Grape (441ページ)
  - Rails (443ページ)
  - Sinatra (443ページ)
- Web サーバ (444ページ)
  - Passenger (444ページ)
  - Puma (447ページ)
  - Thin (450ページ)
  - Unicorn (451ページ)

関連項目

- Ruby エージェントのサポート対象テクノロジ (416ページ)

Ruby エージェントの YAML テンプレート

YAML 設定ファイルを使用して Ruby エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、YAML 設定の説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します：`/etc/contrast/contrast_security.yaml`

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
```

```
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
```

```
# ============================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
# ============================================================================
==
# agent:

  # \
# ============================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
# ============================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` for the agent to tag
    # logs with `!AM!` for the metrics tool.
    # metrics: true

  # \
# ============================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
# ============================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log
```

```
    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
```

```
  # \
================================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

    # Set to `true` for the agent to trigger garbage collection before
    # taking a heap dump to remove temproary objects from the dump.
    # clean: false

  # \
================================================================================
  # agent.ruby
  # The following properties apply to any Ruby agent-wide configurations.
  # \
================================================================================
  # ruby:

    # Allow the agent to track frozen Objects returned by
    # source methods. This configuration is on by default.
    # track_frozen_sources: NEEDS_TO_BE_SET

    # Allow the agent to track propagation through interpolated
    # Strings. This configuration is on by default.
    # interpolate: NEEDS_TO_BE_SET

    # Set a comma-separated string of rake tasks
    # in which to disable agent operation.
    # disabled_agent_rake_tasks: \
about,assets:clean,assets:clobber,assets:environment,assets:precompile,asset
s:precompile:all,db:create,db:drop,db:migrate:status,db:rollback,db:schema:c
ache:clear,db:schema:cache:dump,db:schema:dump,db:schema:load,db:seed,db:set
up,db:structure:dump,db:version,doc:app,log:clear,middleware,notes,notes:cus
tom,rails:template,rails:update,routes,secret,spec,spec:features,spec:reques
ts,spec:controllers,spec:helpers,spec:models,spec:views,spec:routing,spec:rc
ov,stats,test,test:all,test:all:db,test:recent,test:single,test:uncommitted,
time:zones:all,tmp:clear,tmp:create,webpacker:compile
```

```
# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
```

```
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET
```

```
    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
```

```
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
```

```
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET
```

```
  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
```

```
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET
```

```
    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
```

```
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` for the agent to tag
    # logs with `!AM!` for the metrics tool.
    # metrics: true

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET
```

```
      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
  # \
================================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

    # Set to `true` for the agent to trigger garbage collection before
    # taking a heap dump to remove temproary objects from the dump.
```

```
    # clean: false

  # \
================================================================================
  # agent.ruby
  # The following properties apply to any Ruby agent-wide configurations.
  # \
================================================================================
  # ruby:

    # Allow the agent to track frozen Objects returned by
    # source methods. This configuration is on by default.
    # track_frozen_sources: NEEDS_TO_BE_SET

    # Allow the agent to track propagation through interpolated
    # Strings. This configuration is on by default.
    # interpolate: NEEDS_TO_BE_SET

    # Set a comma-separated string of rake tasks
    # in which to disable agent operation.
    # disabled_agent_rake_tasks: \
about,assets:clean,assets:clobber,assets:environment,assets:precompile,asset
s:precompile:all,db:create,db:drop,db:migrate:status,db:rollback,db:schema:c
ache:clear,db:schema:cache:dump,db:schema:dump,db:schema:load,db:seed,db:set
up,db:structure:dump,db:version,doc:app,log:clear,middleware,notes,notes:cus
tom,rails:template,rails:update,routes,secret,spec,spec:features,spec:reques
ts,spec:controllers,spec:helpers,spec:models,spec:views,spec:routing,spec:rc
ov,stats,test,test:all,test:all:db,test:recent,test:single,test:uncommitted,
time:zones:all,tmp:clear,tmp:create,webpacker:compile

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
```

```
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
```

```
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
```

```
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
```

```
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET
```

```
  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost
```

```
  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

Ruby フレームワーク

アプリケーションフレームワークとはソフトウェアのライブラリで、アプリケーション開発を支援するための基礎的な構造を提供します。

お使いのフレームワークに合わせて、以下の各ガイドラインに従ってください。

Grape での設定

Grape フレームワークを使用している場合、Ruby エージェントを使用するためにアプリケーションを設定する必要があります。Ruby エージェントを使用するように設定した簡単なアプリケーションは、以下の例のようになります。

```
require 'grape'
require 'contrast-agent'

class App < Grape::API
  use Contrast::Agent::Middleware, true
end
```

config.ru で Grape を設定

クラスが拡張されていない場合、`Grape::API` または `config.ru` が、以下の設定で Rack アプリケーションを起動するデフォルトの方法です。

`config.ru`

```
# frozen_string_literal: true

require 'rack'
# Require Grape early for Framework support to detect it.
require 'Grape'
```

```
# example app.rb, could be any file implementing Grape
# endpoints and logic.
#
# For instance:
#
# # frozen_string_literal: true
#
# require 'Grape'
#
# class App
#   def initialize
#      @filenames = ['', '.html', 'index.html', '/index.html']
#      @rack_static = ::Rack::Static.new(
#        lambda { [404, {}, []] },
#        root: File.expand_path('../public', __dir__),
#        urls: ['/']
#      )
#    end
#
#    def self.instance
#      @instance ||= Rack::Builder.new do
#        use Rack::Cors do
#          allow do
#            origins '*'
#            resource '*', headers: :any, methods: :get
#          end
#        end
#
#        run App.new
#      end.to_app
#    end

#   def call(env)
#     # Grape::API impleted in API module:
#     API.call(env)
#      # handle response
#     ......
#    end
# end
#
require './app.rb'

# Contrast Agent needs to be required after Grape.
require 'contrast-agent'

# Example for requiring gems:
require 'bundler/setup'
Bundler.require(:default)

# Add Contrast Agent middleware to the rack stack:
use Contrast::Agent::Middleware, true

# Run Grape application:
run App.instance
```

アプリケーションを起動します。

```
bundle exec rackup
```

または

```
bundle exec rackup config.ru
```

Rails での設定

Rails を使用している場合、Ruby エージェントは Railtie として機能するため追加の設定は必要ありません。

Sinatra での設定

Sinatra フレームワークを使用している場合、Ruby エージェントを使用するためにアプリケーションを設定する必要があります。Ruby エージェントを使用するように設定した簡単なアプリケーションは、以下の例のようになります。

```
require 'sinatra'
require 'contrast-agent'

class App < Sinatra::Base
  use Contrast::Agent::Middleware, true
end
```

config.ru で Sinatra を設定

クラスが拡張されていない場合、`Sinatra::Base` または `config.ru` が、以下の設定で Rack アプリケーションを起動するデフォルトの方法です。

`config.ru`

```
# frozen_string_literal: true

# Require Sinatra early for Framework support to detect it.
require 'sinatra'

# example app.rb, could be any file implementing Sinatra
# endpoints and logic.
#
# For instance:
#
#   # frozen_string_literal: true
#
#   require 'sinatra'
#
#    get '/frank-says' do
#      'Put this in your pipe & smoke it!'
#    end
#
require './app.rb'

# Contrast Agent needs to be required after sinatra.
require 'contrast-agent'

# Example for requiring gems:
require 'bundler/setup'
Bundler.require(:default)

# Add Contrast Agent middleware to the rack stack:
use Contrast::Agent::Middleware, true
```

```
# Run Sinatra application:
run Sinatra::Application
```

アプリケーションを起動します。

```
bundle exec rackup
```

または

```
bundle exec rackup config.ru
```

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Ruby Web サーバ

Web サーバは、Web アプリケーションフレームワークをデプロイするための技術です。これらのサーバは、アプリケーションプロセスを管理し、HTTP リクエストを受信して、Web アプリケーションフレームワークに転送します。

お使いの Web サーバのガイドラインに従ってください。

- Passenger (444ページ)
- Puma (447ページ)
- Thin (450ページ)
- Unicorn (451ページ)

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Passenger での設定

Ruby エージェントがアプリケーションをタイムアウトさせてしまい、サーバが起動できなくなることがあります。これはサーバの設定によって防ぐことができます。

Passenger の設定は、スタンドアロンモードでも、HTTP サーバ(*Passenger + NGINX* や *Passenger + Apache*)の組合せでも指定できます。

スタンドアロンモードでの標準的な設定方法は、次の 3 つから選択できます。

1. コマンドライン引数 :

   ```
   $ passenger start --start-timeout 100
   ```
2. 環境変数 :

   ```
   $ env PASSENGER_START_TIMEOUT=100 passenger start
   ```
3. Passengerfile.json(アプリケーションディレクトリに配置) :

   ```
   {
     "start_timeout": "100"
   }
   ```

設定の順序(優先順位の高い順) :

- コマンドライン引数

- 環境変数
- `Passengerfile.json` ファイル

注記
大規模なデプロイメントの場合は例外で、コマンドライン引数と環境変数の両方の設定は、アプリケーションで事前に設定された値で上書きされます。

Passenger を NGINX や Apache と組合わせる場合は、NGINX や Apache の設定ファイルを使用します。この場合、`Passengerfile.json` は使用しません。

```
# example of an Nginx configuration file which also configures Passenger:

server {
    server_name yourserver.com;
    root /var/www/myapp/code/public;
    passenger_enabled on;
    passenger_ruby /usr/bin/ruby2.0;
    passenger_sticky_sessions on;
}
```

```
# example of an Apache configuration file which also configures Passenger:

<VirtualHost *:80>
    ServerName yourserver.com
    DocumentRoot /var/www/myapp/code/public
    PassengerStickySessions on

    <Directory /var/www/myapp/code/public>
        Allow from all
        Options -MultiViews
        Require all granted
    </Directory>
</VirtualHost>
```

タイムアウトの設定

- **最大リクエスト時間** - アプリケーションプロセスがリクエストを処理するのにかかる最大時間を秒単位で指定します。リクエストの処理にかかる時間がこの時間を超えた場合、アプリケーションプロセスは強制的にシャットダウンされ、次のリクエスト時に再起動されます。値が 0 の場合は、時間制限が無いことを意味します。これは Passenger エンタプライズ版のみ有効な設定です。
  - デフォルト値：0
  - コマンドライン：

    ```
    $ passenger start --max-request-time SECONDS
    ```
  - 環境変数：
  - `Passengerfile.json` ファイル：

    ```
    {
       "max_request_time": integer
    }
    ```

- **最大リクエストキュー時間** - 同時に処理されるリクエストが最大数に達すると、Passenger は全ての受信リクエストをキューに入れます。このオプションでは、リクエストがキューに保持される最大時間を指定します。キュー内のリクエストが指定された上限に達した場合、そのリクエストに対して「504 Gateway Timeout」エラーが返されます。値が 0 の場合は、キューの時間に制限が無いことを意味します。これは Passenger エンタプライズ版のみ有効な設定です。
  - デフォルト値：0
  - コマンドライン：

    ```
    $ passenger start --max-request-queue-time NUMBER
    ```
  - 環境変数：

    ```
    PASSENGER_MAX_REQUEST_QUEUE_TIME=integer
    ```
  - `Passengerfile.json` ファイル：

    ```
    {
       "max_request_queue_time": integer
    }
    ```
- **プールのアイドル時間** - アプリケーションプロセスの最大アイドル時間です。指定された秒数を超えてもアプリケーションプロセスが何のトラフィックも受信しない場合、メモリを節約するためにアプリケーションプロセスをシャットダウンします。この値を 0 に設定すると、手動で停止するかクラッシュが発生しない限り、アプリケーションプロセスはシャットダウンされません。この値を小さくすると、アプリケーションプロセスの起動がより頻繁に必要になります。
  - デフォルト値：300(5 分)
  - コマンドライン：

    ```
    $ passenger start --pool-idle-time SECONDS
    ```
  - 環境変数：

    ```
    PASSENGER_POOL_IDLE_TIME=integer
    ```
  - `Passengerfile.json` ファイル：

    ```
    {
       "pool_idle_time": integer
    }
    ```
- **最大プリロードアイドル時間** - 一定時間プリロードプロセスが何も処理をしていない場合に、自動的にシャットダウンするためのタイムアウトの時間です。このオプションは秒数で設定します。値が 0 の場合は、アイドル状態のタイムアウトが発生しないことを意味します。値を大きくすると、プリロードプロセスが長く存在することになり、メモリ使用量が若干増加する可能性があります。但し、プリロードサーバが実行されている限り、Ruby アプリケーションプロセスの起動に要する時間は、通常起動に必要とされる時間の約 10%で済みます。
  - デフォルト値：300(5 分)
  - コマンドライン：

    ```
    $ passenger start --max-preloader-idle-time SECONDS
    ```
  - 環境変数：

    ```
    PASSENGER_MAX_PRELOADER_IDLE_TIME=integer
    ```
  - `Passengerfile.json` ファイル：

    ```
    {
       "max_preloader_idle_time": integer
    }
    ```
- **起動タイムアウト** - アプリケーションの起動のタイムアウトです。アプリケーションプロセスがこのタイムアウト時間内に起動できなかった場合、シグナル(SIGKILL)で強制終了され、エラーがログに記録されます。
  - デフォルト値：90

- コマンドライン：

```
$ passenger start --start-timeout SECONDS;
```

- 環境変数：

```
PASSENGER_START_TIMEOUT=integer
```

- `Passengerfile.json` ファイル：

```
{
   "start_timeout": integer
}
```

フォーク

Passenger はプロセスマネージャのようなもので、自分のプロセス領域でアプリケーションを実行するのではなく、外部プロセスとして起動して、その管理を行います。これには、使用されていないプロセスのシャットダウンやクラッシュしたプロセスの再起動などが含まれます。アプリケーションのインスタンスは、プロセスと呼ばれます。Passenger はアプリケーションの起動や停止を行います。

Passenger がアプリケーションのインスタンスを起動すると、プロセスの生成が行われます。
Passenger でアプリケーションのプロセスを生成するには、次の 2 つの方法があります。

- **direct**：アプリケーションコードと Web フレームワークの完全なコピーをメモリ上に持つ新しい Ruby プロセスを生成します。この方法は、より多くのメモリを使用し、起動に時間がかかります。
- **smart**： Ruby アプリケーションの場合、この方法がデフォルトです。最初にプリロード(preloader)処理が実行されます。この処理は、config.ru ファイルを読み込むことで、Web フレームワークと一緒にアプリケーション全体をロードします。プリロード処理は、リクエスト処理には関与しません。新しいアプリケーションプロセスが必要になるたびに、子プロセスを生成します(fork システムコールを使用)。
  新しいフォークを作成するためのコマンドは以下のようになります。

```
$ bundle exec passenger start --min-instances 2
```

  これにより、Passenger はアプリケーションのインスタンスを 2 つ保持することになります。
  - デフォルト値：1
  - インスタンスの最大プール数のデフォルト：6
  - `Passengerfile.json` ファイル：

```
{
  "max_pool_size": 6
}
```

Passenger は、リクエストを受け付けると、リクエスト数が最も少ないプロセスにそのリクエストを渡します。プロセスが強制終了すると、Passenger が自動的にプロセスを再起動します。また、プロセスはトラフィックに応じて動的に拡張され、最大プール数までフォークにより新しいプロセスを生成します。

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Puma での設定

Ruby エージェントがアプリケーションをタイムアウトさせてしまい、サーバが起動できなくなることがあります。これは、サーバの設定によって防ぐことができます。

Puma の設定は、CLI を使用して `config/puma.rb` または `config.ru` ファイルで直接指定できます。

タイムアウトの設定

Contrast エージェントは、デフォルトの設定またはカスタム設定で動作するはずですが、最初のリクエスト処理でオーバーヘッドがかかります。 そのため、タイムアウトの設定時間を増やす必要がある場合があります。

重要
一部のオプションは、クラスタモードでのみ利用できます。 タイムアウトに利用できる全てのオプションは、puma/dsl.rb に記載されています。

- **persistent_timeout(seconds)** - Puma が持続的接続を閉じるまでのアイドル時間を定義します。seconds(秒数)には、整数を指定します。
- **first_data_timeout(seconds)** - データが受信されなかった場合に、TCP ソケットを開けている時間を定義します。seconds(秒数)には、整数を指定します。
- ***force_shutdown_after(val=:forever)** * - スレッドをシャットダウンする際に、スレッドが停止するまで待機する時間を定義します。秒数も指定することができますが、`:forever` もしくは`:immediately` を指定できます。
  - `:forever` - 値は-1 に設定されます。
  - `:immediately` - 値は 0 に設定されます。
  - `seconds` - タイムアウト値として、秒数を直接設定します。

注記
Puma は即時モードであっても、シャットダウンの前に常に数秒間は待機します。

以下のオプションは、クラスタモードでのみ利用できます。

- **worker_timeout(seconds)** - 指定したタイムアウト(秒数)内に、全てのワーカがマスタプロセスにチェックインしたかどうかを検証します。これはリクエストのタイムアウトではなく、プロセスの停止やハングアップを防ぐためのものです。この値を設定しても、時間の掛かるリクエストを防げるわけではありません。最小値は 6 秒、デフォルト値は 60 秒です。
- **worker_boot_timeout(seconds)** - ワーカが起動する際のデフォルトのタイムアウトを変更します。指定されていない場合は、デフォルトは worker_timeout の値になります。
- **worker_shutdown_timeout(seconds)** - ワーカがシャットダウンする際のタイムアウトを設定します。
- **wait_for_less_busy_worker(val=0.005)** - ソケットのリッスンを遅らせることにより、ビジーでないワーカにトラフィックをルーティングし、リクエストを最初に取得してもらうための待ち時間です。

注記
この設定は MRI でのみ機能します。他のインタプリタでは、この設定は無効です。

Puma では、最初に 2 つのデフォルトのタイムアウト値が設定されます。

- **DefaultWorkerTimeout** = 60
- **DefaultWorkerShutdownTimeout** = 30

全てのタイムアウトの設定を適用するには、Puma をクラスタモードに設定してください。

フォーク

クラスタモードが Puma 5 から導入され、Puma はマスタプロセスから直接ではなく、ワーカ 0 からワーカをフォークできるようになりました。

`preload_app` オプションと同様に、`fork_worker` オプションを使用すると、アプリケーションは一度だけ初期化され、コピーオンライト(COW)メモリを削減できます。

このモードにはいくつかの利点があり、まず第一に、段階的な再起動(phased restart)と互換性があります。マスタプロセス自体は最初にアプリケーションをプリロードしませんので、このモードによって段階的な再起動が有効になります。段階的な再起動の一環としてワーカ 0 がリロードされると、最初にアプリケーションの新しいコピーが初期化されます。次に、他のワーカが、この新しいプリロード済みのアプリケーションが既にある新しいワーカから、フォークされてリロードされます。

ヒント

段階的な再起動は、Puma クラスタ内の実行中のすべてのワーカを再起動します。これは、最初に古いワーカを終了させて、新しいワーカを起動し、新しいワーカが正常に起動するまで待ってから次のワーカに進みます。これを全てのワーカをに対して行います。 マスタプロセスは再起動されません。

この段階的な再起動は、ホットリスタートのように短時間で完了しつつ、クラスタのワーカを順に再起動することによりダウンタイムを最小限に抑えることができます。

もう一つの利点は、新しく追加された `refork` コマンドによるコピーオンライト(COW)の改善です。ワーカ 0 からリフォーク(refork)することで、ワーカ 0 以外の全てのワーカをリロードします。

このコマンドは、起動時に完全に初期化できないような大規模や複雑なアプリケーションでのメモリ使用率を改善できる可能性があります。これは、リフォークされたワーカが、すでにリクエストを処理している実行中のワーカとコピーオンライト(COW)メモリを共有できるためです。

また、リフォークは、一定の数のリクエスト(デフォルトは 1000)がワーカ 0 によって処理された場合にも、自動的に 1 度トリガーされます。自動リフォークされるまでのリクエスト数を設定するには、`fork_worker` に正の整数値(例えば、fork_worker 1000)を指定するか、もしくは 0 を指定して無効にします。

制限事項：

- クラスタモードは preload_app と互換性がありません。
- 新しいワーカを正常にフォークするために、ワーカ 0 はサーバをシャットダウンし、リクエストの処理を停止します。これにより、プロセス間で共有されてオープン中のファイル記述子やその他のグローバルな共有状態がなくなり、新しくフォークされたワーカ間でコピーオンライト(COW)の効率が最大化されます。このため、段階的な再起動/リフォーク中にクラスタの総処理能力が一時的に減少する場合があります。

fork_worker および refork コマンドについて説明しましたが、ほかにも以下のようなクラスタコマンド(fork ワーカ)があります。

- ***workers(count)** * - 実行するワーカプロセスの数。通常、使用可能な CPU コア数を設定します。 環境変数 WEB_CONCURRENCY に値が設定されている場合は、それがデフォルトです。それ以外の場合は 0 です。
- **before_fork(&block)** - マスタプロセスがワーカをフォークする前に実行(起動時に 1 回)するコード。プロセスが終了する前に、Puma が関与しないバックグラウンド操作が終了するのを待機する必要がある場合に指定します。

- **on_worker_boot(&block)** - アプリを起動する前にプロセスをセットアップするために、ワーカの起動時に実行するコード。
- **on_worker_shutdown(&block)** - ワーカがシャットダウンする直前(HTTP リクエストの処理が完了した後)に実行するコード。
- **on_worker_fork(&block)** - ワーカが起動する直前にマスタプロセスで実行するコード。ワーカのインデックスが引数として渡されます。
- **after_worker_fork(&block)** - ワーカが起動された後にマスタプロセスで実行するコード。ワーカのインデックスが引数として渡されます。
- **on_refork(&block)** - サーバがリクエストの受付けを一時的に停止した後、ワーカ 0 が他の全てのワーカをこのプロセスからリフォークする前に実行するコード(リフォークごとに 1 回呼び出されます)。コピーオンライト(COW)の効率を最大化するために追加のガベージコレクションのトリガとして使用できます。また、リモートサーバ (データベース、Redis など)との接続を閉じるためにも使用できます。
- **out_of_band(&block)** - ワーカがアイドル状態の時に不定期に実行されるコード。リクエスト処理が終わりワーカにビジー状態のスレッドが存在しなくなった直後に実行されます。このコードが終了するまで、ワーカは新しいリクエストを受付けません。これは、不定期のガベージコレクションを実行したり、レスポンスの後に実行する非同期タスクをスケジュールする場合に便利です。
- **fork_worker(after_request=1000)** - この設定を有効にすると、ワーカはマスタプロセスからではなくワーカ 0 からフォークされます。このオプションは、アプリケーションがフォークする前にプリロードされるため"preload_app"に似ていますが、段階的な再起動と互換性があります。このオプションで `refork` コマンドも有効になります。
- **nakayoshi_fork(enabled=true)** - これは少し異なるオプションですが、ワーカをフォークする前に Puma が 4 回 GC(ガベージコレクション)を実行します。起動とフォークの所要時間が長くなります。起動処理にかかる時間の詳細については、ログを参照してください。ほとんどのアプリケーションでは 1 秒以下になると考えられます。このフォーク方法は、笹田耕一氏と Aaron Patterson 氏の研究に基づいており、このオプションはプリロードを有効にしたクラスタモードの Puma のメモリ使用量を削減できる可能性があります。

注記
利用可能(Ruby 2.7 以降)な場合、`GC.compact` も実行されます。

MRI 以外の Ruby では推奨されません。

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Thin での設定

Ruby エージェントがアプリケーションをタイムアウトさせてしまい、サーバが起動できなくなることがあります。 これはサーバの設定によって防ぐことができます。

Thin は軽量の Web サーバで、クラスタをサポートし、タイムアウトと待機時間を設定するオプションを提供しています。Thin では、すべてのオプションを CLI コマンドの引数、または `config.yml` ファイルで設定できます。

タイムアウトの設定

wait オプションは、クラスタ内でサーバを再起動する際の最大待機時間です。

この**タイムアウト**オプションは、通信が切断されるまでデータ受信を待機する最大秒数です。

フォーク

Thin はクラスタをサポートしており、異なるポートで複数のサーバインスタンスを実行できます。

利用可能なオプションは次のとおりです。

```
cluster options:
-s, --servers NUM                Number of servers to start
-o, --only NUM                   Send command to only one server of the \
cluster
-C, --config FILE                Load options from config file
-O, --onebyone                   Restart the cluster one by one (only \
works with restart command)
-w, --wait NUM                   Maximum wait time for server to be \
started in seconds (use with -O)
```

クラスタと一緒に実行できる実験的なチューニングオプションもあります。

```
--threaded                    Call the Rack application in threads \
[experimental]
```

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Unicorn での設定

Unicorn はシングルスレッドで、マルチプロセスで動作します。Unicorn には、トラフィックに基づいて自動的にプロセス数を調整する機能がありません。これは、`unicorf.conf.rb` か `unicorn.conf.minimal.rb` ファイル、またはスクリプトを使用して設定する必要があります。

- `unicorn.conf.rb` の例：

```
# Define your root directory.
root = "/home/deployer/apps/gifroll/current"

# Define worker directory for Unicorn.
working_directory "/path/to/app/current"

# Define number of worker processes.
# Each forked OS process consumes additional memory.
worker_processes 4

# Define timeout for hanging workers before they are restarted.
timeout 30

# Location of PID file.
pid "/path/to/app/shared/pids/unicorn.pid"

# Define log paths:

# Allow redirecting $stderr to a given path. Unlike doing this from the \
shell,
# this allows the Unicorn process to know the path being written to and \
rotates
# the file if it is used for logging.
stderr_path "#{root}/log/unicorn.log"

# Same as stderr_path, except for $stdout. Not many Rack applications \
write
# to $stdout, but any that do will have their output written here.
stdout_path "#{root}/log/unicorn.log"
```

```
# Loads Rails before forking workers for better worker spawn time.
# Preloading your application reduces the startup time of individual
# Unicorn worker_processes and allows you to manage the external \
connections
# of each individual worker using the before_fork and after_fork calls.
#
# Please check if other external connections work properly with
# Unicorn forking. Many popular gems (dalli memcache client, Redis) will \
have
# compatibility confirmation with Unicorn and the process model.
# Check the gem documentation for more information.
preload_app true

# When enabled, Unicorn will check the client connection by writing the
# beginning of the HTTP headers before calling the application. This will
# prevent calling the application for clients who have disconnected while
# their connection was queued.
check_client_connection false

# Enable a local variable to guard against running a hook (before_fork, \
after_fork)
# multiple times
run_once = true

# For example, use of before_fork and after_fork:
#
# POSIX Signals are a form of interprocess communication, and signal
# events or state changes.
# QUIT: Signals a process to exit, but creates a core dump.
# TERM: Tells a process to terminate, but allows the process
# to clean up after itself.
#
# Unicorn uses the QUIT signal to indicate a graceful shutdown.
# The master process receives it and sends it to all workers, telling \
them to
# shutdown after any in-flight request.
before_fork do |server, worker|
  Signal.trap 'TERM' do
    puts 'Unicorn master intercepting TERM and sending myself QUIT \
instead'
    Process.kill 'QUIT', Process.pid
  end

  # You may want to execute code in the master process, before the forking
  # begins, to deal with operations that causes changes in state.
  # You need them to run once:
  if run_once
    # do_something_once_here ...
    run_once = false # prevent from firing again
  end
end

after_fork do |server, worker|
  Signal.trap 'TERM' do
    puts 'Unicorn worker intercepting TERM and doing nothing. Wait for \
```

```
master to send QUIT'
  end
  # ...
end

# For more information, check the Unicorn Configurator: https://msp-
greg.github.io/unicorn/Unicorn/Configurator.html
```

- `unicorn.conf.minimal.rb` の例：

```
listen 2007 # by default Unicorn listens on port 8080
worker_processes 2 # this should be >= nr_cpus
pid "/path/to/app/shared/pids/unicorn.pid"
stderr_path "/path/to/app/shared/log/unicorn.log"
stdout_path "/path/to/app/shared/log/unicorn.log"
```

フォークの設定

Unicorn には、利用可能な CPU コアをより有効に活用するためのマルチプロセスアーキテクチャがあります。起動時に、Unicorn マスタプロセスはアプリケーションコードをロードし、マスタプロセスからアプリケーションコードを継承するワーカを生成します。リクエストはワーカによってのみ処理され、マスタプロセスでは処理されません。オペレーティングシステムのネットワークスタックは、受信したリクエストをキューに入れ、ワーカに配布されます。

Unicorn はユーザのリクエストを切らずに、クラッシュしたワーカを再起動するように設計されています。ワーカがリクエストタイムアウトに達すると、マスタプロセスはワーカを `kill -9` で終了させ、新しいプロセスに置き換えます。ワーカプロセスの数やリクエストのタイムアウトは、設定できます。

設定	説明
`worker_processes (nr)`	`worker_processes`(ワーカプロセス)の数を `nr` に指定します。各ワーカプロセスは、一度に 1 つのクライアントだけを処理します。`SIGTTIN` または `SIGTTOU` のシグナルをマスタプロセスに送信することで、設定ファイルをリロードすることなく、実行時にこの値を増減できます。 シグナルについての詳細は、Unicorn のサイトを参照ください。
`Unicorn::Configurator#after_fork`	`after_fork`(フォーク後)に特定のブロックに対してフックを指定します。
`Unicorn::Configurator#before_fork`	`before_fork`(フォーク前)に特定のブロックに対してフックを指定します。
`Unicorn::Configurator#preload_app(bool)` `ObjectEnabling`	この設定は、ワーカプロセスをフォークする前にアプリケーションをプリロードします。これは、コピーオンライトに適した GC を使用する際にメモリを節約できますが、ソケットのようなリソースがマスタプロセスによってロード時に開かれて複数の子プロセスで共有される場合、問題が生じる可能性があります。

タイムアウトの設定

変数	説明	デフォルトの値
timeout 30	ワーカプロセスのタイムアウトを 30 秒に設定します。デフォルトは、60 秒です。タイムアウトの設定により、この制限を超えたワーカを終了させることができます。このタイムアウトは、マスタプロセスによって強制実行され、ワーカプロセスによるスケジューリングの制限の影響を受けません。	60 秒
delay integer	ソケットのバインドの試行を待機する時間を秒単位で指定します。	0.5 秒

ヒント
Unicorn は、NGINX と連携する場合に多くの設定があります。

APM ツール の設定
次の APM ツールは Unicorn をサポートしています。

- **Scout** には、Unicorn とインテグレーションするためのクラスがあります。
- **New Relic**
- **AppDynamics**

関連項目

- Ruby エージェントの設定 (419ページ)
- Ruby エージェントのサポート対象テクノロジ (416ページ)

Ruby エージェントの YAML テンプレート
YAML 設定ファイルを使用して Ruby エージェントを設定する場合には、以下のテンプレートをご利用ください(YAML 設定については、YAML 設定の説明 (63ページ)を参照してください)。

YAML 設定ファイルは、デフォルトの場所に配置します：`/etc/contrast/contrast_security.yaml`

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast
```

```
  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET
```

```
    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` for the agent to tag
    # logs with `!AM!` for the metrics tool.
    # metrics: true

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
```

```
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET

      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING
```

```
  # \
================================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
  # \
================================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

    # Set to `true` for the agent to trigger garbage collection before
    # taking a heap dump to remove temproary objects from the dump.
    # clean: false

  # \
================================================================================
  # agent.ruby
  # The following properties apply to any Ruby agent-wide configurations.
  # \
================================================================================
  # ruby:

    # Allow the agent to track frozen Objects returned by
    # source methods. This configuration is on by default.
    # track_frozen_sources: NEEDS_TO_BE_SET

    # Allow the agent to track propagation through interpolated
    # Strings. This configuration is on by default.
    # interpolate: NEEDS_TO_BE_SET

    # Set a comma-separated string of rake tasks
    # in which to disable agent operation.
    # disabled_agent_rake_tasks: \
about,assets:clean,assets:clobber,assets:environment,assets:precompile,asset
s:precompile:all,db:create,db:drop,db:migrate:status,db:rollback,db:schema:c
ache:clear,db:schema:cache:dump,db:schema:dump,db:schema:load,db:seed,db:set
```

```
up,db:structure:dump,db:version,doc:app,log:clear,middleware,notes,notes:cus
tom,rails:template,rails:update,routes,secret,spec,spec:features,spec:reques
ts,spec:controllers,spec:helpers,spec:models,spec:views,spec:routing,spec:rc
ov,stats,test,test:all,test:all:db,test:recent,test:single,test:uncommitted,
time:zones:all,tmp:clear,tmp:create,webpacker:compile

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
```

```
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:
```

```
    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
```

```
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:
```

```
      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
==========================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
==========================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET
```

```
  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
```

```
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:
```

```
    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Certificate
    # PEM file for communication with the Contrast UI.
    # cert_file: NEEDS_TO_BE_SET

    # Set the absolute or relative path to the Key PEM
    # file for communication with the Contrast UI.
    # key_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate with
    # the Contrast web interface over a proxy. Set value to
    # `false` if you don't want to use the proxy. If no value is
    # indicated, the presence of a valid **contrast.proxy.host**
    # and **contrast.proxy.port** will enable the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
```

```
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Override the name of the process the agents uses in logs.
    # progname: Contrast Agent

    # Set to `true` for the agent to tag
    # logs with `!AM!` for the metrics tool.
    # metrics: true

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

    # \
================================================================================
    # agent.security_logger.syslog
    # Define the following properties to set Syslog values. If the \
properties
    # are not defined, the agent uses the Syslog values from the Contrast \
UI.
    # \
================================================================================
    # syslog:

      # Set to `true` to enable Syslog logging.
      # enable: NEEDS_TO_BE_SET
```

```
      # Set the IP address of the Syslog server
      # to which the agent should send messages.
      # ip: NEEDS_TO_BE_SET

      # Set the port of the Syslog server to
      # which the agent should send messages.
      # port: NEEDS_TO_BE_SET

      # Set the facility code of the messages the agent sends to Syslog.
      # facility: 19

      # Set the log level of Exploited attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_exploited: ALERT

      # Set the log level of Blocked attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked: NOTICE

      # Set the log level of Blocked At Perimeter
      # attacks. Value options are `ALERT`, `CRITICAL`,
      # `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_blocked_perimeter: NOTICE

      # Set the log level of Probed attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_probed: WARNING

      # Set the log level of Suspicious attacks. Value options are `ALERT`,
      # `CRITICAL`, `ERROR`, `WARNING`, `NOTICE`, `INFO`, and `DEBUG`.
      # severity_suspicious: WARNING

  # \
================================================================================
  # agent.heap_dump
  # The following properties are used to trigger heap dumps from within
  # the agent to snapshot the behavior of instrumented applications.
  # \
================================================================================
  # heap_dump:

    # Set to `true` for the agent to automatically
    # take heap dumps of the instrumented application.
    # enable: false

    # Set the location to which to save the heap dump files. If relative,
    # the path is determined based on the process' working directory.
    # path: contrast_heap_dumps

    # Set the amount of time to wait, in milliseconds,
    # after agent startup to begin taking heap dumps.
    # delay_ms: 10_000

    # Set the amount of time to wait, in milliseconds, between each heap \
dump.
```

```
    # window_ms: 10_000

    # Set the number of heap dumps to take before disabling this feature.
    # count: 5

    # Set to `true` for the agent to trigger garbage collection before
    # taking a heap dump to remove temproary objects from the dump.
    # clean: false

  # \
================================================================================
  # agent.ruby
  # The following properties apply to any Ruby agent-wide configurations.
  # \
================================================================================
  # ruby:

    # Allow the agent to track frozen Objects returned by
    # source methods. This configuration is on by default.
    # track_frozen_sources: NEEDS_TO_BE_SET

    # Allow the agent to track propagation through interpolated
    # Strings. This configuration is on by default.
    # interpolate: NEEDS_TO_BE_SET

    # Set a comma-separated string of rake tasks
    # in which to disable agent operation.
    # disabled_agent_rake_tasks: \
about,assets:clean,assets:clobber,assets:environment,assets:precompile,asset
s:precompile:all,db:create,db:drop,db:migrate:status,db:rollback,db:schema:c
ache:clear,db:schema:cache:dump,db:schema:dump,db:schema:load,db:seed,db:set
up,db:structure:dump,db:version,doc:app,log:clear,middleware,notes,notes:cus
tom,rails:template,rails:update,routes,secret,spec,spec:features,spec:reques
ts,spec:controllers,spec:helpers,spec:models,spec:views,spec:routing,spec:rc
ov,stats,test,test:all,test:all:db,test:recent,test:single,test:uncommitted,
time:zones:all,tmp:clear,tmp:create,webpacker:compile

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Apply a list of labels to libraries. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Value options are `ALL`, `SOME`, or `NONE`.
  # stacktraces: ALL

  # \
================================================================================
  # assess.sampling
  # Use the following properties to control sampling in the agent.
  # \
================================================================================
  # sampling:

    # Set to `true` to enable sampling.
    # enable: false

    # This property indicates the number of requests
    # to analyze in each window before sampling begins.
    # baseline: 5

    # This property indicates that every *nth*
    # request after the baseline is analyzed.
    # request_frequency: 10

    # This property indicates the duration for which a sample set is valid.
    # window_ms: 180_000

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:
```

```
    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.rules
  # Use the following properties to set simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Protect rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Protect rules. The rules must be formatted as a comma-delimited list.
    # disabled_rules: NEEDS_TO_BE_SET

    # \
================================================================================
    # protect.rules.bot-blocker
    # Use the following selection to configure if the
    # agent blocks bots. Set to `true` to enable blocking.
    # \
================================================================================
    # bot-blocker:

      # Set to `true` for the agent to block known bots.
      # enable: false

    # \
================================================================================
    # protect.rules.sql-injection
    # Use the following settings to configure the sql-injection rule.
    # \
================================================================================
```

```
    # sql-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or off.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.cmd-injection
    # Use the following properties to configure
    # how the command injection rule works.
    # \
================================================================================
    # cmd-injection:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.path-traversal
    # Use the following properties to configure
    # how the path traversal rule works.
    # \
================================================================================
    # path-traversal:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.method-tampering
    # Use the following properties to configure
    # how the method tampering rule works.
    # \
================================================================================
    # method-tampering:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
```

```
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.reflected-xss
    # Use the following properties to configure how
    # the reflected cross-site scripting rule works.
    # \
================================================================================
    # reflected-xss:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

    # \
================================================================================
    # protect.rules.xxe
    # Use the following properties to configure
    # how the XML external entity works.
    # \
================================================================================
    # xxe:

      # Set the mode of the rule. Value options are
      # `monitor`, `block`, `block_at_perimeter`, or `off`.
      #
      # Note - If a setting says, "if blocking is enabled",
      # the setting can be `block` or `block_at_perimeter`.
      #
      # mode: off

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
```

```
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
```

```
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

Ruby のテレメトリ

Ruby エージェントは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、エージェントを組み込んだアプリケーションでエージェントのセンサーが最初にロードされた時に収集され、その後も定期的に(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切 (964ページ)であると考えています。 このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、Contrast に安全に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能で収集されるのは、以下のデータです。

エージェントのバージョン	データ
Ruby 4.13	エージェントのバージョン
	オペレーティングシステムとバージョン
	Ruby のバージョン
	アプリケーションのフレームワークとバージョン
	Web サーバとバージョン
	Contrast インスタンスが SaaS 版かオンプレミス版であるか

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、telemetry.ruby.contrastsecurity.com に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

Go エージェント

Go エージェントは、ライブラリや脆弱性のレポートのために Go アプリケーションに組み込まれるソースコードリライタ(rewriter)です。これにより、アプリケーションを構成するソースコードとライブラリを実行時に把握することができます。

注記

ソースコードリライタとして、アプリケーションに Go エージェントをインストールするには、アプリケーションのビルド環境へのアクセスが必要になります。

Go エージェントは現在、Contrast Assess と Contrast SCA のみに対応しています。

次のステップとして、以下のことができます。

- Go エージェントをインストールする (477ページ)
- Go エージェントのサポート対象テクノロジを確認する (476ページ)

Go エージェントのサポート対象テクノロジ

このエージェントでは、以下のテクノロジをサポートしています。

テクノロジ	サポート対象バージョン	備考
言語のバージョン	• 1.20 • 1.21	Contrast は、Go リリースのサポートポリシーに従います。このポリシーは、直近にリリースされた 2 つのメジャーバージョンをサポートします。Go 言語のサポート対象バージョンは、新しいメジャーバージョンがリリースされると変わります。アプリケーションの依存関係は、`go.mod` ファイルで指定する必要があります。 **サポート対象外：** • 1.19：サポート対象だった最後のバージョンは 5.12.0 • 1.18：サポート対象だった最後のバージョンは 4.2.0 • 1.17：サポート対象だった最後のバージョンは 3.6.0 • 1.16：サポート対象だった最後のバージョンは 3.6.0 • 1.15：サポート対象だった最後のバージョンは 1.12.0
プラットフォーム	• darwin/amd64 • darwin/arm64 • linux/amd64 • linux/arm64	Contrast では、一覧にあるプラットフォームをターゲットにして `contrast-go` をビルド・テストしています。 その他のプラットフォームに対して `contrast-go` をクロスコンパイルすることもできますが、互換性は保証されません。
依存関係の管理方法	Go mod	アプリケーションは、Go Modules の一部である必要があります。`go mod init` を実行することで、アプリケーションで Go Modules を初期化できます。
プロトコルスタック	• net/http • github.com/valyala/fasthttp v1.46.0 以降 • google.golang.org/grpc v1.17.0 以降	エージェントは、プロトコルの実装に依存して、アプリケーションの動きを理解します。サポートされていないプロトコルパッケージがアプリケーションで使用されている場合、エージェントは脆弱性やルート情報を報告することができません。

テクノロジ	サポート対象バージョン	備考
ルーターとフレームワーク	• github.com/gofiber/fiber v2.40.0 以降 • github.com/go-chi/chi/v5 • github.com/julienschmidt/httprouter • github.com/gin-gonic/gin v1 以降 • github.com/go-openapi/swag	フレームワークの基礎となる HTTP 実装がサポートされている限り、この一覧にないフレームワークでもエージェントは動作します。ただし、一部の機能が使用できなくなるか、精度が低くなる可能性があります。 例えば、認識できないルータに登録されたルートを、エージェントが正しく検出しない可能性があります。 ご利用のフレームワークがサポート対象のフレームワークの一覧にない場合やサポートが必要な場合は、Contrast サポートにお問い合わせください。
データベースのサポート	`database/sql`	エージェントは、データベースドライバとして登録されたデータベースをサポートするために、標準ライブラリの `database/sql` パッケージに組み込むことができます。SQLDrivers の例を参照ください。

Go エージェントのインストール

Go エージェントは、`contrast-go` と呼ばれるツールを使用して、ビルド時にアプリケーションに組み込まれます。Go エージェントが組み込まれたアプリケーションを実行すると、Go エージェントが自動的に起動し、アプリケーションの実行が監視されて、脆弱性が検出されます。

ヒント

`contrast-go` のコマンドライン引数で利用可能なフラグの一覧を表示するには、`contrast-go -h` を入力します。

手順

1. インストーラで、`contrast-go` をインストールします。

   ```
   go run github.com/contrast-security-oss/contrast-go-installer@latest \
   latest
   ```
2. `contrast-go` を使用して、アプリケーションをビルドします。

   ```
   contrast-go build -o output-name-of-application
   ```
3. Go エージェントの YAML テンプレート (481ページ)や環境変数を使用して、Go エージェントを設定 (480ページ)します。
4. 手順 2 で生成された実行可能ファイルを使用して、アプリケーションを実行します。
5. アプリケーションの疎通やテストを行います。
6. Contrast Web インターフェイスを使用すると、脆弱性やライブラリの使用状況など、エージェントによって報告される検出結果を確認できます。
   デフォルトでは、アプリケーション名はアプリケーションの Go モジュールに基づいています。アプリケーションの一覧で検索を使用すれば、アプリケーションをすぐに見つけることができます。

コンテナに Go エージェントをインストール

Go エージェントのコンテナへのインストールは、基本的に標準のインストール手順と同じですが、インストールがコンテナ内で行われる点と、ベストプラクティスに従い環境変数を使用して Contrast の認証情報を設定する必要がある点が異なります。

Go エージェントをコンテナにインストールする場合、環境変数を使用するのが最も安全な方法です。コンテナは QA や本番システムから移行されることが多いため、コンテナ定義に認証情報をハードコーディングしないことをお勧めします。

ヒント
Dockerfile で Go エージェントを使用するサンプルアプリケーションを参照したい場合は、Go Test Bench プロジェクトをご覧ください。

開始する前に

- コンテナや関連ソフトウェアの仕組みを基本的に理解している必要があります。
- 必要に応じて、お客様の環境に合わせて手順を調整してください。

Go アプリケーションをインストール、ビルド、実行

1. Go がインストールされていることを確認します。
2. 以下のコマンドで、`contrast-go` をインストールします。

   ```
   RUN github.com/contrast-security-oss/contrast-go-installer@latest latest
   ```
3. 通常の `go build` コマンドを `contrast-go build` に置き換えて、アプリケーションをビルドします。このステップによって、Contrast が組み込まれた実行ファイルがビルドされます。

   ```
   RUN contrast-go build ./app
   ```
4. 環境変数で、エージェントを設定 (480ページ)します。

Docker の例

Docker コンテナに Go アプリケーションをインストール、ビルド、および実行する方法の例として、以下を参照ください。

```
# Step 1: Install Go. You can use a different base image than the one \
shown in
# this example.
FROM golang:1.21 AS builder
WORKDIR /build

COPY . .

# Step 2: This step installs contrast-go and makes sure it's in your $PATH \
so
# you can use it in the next step.
RUN go run github.com/contrast-security-oss/contrast-go-installer@latest \
latest

# Step 3: This step is your normal build step, but uses contrast-go \
instead of
# go. This step doesn't replace Go; it just wraps it so that it can add
# instrumentation during the build process.
RUN contrast-go build ./app

# Optional: Move the finished build to a new container.
# Not required, but nice to have!
FROM alpine:latest
COPY --from=builder /build/app .

# Step 4: Configure the agent using enviornment variables.

ENTRYPOINT ["./app"]
```

環境変数の設定例

> **注記**
> Contrast エージェント設定エディタ (64ページ)の Export(エクスポート)オプションを使用すると、Contrast の認証情報の環境変数を簡単に作成できます。
>
> クラウドプロバイダの使用時に環境変数を設定するプロセスでは、一般的には、シークレットマネージャを使用し、それらのシークレットの値を環境変数にリンクすることになります。

例えば、次のコマンドを使用して、コンテナを構築することができます。

```
docker build -t my-app-image
```

そして、コンテナの実行時には次のコマンドを使用します。

```
docker run -p 3000:3000 --name my-app-instance \
-e "CONTRAST__API__URL=your-ts-url" \
-e "CONTRAST__API__API_KEY=your-api-key" \
-e "CONTRAST__API__SERVICE_KEY=your-service-key" \
-e "CONTRAST__API__USER_NAME=your-user-name" \
my-app-image
```

関連項目

- Kubernetes と Contrast エージェント
- AWS Fargate と Contrast エージェント

Go エージェントを直接ダウンロードしてインストール

Go エージェントをインストールするには：

1. `https://pkg.contrastsecurity.com` からエージェントをダウンロードします。`contrast-go` 実行可能ファイルは、Mac および Linux オペレーティングシステム用に直接ダウンロードできます。利用可能なバージョンは、go-agent-release のサイトで確認できます。`<version>`の箇所を利用するバージョン番号に置き換えます。または、最新バージョンを利用する場合は `latest` に置き換えます。
   - **Mac の場合：**

     ```
     wget https://pkg.contrastsecurity.com/go-agent-release/<version>/
     darwin-amd64/contrast-go
     ```

     または

     ```
     curl -L https://pkg.contrastsecurity.com/go-agent-release/<version>/
     darwin-amd64/contrast-go > contrast-go
     ```
   - **Linux の場合：**

     ```
     wget https://pkg.contrastsecurity.com/go-agent-release/<version>/linux-
     amd64/contrast-go
     ```

     または

```
curl -L https://pkg.contrastsecurity.com/go-agent-release/<version>/
linux-amd64/contrast-go > contrast-go
```

2. ダウンロードしたら、エージェントが実行可能であることを確認します。例：

```
chmod u+x contrast-go
```

3. 依存関係を表すために必要な go.mod ファイルがアプリケーションにあることを確認してください。アプリケーションのソースディレクトリで、以下のコマンドを実行します。

```
go mod init
```

4. アプリケーションをビルドします。

```
./contrast-go build -o output-name-of-application
```

5. Go エージェントの YAML テンプレート (481ページ)や環境変数を使用して、Go エージェントを設定 (480ページ)します。
6. 前述の手順でビルドした実行可能ファイルを使用して、アプリケーションを起動します。
7. アプリケーションの疎通やテストを行います。
8. Contrast でアプリケーションが認識されていることを確認します。

Go エージェントの設定

エージェントを設定するには、contrast_security.yaml と呼ばれる YAML ファイルに設定を指定します。最も簡単な方法は、Contrast Web インターフェイスから設定ファイルをダウンロード (49ページ)することです。このファイルには、組織に必要な設定があらかじめ入力されています。

必要な設定は、以下のとおりです。

```
api:
  api_key: <key>
  service_key: <key>
  user_name: <key>
```

また、Contrast Web インターフェイスで**組織の設定 > エージェント**にて、エージェントキーを見つける (61ページ)ことができます。

ヒント

YAML ファイルと環境変数の両方に値を設定した場合、エージェントは優先順位 (62ページ)に従って環境変数を使用します。

YAML ファイルと環境変数の両方に値を設定した場合、エージェントは優先順位 (62ページ)に従って環境変数を使用します。

Go 設定ファイルの場所

Go エージェントの設定については、以下のディレクトリで contrast_security.yaml ファイルが見つかるまで検索されます。

- 現在のディレクトリ
- /etc/contrast/go/
- /etc/contrast/
- **Darwin**： $HOME/Library/Preferences/contrast/
- **Darwin**： $HOME/Library/Preferences/contrast/go/
- **Linux**： $XDG_CONFIG_DIR/contrast/
- **Linux**： $XDG_CONFIG_DIR/contrast/go/

Go エージェントの YAML テンプレート

YAML テンプレートには、Go エージェントで使用可能な全ての設定が含まれています。(YAML 設定については、こちらの説明 (63ページ)を参照してください。)

また、Contrast エージェント設定エディタ (64ページ)を使用して、設定を検索して、カスタムの設定ファイルを作成することもできます。

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==


# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
```

```
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
```

```
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

  # \
================================================================================
  # agent.go
  # The following properties apply to any Go agent-wide configurations.
  # \
================================================================================
  # go:
```

```
    # \
================================================================================
    # agent.go.preview
    # Enable opt-in Go agent features.
    # \
================================================================================
    # preview:

      # Enable Assess gRPC sources.
      # grpc: false

    # \
================================================================================
    # agent.go.profile
    # Enable Go agent self-profiling features.
    # \
================================================================================
    # profile:

      # Enable CPU profiling for running application.
      # cpu: false

      # Enable memory profiling for running application.
      # mem: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
```

```
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

# \
================================================================================
==
```

```
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET

  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
```

```
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:

  # Override the reported server name.
  # name: localhost

  # Override the reported server path.
  # path: NEEDS_TO_BE_SET

  # Override the reported server type.
  # type: NEEDS_TO_BE_SET

  # Set the environment directly to override the default set
  # by the Contrast UI. This allows the user to configure the
  # environment dynamically at startup rather than manually
  # updating the Server in the Contrast UI themselves afterwards.
  #
  # Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
  # For example, `PRODUCTION` registers this Server as
  # running in a `PRODUCTION` environment, regardless of the
  # organization's default environment in the Contrast UI.
  #
  # environment: NEEDS_TO_BE_SET

  # Apply a list of labels to the server. Labels
  # must be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET
```

```
# \
================================================================================
==
# Use the properties in this YAML file to configure a Contrast agent.
# Go to https://docs.contrastsecurity.com/en/order-of-precedence.html
# to determine the order of precedence for configuration values.
# \
================================================================================
==
```

```
# Use this setting if you want to temporarily disable a Contrast agent.
# Set to `true` to enable the agent; set to `false` to disable the agent.
# enable: true

# \
================================================================================
==
# api
# Use the properties in this section to connect the agent to the Contrast \
UI.
# \
================================================================================
==
api:

  # ********************** REQUIRED **********************
  # Set the URL for the Contrast UI.
  url: https://app.contrastsecurity.com/Contrast

  # ********************** REQUIRED **********************
  # Set the API key needed to communicate with the Contrast UI.
  api_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the service key needed to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  service_key: NEEDS_TO_BE_SET

  # ********************** REQUIRED **********************
  # Set the user name used to communicate with the Contrast
  # UI. It is used to calculate the Authorization header.
  user_name: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.certificate
  # Use the following properties for communication
  # with the Contrast UI using certificates.
  # \
================================================================================
  # certificate:

    # If set to `false`, the agent will ignore the
    # certificate configuration in this section.
    # enable: true

    # Set the absolute or relative path to a CA for communication
    # with the Contrast UI using a self-signed certificate.
    # ca_file: NEEDS_TO_BE_SET

  # \
================================================================================
  # api.proxy
  # Use the following properties for communication
  # with the Contrast UI over a proxy.
```

```
  # \
================================================================================
  # proxy:

    # Set value to `true` for the agent to communicate
    # with the Contrast web interface over a proxy. Set
    # value to `false` if you don't want to use the proxy.
    # enable: NEEDS_TO_BE_SET

    # Set the proxy host. It must be set with port and scheme.
    # host: localhost

    # Set the proxy port. It must be set with host and scheme.
    # port: 1234

    # Set the proxy scheme (e.g., `http` or
    # `https`). It must be set with host and port.
    # scheme: http

    # Set the URL for your Proxy Server. The URL form is `scheme://
host:port`.
    # url: NEEDS_TO_BE_SET

    # Set the proxy user.
    # user: NEEDS_TO_BE_SET

    # Set the proxy password.
    # pass: NEEDS_TO_BE_SET

# \
================================================================================
==
# agent
# Use the properties in this section to control the way and frequency
# with which the agent communicates to logs and the Contrast UI.
# \
================================================================================
==
# agent:

  # \
================================================================================
  # agent.logger
  # Define the following properties to set logging values.
  # If the following properties are not defined, the
  # agent uses the logging values from the Contrast UI.
  # \
================================================================================
  # logger:

    # Enable diagnostic logging by setting a path to a log file.
    # While diagnostic logging hurts performance, it generates
    # useful information for debugging Contrast. The value set here
    # is the location to which the agent saves log output. If no
    # log file exists at this location, the agent creates a file.
```

```
    #
    # Example - `/opt/Contrast/contrast.log` creates a log in the
    # `/opt/Contrast` directory, and rotates it automatically as needed.
    #
    # path: ./contrast_agent.log

    # Set the the log output level. Valid options are
    # `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: INFO

    # Set to `true` to redirect all logs to
    # `stdout` instead of the file system.
    # stdout: false

  # \
================================================================================
  # agent.security_logger
  # Define the following properties to set security
  # logging values. If not defined, the agent uses the
  # security logging (CEF) values from the Contrast UI.
  # \
================================================================================
  # security_logger:

    # Set the file to which the agent logs security events.
    # path: ./contrast/security.log

    # Set the log level for security logging. Valid options
    # are `ERROR`, `WARN`, `INFO`, `DEBUG`, and `TRACE`.
    # level: ERROR

  # \
================================================================================
  # agent.go
  # The following properties apply to any Go agent-wide configurations.
  # \
================================================================================
  # go:

    # \
================================================================================
    # agent.go.preview
    # Enable opt-in Go agent features.
    # \
================================================================================
    # preview:

      # Enable Assess gRPC sources.
      # grpc: false

    # \
================================================================================
    # agent.go.profile
    # Enable Go agent self-profiling features.
    # \
```

```
================================================================================
    # profile:

      # Enable CPU profiling for running application.
      # cpu: false

      # Enable memory profiling for running application.
      # mem: false

# \
================================================================================
==
# inventory
# Use the properties in this section to override the inventory features.
# \
================================================================================
==
# inventory:

  # Set to `false` to disable inventory features in the agent.
  # enable: true

  # Set to `false` to disable library analysis.
  # analyze_libraries: true

# \
================================================================================
==
# assess
# Use the properties in this section to control Assess.
# \
================================================================================
==
# assess:

  # Include this property to determine if the Assess
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # Apply a list of labels to vulnerabilities and preflight
  # messages. Labels must be formatted as a comma-delimited list.
  # Example - `label1, label2, label3`
  #
  # tags: NEEDS_TO_BE_SET

  # \
================================================================================
  # assess.rules
  # Use the following properties to control simple rule configurations.
  # \
================================================================================
  # rules:

    # Define a list of Assess rules to disable in the agent. To view a
```

```
    # list of rule names, in Contrast go to user menu > Policy Management >
    # Assess rules. The rules must be formatted as a comma-delimited list.
    #
    # Example - Set `reflected-xss,sql-injection` to disable
    # the reflected-xss rule and the sql-injection rule.
    #
    # disabled_rules: NEEDS_TO_BE_SET

# \
================================================================================
==
# protect
# Use the properties in this section to override Protect features.
# \
================================================================================
==
# protect:

  # Include this property to determine if the Protect
  # feature should be enabled. If this property is not
  # present, the decision is delegated to the Contrast UI.
  # enable: false

  # \
================================================================================
  # protect.probe_analysis
  # Use the settings in this section to
  # control the behavior of probe analysis.
  # \
================================================================================
  # probe_analysis:

    # Set to `false` to disable probe analysis.
    # enable: true

# \
================================================================================
==
# application
# Use the properties in this section for
# the application(s) hosting this agent.
# \
================================================================================
==
# application:

  # Override the reported application name.
  #
  # Note - On Java systems where multiple, distinct applications may be
  # served by a single process, this configuration causes the agent to \
report
  # all discovered applications as one application with the given name.
  #
  # name: NEEDS_TO_BE_SET
```

```
  # Override the reported application path.
  # path: NEEDS_TO_BE_SET

  # Add the name of the application group with which this
  # application should be associated in the Contrast UI.
  # group: NEEDS_TO_BE_SET

  # Add the application code this application should use in the Contrast UI.
  # code: NEEDS_TO_BE_SET

  # Override the reported application version.
  # version: NEEDS_TO_BE_SET

  # Apply labels to an application. Labels must
  # be formatted as a comma-delimited list.
  # Example - `label1,label2,label3`
  #
  # tags: NEEDS_TO_BE_SET

  # Define a set of `key=value` pairs (which conforms to RFC 2253) for
  # specifying user-defined metadata associated with the application. The
  # set must be formatted as a comma-delimited list of `key=value` pairs.
  # Example - `business-unit=accounting, office=Baltimore`
  #
  # metadata: NEEDS_TO_BE_SET

  # Provide the ID of a session which already exists in the Contrast
  # UI. Vulnerabilities discovered by the agent are associated with
  # this session. If an invalid ID is supplied, the agent will be
  # disabled. This option and `application.session_metadata` are
  # mutually exclusive; if both are set, the agent will be disabled.
  # session_id: NEEDS_TO_BE_SET

  # Provide metadata which is used to create a new session ID in the
  # Contrast UI. Vulnerabilities discovered by the agent are associated with
  # this new session. This value should be formatted as `key=value` pairs
  # (conforming to RFC 2253). Available key names for this configuration
  # are branchName, buildNumber, commitHash, committer, gitTag, repository,
  # testRun, and version. This option and `application.session_id` are
  # mutually exclusive; if both are set the agent will be disabled.
  # session_metadata: NEEDS_TO_BE_SET

# \
================================================================================
==
# server
# Use the settings in this section to set metadata for the server
# hosting this agent. Contrast recognizes common, supported server
# names, paths, types and environments. Doing this may require a new
# server or license, and it may affect functionality of some features.
# \
================================================================================
==
# server:
```

```
# Override the reported server name.
# name: localhost

# Override the reported server path.
# path: NEEDS_TO_BE_SET

# Override the reported server type.
# type: NEEDS_TO_BE_SET

# Set the environment directly to override the default set
# by the Contrast UI. This allows the user to configure the
# environment dynamically at startup rather than manually
# updating the Server in the Contrast UI themselves afterwards.
#
# Valid values include `QA`, `PRODUCTION` and `DEVELOPMENT`.
# For example, `PRODUCTION` registers this Server as
# running in a `PRODUCTION` environment, regardless of the
# organization's default environment in the Contrast UI.
#
# environment: NEEDS_TO_BE_SET

# Apply a list of labels to the server. Labels
# must be formatted as a comma-delimited list.
# Example - `label1,label2,label3`
#
# tags: NEEDS_TO_BE_SET
```

Contrast サービス

重要

Contrast サービスは、 旧バージョンの Node エージェントと Python エージェント (Python バージョン 5.19.0 より前)で動作します。新しいバージョンのエージェントでは、Contrast サービスを必要としない、より高性能でネイティブな解析を使用します。この変更が発生するバージョンを確認するには、各エージェントのドキュメントを参照してください。

Contrast サービスはスタンドアロンの実行可能ファイルで、マルチプロセスの動的言語エージェント (Node.js エージェント、Python エージェント)と Contrast サーバ間の通信を可能にします。Contrast サービスは、Contrast 側の設定オプションをエージェントに渡したり、エージェントからの情報を集約して Contrast へ送り返します。

Contrast サービスは、サポート対象のアーキテクチャ用にコンパイルされています。

- Linux 64 ビット
- Macintosh 64 ビット
- Windows 64 ビット

Node.js エージェントと Python エージェントには、サービスが一緒にパッケージ化されており、エージェントを有効にしたアプリケーションが起動すると自動的に開始します。Go エージェントには、サービスはパッケージ化されておらず、自動的に開始しません。Go エージェントでサービスを機能させる

には、サービスをインストールして設定し、起動する必要があります。Node.js や Python エージェントで実行する場合でも、同様の設定を行い、より管理しやすくすることもできます。

Contrast サービスのインストール

インストール方法は、お使いのシステムによって異なります。

- **Linux**：システムのパッケージマネージャで Contrast サービスをインストールします。
- **Debian**：コマンドを使用して、適切な Debian リポジトリからインストールします。

1. 使用している Ubuntu の CODENAME(コードネーム)を調べます。

   ```
   grep VERSION_CODENAME /etc/os-release
   ```
2. その CODENAME で以下のコマンドを更新し、コマンドを実行します。

   ```
   curl https://pkg.contrastsecurity.com/api/gpg/key/public | sudo apt-
   key add -
   echo "deb https://pkg.contrastsecurity.com/debian-public/ CODENAME \
   contrast" | sudo tee /etc/apt/sources.list.d/contrastc.list
   ```
3. Contrast サービスをインストールします。

   ```
   sudo apt-get update && sudo apt-get install contrast-service
   ```
4. Contrast サービスを設定 (495ページ)します。

- **RPM(Red Hat Package Manager)**：以下のコマンドを使用して、Contrast の yum リポジトリからインストールします。

1. リポジトリを使用するよう、システムを設定します。

   ```
   OSREL=$(rpmquery -E "%{rhel}")
   sudo -E tee /etc/yum.repos.d/contrast.repo << EOF
   [contrast]
   name=contrast repo
   baseurl=https://pkg.contrastsecurity.com/rpm-public/centos-$OSREL/
   gpgcheck=0
   enabled=1
   EOF
   ```
2. Contrast サービスをインストールします。

   ```
   yum install contrast-service
   ```
3. Contrast サービスを設定 (495ページ)します。

ヒント

`contrast-service package` を削除するには、`apt-get remove contrast-service` または `yum remove contrast-service` を実行してください。

Contrast サービスの設定

Contrast サービスには接続パラメータが事前に設定されていません。YAML 設定ファイルを使用してサービスを設定する必要があります。

Contrast サービスをシステムサービスとしてインストールした場合、Contrast サービスは`/etc` ディレクトリ にある YAML 設定ファイルによって制御されます。サービスが、同じサーバ上の他のアプリケーションと同じ YAML 設定ファイル(`contrast_security.yaml`)を共有することがよくありますので、全ての接続値(ソケット名やポート番号など)が一致していることを確認してください。

アプリケーション用の設定ファイルがアプリケーションの作業ディレクトリにまだインストールされていない場合、YAML 設定ファイルの場所によって、同じサーバ上のエージェントと YAML ファイルを共有するかが決まります。

- **共有しない**場合は、YAML 設定ファイル `/etc/contrast/webserver/contrast_security.yaml` に置きます。
- **共有する**場合は、YAML 設定ファイルを`/etc/contrast/contrast_security.yaml` に置きます。

デフォルトの YAML 設定ファイルが、Contrast サービスの Linux パッケージと一緒に`/etc/contrast/webserver/contrast_security.yaml` にインストールされます。 このテンプレートファイルには、エージェントの設定に最低限必要な項目のプレースホルダがありますが、以下の項目は更新してください。

- **api**：API のプロパティを設定します。この設定によって、Contrast サービスがどのように Contrast サーバに接続するかがが決まります。
- **agent**：エージェントに関連する設定を構成するセクションの最上位レベルです。
  - **service**：このセクションのオプションは、エージェントと Contrast サービス間の通信に影響します。接続の設定は、Contrast サービスと、そのサービスと通信するエージェント間で同一である必要があります。
    - **socket**：ローカルの Unix ソケットへのパス(例、*/tmp/contrast.sock*)です。
    - **host および port**：オプションとして、ソケット(socket)の代わりに、ホストとポートで接続するよう Contrast サービスを設定できます。
    - **grpc**：(Go と Node.js エージェントにのみ適用)エージェントからサービスへの通信に gRPC を使用するには、"true"を設定します。この設定は任意ですが、エージェントのパフォーマンスを若干向上させることができます。

設定に問題があったり、正しい値が指定されていない場合、または Contrast サービスが Contrast に接続できない場合などは、`/var/log/contrast/service.log` で接続の失敗をトラブルシューティングできます。

Contrast サービスのインストール

インストール方法は、お使いのシステムによって異なります。

- **Linux**：システムのパッケージマネージャで Contrast サービスをインストールします。
- **Debian**：コマンドを使用して、適切な Debian リポジトリからインストールします。

  1. 使用している Ubuntu の `CODENAME`(コードネーム)を調べます。

     ```
     grep VERSION_CODENAME /etc/os-release
     ```
  2. その `CODENAME` で以下のコマンドを更新し、コマンドを実行します。

     ```
     curl https://pkg.contrastsecurity.com/api/gpg/key/public | sudo apt-
     key add -
     echo "deb https://pkg.contrastsecurity.com/debian-public/ CODENAME \
     contrast" | sudo tee /etc/apt/sources.list.d/contrastc.list
     ```
  3. Contrast サービスをインストールします。

     ```
     sudo apt-get update && sudo apt-get install contrast-service
     ```
  4. Contrast サービスを設定 (495ページ)します。
- **RPM(Red Hat Package Manager)**：以下のコマンドを使用して、Contrast の yum リポジトリからインストールします。

  1. リポジトリを使用するよう、システムを設定します。

     ```
     OSREL=$(rpmquery -E "%{rhel}")
     sudo -E tee /etc/yum.repos.d/contrast.repo << EOF
     [contrast]
     ```

```
name=contrast repo
baseurl=https://pkg.contrastsecurity.com/rpm-public/centos-$OSREL/
gpgcheck=0
enabled=1
EOF
```

2. Contrast サービスをインストールします。

```
yum install contrast-service
```

3. Contrast サービスを設定 (495ページ)します。

ヒント

`contrast-service package` を削除するには、`apt-get remove contrast-service` または `yum remove contrast-service` を実行してください。

Contrast サービスの設定

Contrast サービスには接続パラメータが事前に設定されていません。YAML 設定ファイルを使用してサービスを設定する必要があります。

Contrast サービスをシステムサービスとしてインストールした場合、Contrast サービスは`/etc` ディレクトリ にある YAML 設定ファイルによって制御されます。サービスが、同じサーバ上の他のアプリケーションと同じ YAML 設定ファイル(`contrast_security.yaml`)を共有することがよくありますので、全ての接続値(ソケット名やポート番号など)が一致していることを確認してください。

アプリケーション用の設定ファイルがアプリケーションの作業ディレクトリにまだインストールされていない場合、YAML 設定ファイルの場所によって、同じサーバ上のエージェントと YAML ファイルを共有するかが決まります。

- **共有しない**場合は、YAML 設定ファイル `/etc/contrast/webserver/contrast_security.yaml` に置きます。
- **共有する**場合は、YAML 設定ファイルを`/etc/contrast/contrast_security.yaml` に置きます。

デフォルトの YAML 設定ファイルが、Contrast サービスの Linux パッケージと一緒に`/etc/contrast/webserver/contrast_security.yaml` にインストールされます。 このテンプレートファイルには、エージェントの設定に最低限必要な項目のプレースホルダがありますが、以下の項目は更新してください。

- **api**：API のプロパティを設定します。この設定によって、Contrast サービスがどのように Contrast サーバに接続するかがが決まります。
- **agent**：エージェントに関連する設定を構成するセクションの最上位レベルです。
  - **service**：このセクションのオプションは、エージェントと Contrast サービス間の通信に影響します。接続の設定は、Contrast サービスと、そのサービスと通信するエージェント間で同一である必要があります。
    - **socket**：ローカルの Unix ソケットへのパス(例、*/tmp/contrast.sock*)です。
    - **host および port**：オプションとして、ソケット(socket)の代わりに、ホストとポートで接続するよう Contrast サービスを設定できます。
    - **grpc**：(Go と Node.js エージェントにのみ適用)エージェントからサービスへの通信に gRPC を使用するには、"true"を設定します。この設定は任意ですが、エージェントのパフォーマンスを若干向上させることができます。

設定に問題があったり、正しい値が指定されていない場合、または Contrast サービスが Contrast に接続できない場合などは、`/var/log/contrast/service.log` で接続の失敗をトラブルシューティングできます。

Contrast エージェントオペレータ(Kubernetes オペレータ)

Contrast エージェントオペレータは、Kubernetes クラスタおよび OpenShift クラスタ内で実行される標準の Kubernetes オペレータで、既存のワークロードへの Contrast エージェントの組み込み、組み込まれたエージェントの設定、およびエージェントのアップグレードを自動化します。

まずは、エージェントオペレータをインストール (500ページ)するか、導入例としてエージェントオペレータのチュートリアル (500ページ)をご覧ください。

オペレータは、宣言的な Kubernetes ネイティブのリソースタイプを使用して設定します。リソースタイプについては、エージェントオペレータの設定 (509ページ)に記載しています。

注記

Contrast エージェントオペレータは、オープンソースプロジェクトです。コードのレビューや開発への貢献は、こちらへ。

セキュリティポリシー

Contrast エージェントオペレータは、様々なセキュリティポリシーでクラスタをサポートします。インストールを行う前に、これらを理解しておくことをお勧めします。最新のポリシーは、こちらを参照してください。

関連項目

- エージェントオペレータのサポート対象テクノロジ (498ページ)
- エージェントオペレータのネットワーク要件 (499ページ)
- エージェントオペレータのテレメトリ (514ページ)
- エージェントオペレータの最小設定 (505ページ)

エージェントオペレータのサポート対象テクノロジ

Kubernetes/OpenShift のサポート

サポート対象テクノロジの最新の情報は、こちらをご覧ください。

Kubernetes のバージョン	OpenShift のバージョン	オペレータのバージョン	サポート終了
v1.28		v0.14.0+	2024-10-28
v1.27		v0.14.0+	2024-06-28
v1.26	v4.13	v0.14.0+	2024-02-28
v1.25	v4.12	v0.8.0+	2023-10-28
v1.24	v4.11	v0.4.0+	2023-09-29
v1.23	v4.10	v0.1.0+	2023-02-28
v1.22	v4.9	v0.1.0+	2022-10-28
v1.21	v4.8	v0.1.0+	2022-06-28

- Contrast エージェントオペレータは、アップストリームである Kubernetes コミュニティのサポートポリシーに従います。サポート終了日については、Kubernetes のリリースページに記載されています。
- OpenShift のサポートは、含まれる Kubernetes のバージョンに依存します。例えば、OpenShift v4.10 は Kubernetes v1.23 を使用しており、Contrast では 2023 年 2 月 28 日までサポートします。OpenShift に含まれる Kubernetes バージョンについては、Red Hat のサポート記事を参照してください。

- Contrast エージェントオペレータは、Linux の amd64 ホスト上での実行のみをサポートし、互換性のないノードでのスケジューリングは拒否されます。また、Contrast エージェントが他のプラットフォームをサポートしている場合でも、エージェントオペレータは Linux の amd64 ホスト上で実行されているワークロードの組込のみをサポートします。Windows や arm64 での Kubernetes のサポートを希望される場合は、Contrast サポートにお問い合わせください。

エージェントタイプ

エージェント	エージェントタイプ	サポート状況	互換性情報
.NET Core	dotnet-core	サポート対象	サポート対象の.NET Core テクノロジ (225ページ)
Java	java	サポート対象	サポート対象の Java テクノロジ (75ページ)
Node.js	nodejs または nodejs-protect	サポート対象	サポート対象の Node.js テクノロジ (285ページ)
PHP	php	ベータ版	サポート対象の PHP テクノロジ (342ページ)

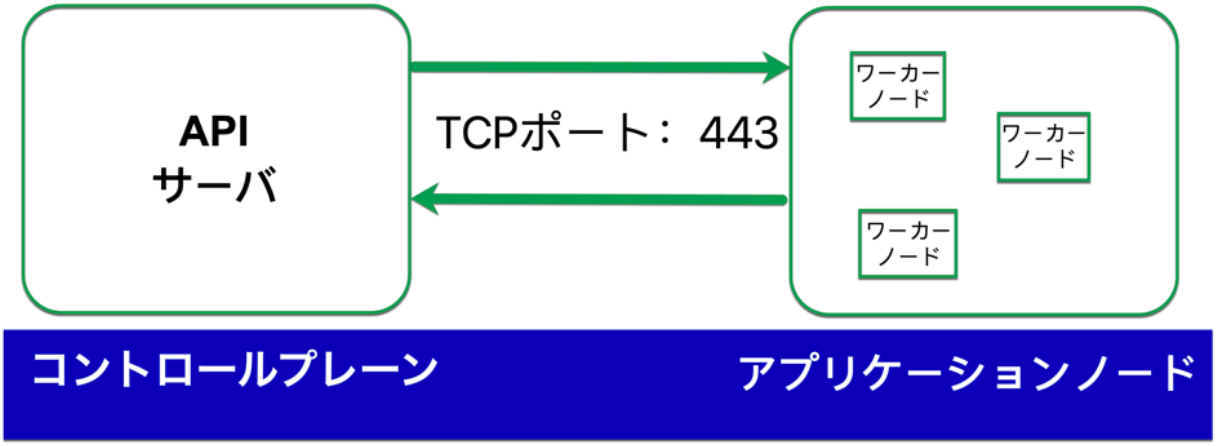
注記

- PHP アプリケーションでのエージェントの組み込みはベータ版です。ベータ版は機能が変更されたり、予期しない動作をしたりする可能性があります。この機能を利用することで、お客様は Contrast ベータ版利用規約 (964ページ)に同意することになります。
- Node.js エージェントを組み込むと、エージェントを組み込まれたアプリケーションの起動時間が大幅に増加する場合があります。起動時間が許容できない場合は、コンパイル時にエージェントを組み込むことをお勧めします。コンパイル時に Node.js エージェントを組み込む場合は、オペレータによるランタイム時の組み込みは無効化して下さい。詳しくは、リライタ CLI (338ページ)を参照してください。

エージェントオペレータのネットワーク要件

エージェントオペレータは、ネットワークの制約によって機能が妨げられない環境にインストールする必要があります。

最適な機能を維持するためには、ポート 443 でコントロールプレーンとワーカーノード間の双方向通信を設定する必要があります。



エージェントオペレータのインストール

Contrast は、クラスタに直接適用できる単一のインストール用 YAML ファイルを提供し、適切なデフォルト値を提供します。お客様の環境に応じて追加で修正することも可能です。

カスタムレジストリ

エアギャップ環境でエージェントオペレータを使用する場合(もしくは、Docker Hub を使用できない場合)は、Contrast カスタムレジストリのサンプルを参考にしてください。

手順

1. クラスタ管理者として kubectl(Kubernetes)または oc(OpenShift)を使用して以下を実行し、オペレータマニフェストを適用します。

```
kubectl apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

```
oc apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

このマニフェストによって以下が行われます。
- contrast-agent-operator ネームスペースを作成
- オペレータのデプロイメントワークロードをインストール
- 必要なカスタムリソース定義(CRD)をインストール
- 最低限必要な権限でロールベースのアクセス制御(RBAC)を設定
- オペレータを Admission Webhook へ登録

> **注記**
> デフォルトの contrast-agent-operator 以外のネームスペースにインストールすることも可能ですが、その場合はデプロイメントマニフェストの修正が必要になります。

2. オペレータマニフェストを適用した後、クラスタが集約されるのを待ちます。

```
kubectl -n contrast-agent-operator wait pod --for=condition=ready --
selector=app.kubernetes.io/name=operator,app.kubernetes.io/part-
of=contrast-agent-operator --timeout=30s
```

```
oc -n contrast-agent-operator wait pod --for=condition=ready --
selector=app.kubernetes.io/name=operator,app.kubernetes.io/part-
of=contrast-agent-operator --timeout=30s
```

3. Wait コマンドが成功すると、オペレータは設定可能 (505ページ)な状態になります。

関連項目

- エージェントオペレータのネットワーク要件 (499ページ)
- エージェントオペレータのチュートリアル (500ページ)
- エージェントオペレータの最小設定 (505ページ)
- エージェントオペレータのテレメトリ (514ページ)

エージェントオペレータのチュートリアル

開始する前に

ここでは、Vanilla Kubernetes を使用して、Contrast エージェントオペレータのインストールと、クラスタ管理者としてサンプルワークロードを組み込む方法について説明します。

OpenShift を使用してこの例を実行するには、Kubernetes のコマンドを同等の OpenShift コマンドに置き換えてください。すべてのコマンドは、Bash などのターミナル内で実行します。

Kubernetes および関連するソフトウェアの仕組みについて、基本的な理解があることを前提としています。必要に応じて、お客様の環境に合わせて手順を調整してください。

手順 1：オペレータのインストール

オペレータをインストールするには、オペレータのマニフェストをクラスタに適用する必要があります。Contrast は、クラスタに直接適用できる単一のインストール用 YAML ファイルを提供し、適切なデフォルト値を提供します。お客様の環境に応じて追加で修正することも可能ですが、その場合は Kustomize などの設定管理ツールを使用することをお勧めします。

注記

このインストール用 YAML ファイルによって、`contrast-agent-operator` ネームスペースが作成されてインストールが行われます。このネームスペースを後で使用します。

クラスタが集約されるのをしばらく待つと、オペレータのステータスが `Running` になります。

```
% kubectl -n contrast-agent-operator get pods
```

出力：

```
NAME                                        READY   STATUS    RESTARTS   AGE
contrast-agent-operator-57f5cfbf7-9svtt     1/1     Running   0          27s
contrast-agent-operator-57f5cfbf7-fp4vp     1/1     Running   0          39s
```

オペレータを設定する準備ができました。

手順 2：オペレータの設定

クラスタのワークロードを組み込む前に、まずオペレータを設定する必要があります。

Kubernetes のシークレットを使用して、接続の認証キーを保存します。なお、以下を実行すると、`default-agent-connection-secret` という名前で Secret が作成され、`contrast-agent-operator` ネームスペースに作成されます。

```
% kubectl -n contrast-agent-operator \
        create secret generic default-agent-connection-secret \
        --from-literal=apiKey=TODO \
        --from-literal=serviceKey=TODO \
        --from-literal=userName=TODO
```

出力：

```
secret/default-agent-connection-secret created
```

注記

`TODO` を、ご利用の Contrast サーバで該当する値に置き換えてください。エージェントキーの検索 (61ページ)で、Contrast Web インタフェースからエージェントキーを取得する方法について説明しています。

接続の設定を完了するには、ClusterAgentConnection が必要です。以下を実行すると、`contrast-agent-operator` ネームスペースで ClusterAgentConnection が作成され、前述の手順で作成した Secret のキー値を参照します。

```
% kubectl apply -f - <<EOF
apiVersion: agents.contrastsecurity.com/v1beta1
kind: ClusterAgentConnection
metadata:
  name: default-agent-connection
  namespace: contrast-agent-operator
spec:
  template:
    spec:
      url: https://app.contrastsecurity.com/Contrast
      apiKey:
        secretName: default-agent-connection-secret
        secretKey: apiKey
      serviceKey:
        secretName: default-agent-connection-secret
        secretKey: serviceKey
      userName:
        secretName: default-agent-connection-secret
        secretKey: userName
EOF
```

出力：

```
clusteragentconnection.agents.contrastsecurity.com/default-agent-
connection created
```

注記
ClusterAgentConnection の名前は重要ではなく、任意の名前を付けることができます。

これでオペレータが設定され、既存のワークロードにエージェントを組み込めるようになりました。

手順 3：ワークロードへの組み込み

この例では、Deployment ワークロードを使用する ASP.NET Core のサンプルアプリケーションに Contrast .NET Core エージェントを組み込む方法について説明します。

まず、ASP.NET Core のサンプルアプリケーションをクラスタにデプロイします。以下を実行すると、`default` のネームスペースに Deployment が作成されます。

```
% kubectl apply -f - <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: spring-petclinic
  namespace: default
  labels:
    arbitrary-label: arbitrary-value
spec:
  selector:
```

```
    matchLabels:
      app: spring-petclinic
  template:
    metadata:
      labels:
        app: spring-petclinic
    spec:
      containers:
        - image: contrastsecuritydemo/spring-petclinic:1.5.1
          name: spring-petclinic
EOF
```

出力：

```
deployment.apps/spring-petclinic created
```

クラスタが集約されるのをしばらく待つと、デプロイされたワークロードのステータスが `Running` になります。

```
% kubectl -n default get pods
```

出力：

```
NAME                                READY   STATUS    RESTARTS   AGE
spring-petclinic-77d97bdbd5-ts2cz   1/1     Running   0          15d
```

次に、AgentInjector 設定エンティティを使用して、.NET Core エージェントを組み込むようにオペレータを設定します。この場合、AgentInjector は、前述の Deployment がデプロイされたのと同じネームスペース(この場合は `default`)に作成する必要があります。

```
% kubectl apply -f - <<EOF
apiVersion: agents.contrastsecurity.com/v1beta1
kind: AgentInjector
metadata:
  name: spring-petclinic-injector
  namespace: default
spec:
  type: java
  selector:
    labels:
      - name: arbitrary-label
        value: arbitrary-value
EOF
```

出力：

```
agentinjector.agents.contrastsecurity.com/spring-petclinic-injector \
configured
```

`spring-petclinic-app` の Pod のログを見ると、Contrast の.NET Core エージェントがアプリケーションに組み込まれていることを確認できます。

```
% kubectl -n default logs Deployment/spring-petclinic
Defaulted container "spring-petclinic" out of: spring-petclinic, contrast-
init (init)
Picked up JAVA_TOOL_OPTIONS: -javaagent:/opt/contrast/contrast-agent.jar
[Contrast] Wed Dec 20 21:47:23 GMT 2023 Loading pre-packaged configuration
[Contrast] Wed Dec 20 21:47:23 GMT 2023 Couldn't find pre-packaged \
```

```
configuration.
[Contrast] Wed Dec 20 21:47:23 GMT 2023 Starting Contrast (build 6.1.1) \
Pat. 8,458,789 B2
[Contrast] Wed Dec 20 21:47:24 GMT 2023 Contrast logger configuration \
errors will be logged to stderr
[Contrast] Wed Dec 20 21:47:26 GMT 2023 Copyright: 2023 Contrast Security, \
Inc
[Contrast] Wed Dec 20 21:47:26 GMT 2023 Contact: \
support@contrastsecurity.com
[Contrast] Wed Dec 20 21:47:26 GMT 2023 License: Commercial
[Contrast] Wed Dec 20 21:47:26 GMT 2023 NOTICE: This Software and the \
patented inventions embodied within may only be used as part of
[Contrast] Wed Dec 20 21:47:26 GMT 2023 Contrast Security's commercial \
offerings. Even though it is made available through public
[Contrast] Wed Dec 20 21:47:26 GMT 2023 repositories, use of this Software \
is subject to the applicable End User Licensing Agreement
[Contrast] Wed Dec 20 21:47:26 GMT 2023 found at https://
www.contrastsecurity.com/enduser-terms-0317a  or as otherwise agreed between
[Contrast] Wed Dec 20 21:47:26 GMT 2023 Contrast Security and the End \
User. The Software may not be reverse engineered, modified,
[Contrast] Wed Dec 20 21:47:26 GMT 2023 repackaged, sold, redistributed or \
otherwise used in a way not consistent with the End User
[Contrast] Wed Dec 20 21:47:26 GMT 2023 License Agreement.
[Contrast] Wed Dec 20 21:47:26 GMT 2023 The Contrast Java agent collects \
usage data in order to help us improve compatibility and security coverage.
[Contrast] Wed Dec 20 21:47:26 GMT 2023 The data is anonymous and does not \
contain application data. It is collected by Contrast and is never shared.
[Contrast] Wed Dec 20 21:47:26 GMT 2023 You can opt-out of telemetry by \
setting the CONTRAST_AGENT_TELEMETRY_OPTOUT environment variable to 'true' \
or '1'
[Contrast] Wed Dec 20 21:47:26 GMT 2023 Read more about Contrast Java \
agent telemetry: https://docs.contrastsecurity.com/en/java-telemetry.html
[Contrast] Wed Dec 20 21:47:27 GMT 2023 Effective instructions: \
Assess=true, Protect=false, Observe=false
[Contrast] Wed Dec 20 21:47:27 GMT 2023 Contrast logger configuration \
errors will be logged to stderr
[Contrast] Wed Dec 20 21:47:41 GMT 2023 Starting JVM [18888ms]
              |\      _,,,--,,_
             /,`.-'`'   ._  \-;;,_
  _______ __|,4-  ) )_   .;.(__`'-'__     ___ __    _ ___ _______
 |       | '---''(_/._)-'(_\_)   |   |   |   |  |  | |   |       |
 |    _  |    ___|_     _|       |   |   |   |   |_| |   |       | __ _ _
 |   |_| |   |___  |   | |       |   |   |   |       |   |       | \ \ \ \
 |    ___|    ___| |   | |      _|   |___|   |  _    |   |      _|  \ \ \ \
 |   |   |   |___  |   | |     |_|       |   | | |   |   |     |_    ) ) ) )
 |___|   |_______| |___| |_______|_______|___|_|  |__|___|_______|  / / / /
 ==========================================================/_/_/_/
 :: Built with Spring Boot :: 1.5.4.RELEASE
 2023-12-20 21:47:45.651  INFO 1 --- [           main] \
 o.s.s.petclinic.PetClinicApplication     : Starting PetClinicApplication \
 v1.5.1 on spring-petclinic-77d97bdbd5-ts2cz with PID 1 (/spring-petclinic/
 spring-petclinic-1.5.1.jar started by root in /spring-petclinic)
```

手順 4：オペレータのアンインストール(任意)

クラスタを元の状態に復元するには、まず既存の AgentInjector を削除します。

```
% kubectl -n default delete agentinjector spring-petclinic-injector
```

出力：

```
agentinjector.agents.contrastsecurity.com "spring-petclinic-injector" \
deleted
```

その後、オペレータは、組み込まれた全てのワークロードを組み込み前の状態に復元します。クラスタが集約されたら、オペレータを安全に削除できます。

```
% kubectl delete -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

出力：

```
namespace "contrast-agent-operator" deleted
customresourcedefinition.apiextensions.k8s.io "agentconfigurations.agents.co
ntrastsecurity.com" deleted
customresourcedefinition.apiextensions.k8s.io "agentconnections.agents.contr
astsecurity.com" deleted
customresourcedefinition.apiextensions.k8s.io "agentinjectors.agents.contras
tsecurity.com" deleted
customresourcedefinition.apiextensions.k8s.io "clusteragentconfigurations.ag
ents.contrastsecurity.com" deleted
customresourcedefinition.apiextensions.k8s.io "clusteragentconnections.agent
s.contrastsecurity.com" deleted
serviceaccount "contrast-agent-operator-service-account" deleted
clusterrole.rbac.authorization.k8s.io "contrast-agent-operator-service-
role" deleted
clusterrolebinding.rbac.authorization.k8s.io "contrast-agent-operator-
service-role-binding" deleted
service "contrast-agent-operator" deleted
deployment.apps "contrast-agent-operator" deleted
poddisruptionbudget.policy "contrast-agent-operator" deleted
mutatingwebhookconfiguration.admissionregistration.k8s.io "contrast-web-
hook-configuration" deleted
```

関連項目

- エージェントオペレータのインストール (500ページ)
- エージェントオペレータの最小設定 (505ページ)
- エージェントオペレータの設定 (509ページ)

エージェントオペレータの最小設定

オペレータのすべての設定は、カスタムリソース定義(CRD)によって定義された Kubernetes ネイティブの設定エンティティを使用することで処理されます。CRD はオペレータと共にデプロイされ、オペレータの設定エンティティの処理方法が定義されます。

Visual Studio Code の Kubernetes 拡張機能などのツールは、クラスタ内でエンティティを正しい構文で作成するのに役立ちます。

完全なスキーマは、エージェントオペレータの設定 (509ページ)に記載しています。本項では、必要な最小限の設定ついてのみ説明し、すべての設定は記載していません。

最小限の設定

最小限の設定では、3 つのマニフェストが必要です。

1. 1 つ目は、標準の Kubernetes の Secret に Contrast サーバインスタンスへの認証に必要な接続キーを含めます。この Secret は、ClusterAgentConnection エンティティと同じネームスペースにデプロイする必要があります。指定するエージェントキーは、エージェントキーの検索 (61ページ)で確認できます。

```
apiVersion: v1
kind: Secret
metadata:
  name: default-agent-connection-secret
  namespace: contrast-agent-operator
type: Opaque
stringData:
  apiKey: TODO
  serviceKey: TODO
  userName: TODO
```

2. 2 つ目は、ClusterAgentConnection の設定エンティティです。ClusterAgentConnection には、クラスタ内のエージェントのデフォルトの接続設定を指定し、接続の認証キーが含まれる前述の Secret をマップします。セキュリティ上、ClusterAgentConnection エンティティは、オペレータを使用するのと同じネームスペースに配置する必要があります。この例では、デフォルトのネームスペース `contrast-agent-operator` がカスタマイズされていないことを前提にしています。

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: ClusterAgentConnection
metadata:
  name: default-agent-connection
  namespace: contrast-agent-operator
spec:
  template:
    spec:
      url: https://app.contrastsecurity.com/Contrast
      apiKey:
        secretName: default-agent-connection-secret
        secretKey: apiKey
      serviceKey:
        secretName: default-agent-connection-secret
        secretKey: serviceKey
      userName:
        secretName: default-agent-connection-secret
        secretKey: userName
```

3. 3 つ目に必要なのは、AgentInjector の設定エンティティです。AgentInjector は、AgentInjector がデプロイされるネームスペース内で `metadata.labels` などのワークロードのラベルを使用して、自動でエージェントを組み込む対象となるワークロードを選択します。

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: AgentInjector
metadata:
  name: dotnet-hello-world
  namespace: default
spec:
  type: dotnet-core
  selector:
    labels:
      - name: app
        value: dotnet-hello-world
```

このマニフェストの例では、Contrast エージェントオペレータは、デフォルトのネームスペースで `app = dotnet-hello-world` というラベルが付いているワークロード(Deployment、DeploymentConfig など)に Contrast の.NET エージェントを自動的に組み込みます。

関連項目

- エージェントオペレータの設定 (509ページ)
- エージェントオペレータのサポート対象テクノロジ (498ページ)

オペレータのアップグレード

Contrast エージェントオペレータは、セマンティックバージョニングに従います。

- バージョンには、オペレーター API への重大な変更が含まれる場合があります。メジャーバージョンのアップグレードでは、マニフェストが変更されていたり、既存の CRD の更新が必要な場合があるため注意が必要です。
- バージョンには新機能が含まれていますが、完全な後方互換性があり既存のクラスタに適用しても安全です。新しい機能を使用するために、オプションのマニフェストの変更が必要な場合があります。
- にはセキュリティおよびバグフィックスが含まれており、完全な後方互換性があるため既存のクラスタに安全に適用することができます。マニフェストの変更は必要ありません。

Contrast では、以下の形式でイメージタグを公開しています。

```
:2
:2.1
:2.1.10
:latest
```

ここで、`:2` は 2.X.X セマンティックバージョンブランチの最新リリースを表します。アップグレードを簡素化するために、リスク許容度に基づいて接頭辞のバージョンを使用することもできます (`imagePullPolicy` が `Always` に設定されていることを確認してください)。

注記

Contrast エージェントオペレータは、複数のレプリカとリーダーリースを使用する可用性の高い構成をサポートしていますが、すべてのオペレータインスタンスが同じバージョンを長期間実行しているデプロイメントのみをサポートします。`imagePullPolicy` オプションを使用して、複数のインスタンスを同じバージョンに維持しないでください。自動アップグレードが必要な場合は、Keel などのオペレータを使用して安全にアップグレードすることをお勧めします。

マイナーおよびパッチアップグレード

新しいバージョンへのアップグレードは、新しいクラスタにインストールするのと同じ手順で行います。クラスタ管理者として `kubectl`(Kubernetes)または oc(OpenShift)を使用して以下を実行し、オペレータマニフェストを適用します。

```
kubectl apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

```
oc apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

メジャーアップグレード

メジャーアップグレードには、マニフェストの追加変更が含まれる場合があります。`contrast-agent-operator` ネームスペースのみを削除すれば、インストールされた CRD(さらにクラスタ構成も)は維持されます。

```
kubectl delete namespace contrast-agent-operator
kubectl apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

```
oc delete project contrast-agent-operator
oc apply -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

ここでの一般的な手順はほとんどのメジャーアップグレードで有効ですが、メジャーアップグレードを確実に行うためにはリリースノートに記載される移行手順がある場合には、その手順に従ってください。

関連項目

- エージェントオペレータの設定 (509ページ)
- オペレータのサポート対象テクノロジ (498ページ)

エージェントオペレータのアンインストール

Contrast エージェントオペレータは、すべてのデータを Kubernetes のバックプレーンに保存し、クラスタから削除するとすべての変更が完全に削除されるように設計されています。すべてを正しくクリアするために、以下のステップを順番に実行することをお勧めします。

```
kubectl delete crd agentconfigurations.agents.contrastsecurity.com
kubectl delete crd agentconnections.agents.contrastsecurity.com
kubectl delete crd agentinjectors.agents.contrastsecurity.com
kubectl delete crd clusteragentconfigurations.agents.contrastsecurity.com
kubectl delete crd clusteragentconnections.agents.contrastsecurity.com
```

```
oc delete crd agentconfigurations.agents.contrastsecurity.com
oc delete crd agentconnections.agents.contrastsecurity.com
oc delete crd agentinjectors.agents.contrastsecurity.com
oc delete crd clusteragentconfigurations.agents.contrastsecurity.com
oc delete crd clusteragentconnections.agents.contrastsecurity.com
```

CRD を削除すると、オペレータの設定エンティティが自動的に削除されます。設定エンティティが削除された後、Contrast エージェントオペレータがクラスタワークロードに加えた変更が元に戻ります。

注記

これにより、オペレータが組み込んだワークロードの数によっては、Kubernetes が影響を受けたワークロードを再度デプロイするため、デプロイされたポッドが大幅に入れ替わる可能性があります。 大規模なクラスタでは注意が必要です。

クラスタが静止した後、オペレータは安全に削除できます。

```
kubectl delete -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

```
oc delete -f https://github.com/Contrast-Security-OSS/agent-operator/
releases/latest/download/install-prod.yaml
```

注記
推奨通りに最初の手順で CRD を削除した場合は、CRD の欠落に関するエラーは正常です。

エージェントオペレータの設定

ここでは、Contrast エージェントオペレータがアクセスする全ての設定エンティティのスキーマについて説明します。一部のエンティティは、オプション(任意)となります。

AgentConfiguration

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: AgentConfiguration
metadata:
  name: example-agent-configuration
  namespace: default
spec:
  yaml: |
    server:
      environment: QA
  suppressDefaultServerName: false
  suppressDefaultApplicationName: false
```

プロパティ	型	必須	デフォルト値	説明
spec.yaml	string	任意		ドキュメントの「YAML 設定」で説明している YAML 設定ファイル
spec.suppressDefaultServerName	boolean	任意	False	False の場合、デフォルト(通常は Pod 名)を使用せずに、挿入されるワークロード('kubernetes-{namespace}')に Contrast サーバ名が自動的に設定されます。
spec.suppressDefaultApplicationName	boolean	任意	False	False の場合、デフォルト(エージェントが生成)を使用せずに、挿入されるワークロード(ワークロード名)に Contrast アプリケーション名が自動的に設定されます。

注記
接続キーは、YAML ファイルで指定しても無視されるので、YAML ファイルで指定しないでください。

AgentConnection

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: AgentConnection
metadata:
  name: example-agent-connection
```

```
  namespace: default
spec:
  url: https://app.contrastsecurity.com/Contrast
  apiKey:
    secretName: example-agent-connection-secret
    secretKey: apiKey
  serviceKey:
    secretName: example-agent-connection-secret
    secretKey: serviceKey
  userName:
    secretName: example-agent-connection-secret
    secretKey: userName
```

プロパティ	型	必須	デフォルト値	説明
spec.url	string	○		Contrast サーバの URL
spec.apiKey.secretName	string	○		apiKey(API キー)を指定している Secret の名前
spec.apiKey.secretKey	string	○		apiKey(API キー)を指定している Secret での値に対するキー
spec.serviceKey.secretName	string	○		serviceKey(エージェントサービスキー)を指定している Secret の名前
spec.serviceKey.secretKey	string	○		serviceKey(エージェントサービスキー)を指定している Secret での値に対するキー
spec.userName.secretName	string	○		userName(エージェントユーザ名)を指定している Secret の名前
spec.userName.secretKey	string	○		userName(エージェントユーザ名)を指定している Secret での値に対するキー

重要
セキュリティ上、これらの Secret は AgentConnection と同じネームスペースに含まれる必要があります。

AgentInjector

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: AgentInjector
metadata:
  name: example-injector-dotnet-core
  namespace: default
spec:
  enabled: true
  version: latest
  type: dotnet-core
  image:
    registry: docker.io/contrast
    name: agent-dotnet-core
    pullSecretName: contrastdotnet-pull-secret
    pullPolicy: Always
  selector:
    images:
```

```
    - "*"
  labels:
    - name: app
      value: example-*
connection:
  name: example-agent-connection
configuration:
  name: example-agent-configuration
```

プロパティ	型	必須	デフォルト値	説明
spec.enabled	boolean	任意	TRUE	このエージェントの組み込みを有効または無効にします。
spec.version	string	任意	latest	組み込む Contrast エージェントのバージョン。リテラルの'latest'は、最新バージョンを組み込みます。バージョンの部分一致に対応しています。例えば、'2' を指定すると、バージョン '2.1.0' が選択されます。
spec.type	agentType	○		組み込むエージェントのタイプ。'dotnet-core'、'java'、'nodejs'か'nodejs-protect'、'php'のいずれかを指定できます。
spec.image.registry	string	任意	docker.io/contrast	エージェントイメージをダウンロードするために使用するイメージレジストリ。このレジストリには、組み込まれる Pod とオペレータがアクセスできる必要があります。
spec.image.name	string	任意	{タイプによる}	組み込むイメージに使用する名前
spec.image.pullSecretName	string	任意		Pod の imagePullSecrets リストに追加するプルシークレットの名前
spec.image.pullPolicy	string	任意	Always	Contrast のイメージを取得する際に使用するプルポリシー。詳細は、Kubernetes の imagePullPolicy を参照してください。
spec.selector.images	string[]	任意	Pod にあるすべてのコンテナを選択	エージェントを組み込むコンテナイメージ。Glob パターン(ワイルドカード)に対応しています。
spec.selector.labels	labelSelector[]	任意	ネームスペースにあるすべてのワークロードを選択	Deployment/StatefulSet/DaemonSet/DeploymentConfig のラベルで、その Pod がエージェントを組み込みむ対象となります。
spec.connection.name	string	任意	ClusterAgentConnection で指定された AgentConnection がデフォルト	AgentConnection のリソース名。同じネームスペース内に存在する必要があります。
spec.configuration.name	string	任意	ClusterAgentConfiguration で指定された AgentConfiguration がデフォルト	AgentConfiguration のリソース名。同じネームスペース内に存在する必要があります。

- 既存の AgentInjector を無効にすると、選択したワークロードから全てのエージェントの組み込みが削除されます。
- 参照する AgentConnection と AgentConfiguration は、AgentInjector と同じネームスペースに存在する必要があります。
- カスタムレジストリを使用する場合、挿入される Pod とオペレータの両方が、デフォルトのプルシークレットまたはカスタムのプルシークレットを通じて、アクセスできる必要があります。

- 本番前の環境でエージェントを使用する場合は、エージェントバージョンの `latest` を推奨します。
- AgentInjector は、Deployment、StatefulSet、DaemonSet、および DeploymentConfig(OpenShift 上)ワークロードを選択することをサポートします。Pod を直接挿入することはサポートされていません。
- 選択したワークロードが 1 つの Pod で多くのコンテナを作成する場合、spec.selector.images を使用して、組み込まれるコンテナをフィルタリングすることができます。

labelSelector

プロパティ	型	必須	デフォルト値	説明
name	string	○		一致させるラベルの名前
value	string	○		一致させるラベルの値。Glob パターン(ワイルドカード)に対応しています。

注記
ラベルの選択(複数)は、論理 AND 演算を使用して累積します。

agentType

エージェント	エージェントタイプ
.NET Core	dotnet-core
Java	java
Node.js	node.js または nodejs-protect
PHP	php

タイプの情報については、オペレータのサポート対象テクノロジ (498ページ)も参照してください。

ClusterAgentConfiguration

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: ClusterAgentConfiguration
metadata:
  name: default-agent-configuration
  namespace: contrast-agent-operator
spec:
  namespaces:
    - default
  template:
    spec:
      yaml: |
        server:
          environment: QA
```

プロパティ	型	必須	デフォルト値	説明
spec.namespace	string[]	任意	全てのネームスペース	この AgentConfiguration テンプレートを適用するネームスペース。Glob パターン(ワイルドカード)に対応しています。

プロパティ	型	必須	デフォルト値	説明
spec.template	AgentConfiguration	○		'spec.namespaces'で選択したネームスペースに適用するデフォルトの AgentConfiguration

注記

セキュリティ上、ClusterAgentConfiguration マニフェストは、オペレータと同じネームスペースにデプロイする必要があります。

ClusterAgentConnection

```
apiVersion: agents.contrastsecurity.com/v1beta1
kind: ClusterAgentConnection
metadata:
  name: default-agent-connection
  namespace: contrast-agent-operator
spec:
  namespaces:
    - default
  template:
    spec:
      url: http://app.contrastsecurity.com/Contrast
      apiKey:
        secretName: default-agent-connection-secret
        secretKey: apiKey
      serviceKey:
        secretName: default-agent-connection-secret
        secretKey: serviceKey
      userName:
        secretName: default-agent-connection-secret
        secretKey: userName
```

プロパティ	型	必須	デフォルト値	説明
spec.namespace	string[]	任意	全てのネームスペース	この AgentConfiguration テンプレートを適用するネームスペース。Glob パターン(ワイルドカード)に対応しています。
spec.template	AgentConnection	○		'spec.namespaces'で選択したネームスペースに適用するデフォルトの AgentConnection

注記

- セキュリティ上、ClusterAgentConnection マニフェストは、オペレータと同じネームスペースにデプロイする必要があります。
- ClusterAgentConnection で参照する Secret は、ClusterAgentConnection エンティティをデプロイするのと同じネームスペースに存在する必要があります。

関連項目

- エージェントオペレータのインストール (500ページ)
- エージェントオペレータのサポート対象テクノロジ (498ページ)
- エージェントオペレータのテレメトリ (514ページ)

.NET Core エージェントによるプロファイラチェーンのサポート

.NET Core エージェントは Contrast エージェントオペレータ (498ページ)と組み合わせて、Dynatrace オペレータを使用する Dynatrace でのプロファイラチェーンをサポートします。Dynatrace オペレータを使用して Dynatrace エージェントをワークロードに組み込むと、自動的にサポートが有効になります。

Dynatrace オペレータを `classicFullStack` モードで使用する場合は、Contrast エージェントオペレータの環境変数の `CONTRAST_ENABLE_EARLY_CHAINING=true` を設定し、対象のポッドを再起動します。こちらのマニフェストのサンプルも参考にしてください。

エージェントオペレータは、組込みの受付コントローラー/ミューテーションコントローラーを使用して、セキュリティコンテキスト制約(SCC)を利用している OpenShift クラスタでの実行をサポートします。Dynatrace オペレータと全てのインジェクションは、`restricted` ポリシーに対してテストされます。デフォルトの restricted ポリシーによって SCC ポリシーの設定が許可されていない OpenShift クラスタでアプリケーションを実行している場合は、`CONTRAST_SUPPRESS_SECCOMP_PROFILE=true` 環境変数がエージェントオペレータのワークロードに適用されていることを確認して下さい。

ベンダー	バージョン	サポート対象の検証日
Dynatrace	Operator バージョン 0.6.0	2022/06/09

今後の Dynatrace のバージョンによっては、チェーンが壊れる可能性があります。チェーンによって非互換性が生じる可能性があり、`agent.dotnet.enable_chaining: false` オプションを使用して無効にすることができます。

エージェントオペレータのテレメトリ

Contrast エージェントオペレータは、テレメトリを使用して、使用状況に関するデータを収集します。テレメトリは、オペレータが最初にクラスタにインストールされた時に収集され、その後も定期的に(数時間ごと)に収集されます。

弊社では、お客様のプライバシーは非常に大切であると考えています。このテレメトリ機能は、アプリケーションのデータを収集するものではありません。データは匿名化されてから、Contrast に安全に送信されます。そして、集約されたデータは暗号化されて保存され、アクセスが制限されます。収集されたデータは、全て 1 年後に削除されます。

テレメトリ機能を停止するには、`CONTRAST_AGENT_TELEMETRY_OPTOUT` という環境変数に `1` または `true` を設定してください。

テレメトリのデータは、*telemetry.dotnet.contrastsecurity.com* に安全に送信されます。ネットワークレベルで通信をブロックすることで、テレメトリ機能を停止することもできます。

テレメトリ機能で収集されるのは、以下のデータです。

オペレータ バージョン 0.3.0

- オペレータのバージョン
- オペレータの稼働時間
- Kubernetes API によって公開されたクラスタのバージョン情報とプラットフォーム
- クラスタ ID の暗号化(SHA256)匿名ハッシュで、オペレータが最初に起動した時にランダムに生成されてオペレータのネームスペースに Secret として格納される GUID

- クラスタ内の監視対象リソースの数(全てのオペレータエンティティ、DaemonSets、DeploymentConfigs、Deployments、Namespaces、Pods、Secrets、StatefulSets など)
- オペレータが内部でスローした例外(ログメッセージ、例外タイプ、例外メッセージ、スタックトレースフレームなど)

エージェントのパフォーマンス

アプリケーションセキュリティにおけるエージェントのパフォーマンスとは、アプリケーションを脆弱性と脅威から保護するために導入されたセキュリティエージェントまたはソフトウェアツールの効果と効率を指します。これらのエージェントは、さまざまなセキュリティの仕組みであり、ファイアウォール、侵入検知システム(IDS)、不正侵入防止システム(IPS)、Web アプリケーションファイアウォール(WAF)、脆弱性スキャナなどが考えられます。

関連項目

Protect 使用時のエージェントのパフォーマンス (515ページ)

Java エージェントのパフォーマンス (516ページ)

.NET Core エージェントのパフォーマンス (517ページ)

Protect 使用時のエージェントのパフォーマンス

Protect 使用時のパフォーマンスに影響を与えるものは何でしょうか？

エージェントは、潜在的なセキュリティリスクを正確かつ迅速に検出して、リアルタイムで対応する必要があるため、パフォーマンスは非常に重要です。ここでは、アプリケーションセキュリティにおけるエージェントのパフォーマンスに関連する主な側面をいくつか紹介します。

1. **精度：**セキュリティエージェントには、セキュリティの脅威を特定し分類する上で、高いレベルの精度が必要です。真のリスクに対処するために、過検知(正当なアクティビティを脅威として誤認すること)と検知漏れ(実際の脅威を検出できないこと)を最小限に抑えることが必要です。
   - 機密性の高いルールが検出された場合、ルールの監査と検証を行います。取り急ぎの対応として、ルールを調整するかオフにできます。エンジニア部門が修正や過検知(FP)の調整を担当して、システムの精度を確保することができます。
2. **処理速度と応答性：**アプリケーションセキュリティのエージェントは、セキュリティに関する問題を検出して迅速に対応するために、リアルタイムまたはほぼリアルタイムで動作する必要があります。対応が遅れると、脆弱性にさらされる時間が長くなり、攻撃が成功するリスクが高くなります。
   - Contrast のインフラサービスには、パフォーマンスに影響を与えることなく高負荷やエラーを処理するのに十分なリソースと容量を持つことが求められます。
3. **拡張性：**エージェントは、大規模なアプリケーションやネットワークのリクエストを処理できる必要があります。システムにはクラウドネイティブなライセンスモデルがあり、クラスタ化された環境でマイクロサービスを簡単に導入・拡張できます。最適なパフォーマンスを確保するために、必要に応じて柔軟にスケールアップやスケールダウンを行うことができます。
   - 更新された設定をデプロイすることによって、マイクロサービスを使用したシステムの効率的なスケールアップとスケールダウンが可能になります。スケーラブルなシステムのデプロイと管理のプロセスが簡素化されて、スムーズな運用が実現します。
4. **リソース使用率：**効果的なセキュリティエージェントは、CPU、メモリ、ネットワーク帯域幅のようなシステムリソースを効率的に使用する必要があります。保護されるアプリケーションのパフォーマンスに悪影響を与えないように、堅牢なセキュリティ対策の必要性に対処し、リソースの消費を最小限に抑える必要があります。
   - 通常、メモリ消費量には上限があります。メモリ使用量の詳細については、エージェント固有のセクションを参照してください。
5. **適応性：**アプリケーションエージェントは、進化する脅威や新しい攻撃ベクトルに適応できなければなりません。セキュリティ対策の継続的な有効性を確保するためには、シグネチャの更新、ルールの更新、機械学習モデルなど、エージェントの機能の定期的な更新と拡張が必要です。

- 最も適切なルールを決定して、そのルールを調整することで、パフォーマンスを最適化できます。また、例外機能や許可リストを使用して不要なスキャンや解析を減らすことで、さらなる調整ができ、パフォーマンスへの影響の軽減を図ることができます。

総じて、アプリケーションセキュリティにおけるエージェントのパフォーマンス目標は、潜在的な脅威に対して堅牢で信頼性の高い防御を提供しながら、過検知、応答時間、およびリソース使用率を最小限にすることです。組織は、セキュリティの有効性とシステムパフォーマンスのバランスをとることによって、アプリケーションのセキュリティ体制を強化し、さまざまな攻撃からアプリケーションを保護することができるのです。

関連項目

Java エージェントのパフォーマンス (516ページ)

.NET Core エージェントのパフォーマンス (517ページ)

Protect 使用時の Java エージェントで想定されるパフォーマンス

パフォーマンスへの影響を理解する手掛かりとなるよう、サンプルの Java アプリケーションを使用して内部テストを実施し、情報を収集しました。実際のパフォーマンス数値は、他のいくつかの要因によって異なる場合があることに注意してください。

CPU 使用率

- Java エージェントは通常、CPU リソースの 5〜15%を使用します。
- この範囲は、エージェントで Protect モードが有効になっている場合に発生する追加の CPU 負荷の概算です。
- アプリケーションの複雑さ、発生する負荷、およびその他の環境要因によって、影響は異なる場合があります。

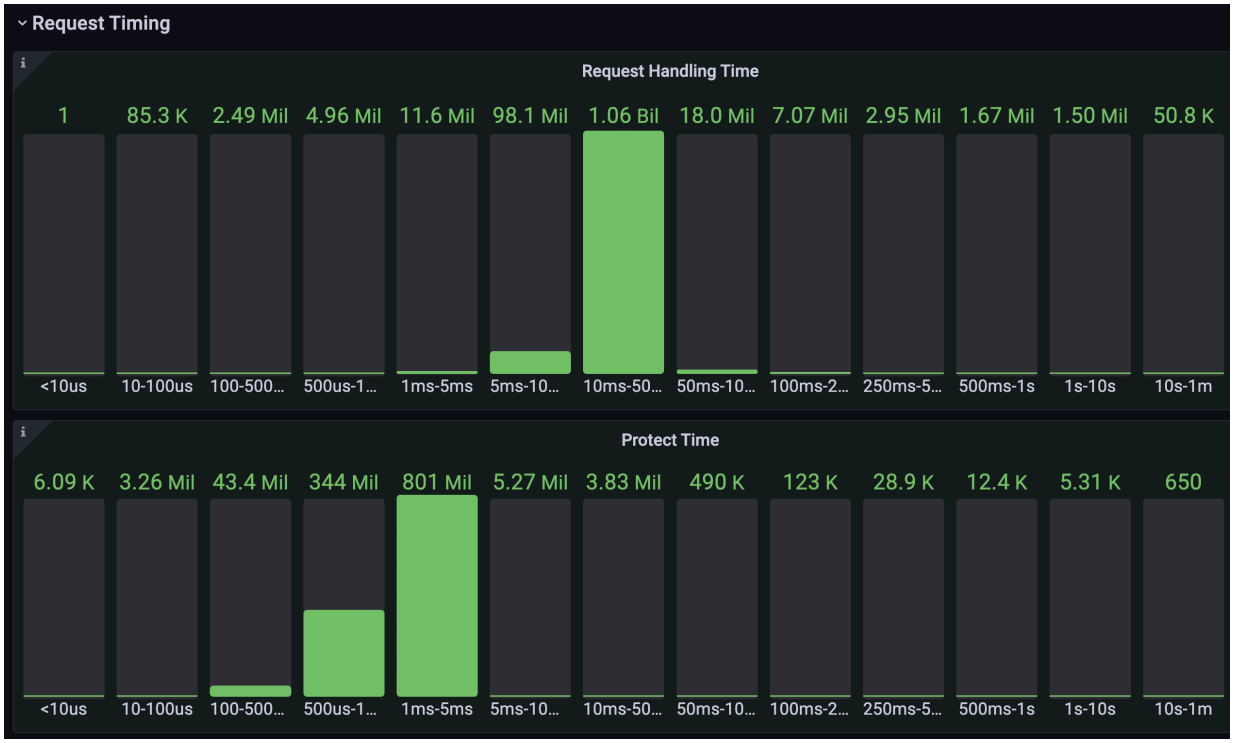
メモリ使用量

- Java エージェントには通常、200〜300MB の追加メモリが必要です。
- このメモリ使用量は、エージェントの操作と関連するデータ構造を考慮したものです。
- 実際のメモリへの影響は、アプリケーションのサイズと複雑さ、処理される同時リクエスト数などによって異なる可能性があります。

レイテンシ

- Java エージェントで Protect モードを有効にすると、リクエストの処理中に追加のレイテンシ(遅延時間)が発生する可能性があります。
- Protect の解析で観測されたリクエストの 99%は、100µs(マイクロ秒) 〜 5ms(ミリ秒)です。
- 具体的な影響は、アプリケーションの負荷、同時実行ユーザ数、リクエストの内容などのさまざまな要因によって異なります。
- 通常発生する遅延は最小限ですが、特定のアプリケーションの性質により異なる可能性があるという点に注意してください。これらのパフォーマンス数値は、Java エージェントがアプリケーションに与える影響を最初に理解する際に参考にするための基準となります。ただし、実際のパフォーマンスへの影響は、お使いの環境やアプリケーションの固有の要素によって異なる可能性があることを強調しておきます。典型的なワークロードや現実的なシナリオを使用して、パフォーマンステストを実施し、アプリケーションのパフォーマンスへの影響を適切に測定することを推奨します。本番環境に近い条件をシミュレートすることで、設定を調整していき、セキュリティとパフォーマンスのトレードオフを評価することができます。弊社では、Java エージェントの性能の向上と、アプリケーションのパフォーマンスに与える影響の最適化に継続して取り組んでいます。皆様からのフィードバックを大切にしておりますので、パフォーマンスに関する意見や懸念事項などがあれば、ぜひご連絡ください。Java エージェントのパフォーマンスに関するサポートや追加情報が必要な場合は、お気軽に弊社サポートまでお問い合わせください。最適なパフォーマンスレベルを維持しながら、アプリケーションのセキュリティと安定性を確保できるようサポートします。

次のグラフは、Contrast エージェントを使用している実際の顧客のホストアプリケーションのリクエスト処理時間をまとめたものです。Java のパフォーマンスへの影響を示す 24 時間のサンプルを示しています。



- 上の行は全体のリクエスト処理時間を示しており、HTTP リクエストのかなりの部分、つまりリクエストの約 93%(10 億 6000 万ヒット)が 10 ミリ秒から 50 ミリ秒の範囲内にあることを示しています。さらに、5 ミリ秒から 10 ミリ秒の処理時間で、9,810 万ヒットが発生し、残りの処理時間は、正規分布に従っていることが分かります。
- 下の行には、Protect 時間を表示しています。これは、Protect RASP 製品が各リクエストを解析するのに要した時間を示しています。特に、全リクエストのうち、8 億件が 1 ミリ秒から 5 ミリ秒の時間枠内で解析されており、レイテンシへの影響がごくわずかであることを示しています。同様に、さらに 3 億 4,400 万リクエストが、0.5〜1 ミリ秒というさらに短い時間枠内で解析されており、レイテンシへの影響が最小限であることを示しています。エージェントを組み込んだホストアプリケーションに対するリクエスト処理時間の分布により、異なる時間範囲に分類されたリクエスト数を確認できます。同時に Protect によってもたらされるパフォーマンスへの影響についても把握できます。
- ガベージコレクションやその他の JVM 要素は、実行を一時停止する可能性がありタイミングに影響することがあります。

Protect 使用時の.NET Core エージェントで想定されるパフォーマンス

パフォーマンスへの影響を理解する手掛かりとなるよう、サンプルの.NET Core アプリケーションを使用して内部テストを実施し、情報を収集しました。実際のパフォーマンス数値は、他のいくつかの要因によって異なる場合があることに注意してください。

CPU 使用率

- .NET Core エージェントは通常、CPU リソースの 5〜10%を使用します。
- この範囲は、エージェントで Protect モードが有効になっている場合に発生する追加の CPU 負荷を表しています。
- 実際の影響は、アプリケーションの複雑さ、処理するワークロード、およびその他の環境上の要因によって異なる場合があります。

メモリ使用量

- .NET Core エージェントには通常、約 200MB の追加メモリが必要です。
- このメモリ使用量は、エージェントの操作と関連するデータ構造を考慮したものです。
- 実際のメモリへの影響は、アプリケーションのサイズ、複雑さ、および処理される同時リクエスト数によって異なる場合があります。

レイテンシ

- .NET Core エージェントで Protect モードを有効にすると、リクエストの処理中に追加のレイテンシ(遅延時間)が発生する可能性があります。
- このレイテンシの増加は、1〜10 ミリ秒の範囲になります。
- レイテンシへの具体的な影響は、アプリケーションの負荷、同時実行ユーザ数、処理されるリクエストの内容などの要因によって異なります。
- 通常発生する遅延は最小限ですが、実際の数値は特定のアプリケーションの性質により異なる可能性があるという点に注意してください。これらのパフォーマンス数値は、.NET Core エージェントがアプリケーションに与える影響を最初に理解する際に参考にするための基準となります。ただし、実際のパフォーマンスへの影響は、お使いの環境やアプリケーションの固有の要素によって異なる可能性を認識することが非常に重要です。典型的なワークロードや現実的なシナリオを使用して、パフォーマンステストを実施し、アプリケーションのパフォーマンスへの影響を適切に評価することを強く推奨します。本番環境に近い条件をシミュレートすることで、設定が最適化され、セキュリティとパフォーマンスのバランスを評価することができます。弊社では、.NET Core エージェントの性能の向上とアプリケーションのパフォーマンスへの影響の最適化に継続して取り組んでいます。皆様からのフィードバックを大切にしておりますので、パフォーマンスに関する意見や懸念事項などがあれば、ぜひご連絡ください。.NET Core エージェントのパフォーマンスに関するサポートや追加情報が必要な場合は、お気軽に弊社サポートまでお問い合わせください。最適なパフォーマンスレベルを維持しながら、アプリケーションのセキュリティと安定性を確保できるようサポートします。

ユーザガイド

[en]

Contrast を操作する方法は、お使いの環境や特定の要件によって異なります。使用するツールやインテグレーション、ユーザに割り当てられたロールや権限、また Contrast へのアクセス方法(Web インターフェイス、コマンドラインツール、または REST API 経由)などによって異なります。

注記
本ドキュメントで使用する全てのコマンドは、Contrast がインストールされたディレクトリから管理者権限でコマンドシェルにて実行してください。

[en]

Contrast ユーザのほとんどが、Edit ロールが権限として割り当てられたアカウントになります(自分のアカウントに割り当てられている権限レベル (523ページ)はユーザの設定 (521ページ)から確認できます)。

Edit 権限では、アプリケーションを検査 (46ページ)して、Contrast で検出結果を確認できます。また、Contrast Web インターフェイスで、以下の基本機能を操作できます。これらは、全て Contrast Web インターフェイスのナビゲーションバーからアクセスできます。

- アプリケーション (526ページ)
  組織のアプリケーションの一覧(検索可能)を表示します。アプリケーションのライセンス付与、マージ、タグ付け、アーカイブ、復元などを行うことができます。
- サーバ (588ページ)
  組織のサーバの一覧(検索可能)を表示します。サーバ環境の指定、Assess および Protect の有効化、設定、タグ付け、削除などを行うことができます。
- ライブラリ (599ページ)
  組織内の全てのアプリケーションで使用されているライブラリの一覧(検索可能)を表示します。ライブラリにタグを付けたり、ライブラリに存在する既知の脆弱性やリスクの高いライブラリに関する統計情報も表示することができます。
- 脆弱性 (692ページ)
  検出された脆弱性の一覧(検索可能)を表示します。この脆弱性の一覧は、組織内の各アプリケーションごとに表示することができます。脆弱性のステータス変更、マージ、共有、タグ付け、エクスポートなどを行うことができます。脆弱性の詳細までドリルダウンすると、その脆弱性を修正するための詳細情報や対策方法を参照できます。
- 攻撃 (707ページ)
  組織内の全てのアプリケーションで発生している攻撃や発生した攻撃の一覧(検索可能)を表示します。攻撃を最上位レベルで表示したり、ドリルダウンして個別の攻撃イベントの詳細情報を参照することができます。

また、Contrast には他にも便利な機能やツールがあります。

- レポート (718ページ)
  データを収集し、CSV または PDF として エクスポートして Contrast の外部で共有できます。
- インテグレーション (726ページ)
  バグ追跡システム、ビルドツール、アプリケーションサーバ、SIEM(Security Incident Event Management)、通知、チャットなどの外部のツールと Contrast を連携して使用できます

- Contrast CLI (685ページ)
  アプリケーションでソフトウェアコンポジション解析(SCA)を実行し、オープンソースライブラリとの依存関係を表示します。

これらの機能の設定に関しては、ほとんどの場合システム管理者 (874ページ)か組織の管理者 (813ページ)であるか、または RulesAdmin (780ページ)ロールが権限が必要になりますが、Edit 権限では以下のことを行えます。

- エージェントのインストールと設定 (46ページ)
- 通知の送信
- スコア設定のカスタマイズ (834ページ)

Contrast でサポートされている言語

以下の表は、Contrast の各コンポーネントでサポートされている言語を示しています。

表のキー：

●サポート対象
●検討中
-- このテクノロジでは該当なし

言語	Contrast Scan	Contrast SCA	Contrast Assess	Contrast Protect	Contrast Serverless
Java	●	●	●	●	● (AWS と Azure)
Kotlin	●	●	●	●	●
Scala	●	●	●	●	●
.NET Framework	●	●	●	●	--
.NET Core	●	●	●	●	●
Node.js	●	●	●	●	●
クライアントサイドの JavaScript	●	●	●	●	--
Ruby	●	●	●	●	●
Python	●	●	●	●	●
Go	●	●	●	●	●
PHP	●	●	●	●	●
C#	●	●	●	●	●
VB.NET	●	●	●	●	●

Contrast Scan による追加のサポート対象

Contrast Scan では、さらに以下の言語とテクノロジがサポートされます。

SAP ABAP
ActionScript
ASP.NET
C および C++
COBOL
HTML
Informix (SQL および 4GL)
JavaScript/TypeScript
JCL
JSP
NATURAL
Objective-C
Oracle Forms
PS-SQL
PowerScript
RPG4
Swift
Transact-SQL
TypeScript
Visual Basic
XML

ユーザの設定の管理

Contrast でユーザの設定を管理するには、**ユーザメニュー** (Contrast Web インターフェイスの右上にある自分の名前)を開いて、**ユーザの設定**を選択します。 ここでは、自分のアカウントに関して以下を管理できます。

- パスワードの変更 (521ページ)
- 2 段階認証の設定 (521ページ)
- プロファイルの編集 (522ページ)
- API キーの確認 (522ページ)
- 通知の管理 (523ページ)
- 権限の表示 (523ページ)

Contrast へのログイン

初めて Contrast にログインする場合は、管理者が作成した招待メールを承認する必要があります。E メールに記載されているリンクを選択して、Contrast にログインします。

組織でシングルサインオン(SSO) (827ページ)を使用している場合は、ログインページのチェックボックスを選択して、パスワード入力フィールドを無効にします。E メールアドレスの入力のみが必要になります。E メールが認証されたら、SSO の認証情報を使用してログインできます。

2 段階認証 (521ページ)を使用している場合は、SSO 認証が成功した*後*にログインプロセスが行われます。

パスワードの変更

アカウントのパスワードを変更するには、以下の手順を実行します。

1. Contrast Web インターフェイスにログインします。
2. ユーザメニューで、**ユーザの設定 > パスワードの変更**を選択します。
3. 画面の各フィールドで、現在のパスワードと新しいパスワードを入力します。最後のフィールドには、確認のために新しいパスワードを再度入力します。

   > **注記**
   > 新しいパスワードは管理者が設定した (826ページ)パスワードポリシーに従う必要があります。パスワードの入力を開始すると、パスワード要件が表示されて判定されます。

4. チェックボックスをオンにして利用規約に同意します。
5. 変更を**保存**します。

> **注記**
> シングルサインオン(SSO)を使用してログイン (521ページ)する場合は、パスワードを変更するオプションはありません。

2 段階認証の設定

組織 (827ページ)またはシステム (921ページ)レベルで管理者が 2 段階認証を有効にしている場合、ユーザ名とパスワード以外に安全にログインする仕組みを指定できます。2 段階認証を設定するには：

1. ユーザメニューで、**ユーザの設定 > 2 段階認証**を選択します。
2. トグルを使用して、2 段階認証を有効にします。

3. ラジオボタンを使用して、認証コードを受け取る方法を選択します。
   - **E メール**：Contrast と関連付けているメールアドレスで、認証用のコードを受け取ることができます。この設定を選択すると、認証コードが記載された E メールが送信されますので、設定画面で認証コードを入力します。
   - **Google Authenticator**：モバイルデバイスにアプリのダウンロードが必要な場合がありますが、モバイルデバイスのアプリで、認証コードを受け取ることができます。この設定を選択すると QR コードが表示されますので、QR コードを読み取り、手順に従ってデバイスを検証します。
4. 2 段階認証の設定を完了する前に、.txt ファイル形式でバックアップコードをダウンロードしておくことができます。このコードにより、ログインエラーが発生した場合や、アカウントがロックアウトされた場合でもログインが許可されます。バックアップコードをダウンロードしたら、安全な場所に保存してください。
5. 認証コードの受け取り方法を変更する場合は、2 段階認証タブの通知設定で再度設定できます。通知設定の選択を変更すると、新しいバックアップコードが自動的に発行されます。通知設定の変更は、保存する必要はありません。

ヒント
いずれの方法でも問題が発生した場合は、提供されたバックアップコードを使用することができます。**サインインできませんか**を選択するか、Google Authenticator の**デバイスをリセット**のリンクを使用してください。

プロファイルの管理

プロファイルの画面では、名前の編集、時刻· 日付形式、言語の設定などを更新して、Contrast の使用環境をカスタマイズできます。

1. ユーザメニューで、**ユーザの設定 > プロファイル**を選択します。
2. **一般情報**で、名前やタイムゾーンなどのアカウントの基本情報を設定します。
3. 新しいプロファイル画像をアップロードするには、サムネイルをクリックします。
4. **あなたのキー**のセクションには、組織のキーと個人のキーが表示 (522ページ)されます。
5. 変更を**保存**します。

API キーの確認

エージェントまたはカスタムスクリプトで Contrast との通信を確立するためには、API キーを使用します。以下の目的により、エージェントおよび Contrast API でキーが必要になります。

- アクセスする組織を識別するため
- 有効なユーザとして識別するため

手順

1. **ユーザメニュー > ユーザの設定 > プロファイル**に移動します。
   あなたのキーのセクションで、API キーと次のキーを含む **API キー情報**を確認できます。
   - **組織 ID**：アクセスする組織を識別するものです。
   - **サービスキー**： 認証情報を使用して Contrast サービスに接続するためのものです。
   - **認証ヘッダー**： 有効なユーザとして識別するためのものです。

2. 認証ヘッダーをコピーするには、**コピー**を選択します。
3. サービスキーをローテーションするには、**ローテーション**をクリックします。
   サービスキーをローテーションすると、このキーを使用しているインテグレーションに影響します。再接続するために、このキーを使用しているインテグレーションで認証情報を更新してください。
4. **サンプル API リクエストを生成**をクリックすると、サンプル API リクエストが生成されてクリップボードにコピーされます。

アカウントの通知の管理

通知の設定を変更するには：

1. **ユーザメニュー**で、**ユーザの設定 > 通知**を選択します。
2. **通知の登録**のフィールドをクリックして、通知を受け取りたいアプリケーションを選択します(複数選択可)。デフォルトでは「全てのアプリケーション」が選択されます。
3. **Contrast 内**と **E メール**の列にあるトグルボタンを使用して、以下のイベントに対する通知を有効または無効にします。
   - **積極的な攻撃**： Protect が有効なアプリケーションへの積極的な攻撃が発生。
   - **新しい脆弱性**: Contrast で新たに脆弱性を検出。特定の深刻度やライブラリの通知を受け取るよう指定するには、フィールドをクリックします。デフォルトでは「全て」が選択されます。
   - **脆弱なライブラリ**： 新たに脆弱性のあるライブラリが検出されたか、ライブラリに新たな CVE が検出された場合に通知を受け取ることができます。
   - **サーバのメッセージ**：サーバの信頼性に関するメッセージを取得。
   - **サーバのオフライン**： サーバへのアクセス不可。
   - **新しいコメント**： チームメンバーが検出結果に関してコメントを投稿。
   - **新しいアセット**： アクセス権 (523ページ)のある新しいアセット(アプリケーションやサーバ)が追加。フィールドをクリックすると、「アプリケーション」か「サーバ」のいずれかにこの通知を設定できます。デフォルトでは「全て」が選択されます。
   - **ライセンス切れ間近**：アプリケーションのライセンスの有効期限が間近。
   - **ポリシー違反**：コンプライアンスポリシー、ライブラリポリシー、修復ポリシーでの違反。チェックボックスをオンにすると、ポリシー違反の E メールをまとめてダイジェストとして通知できます。
   - **E メールダイジェスト**： Contrast での 1 日のアクティビティを要約して毎日配信。

権限の表示

権限のページでは、組織およびアクセス可能なアプリケーションの両方について、割り当てられている権限の詳細が表示されます。権限を表示するには：

1. **ユーザメニュー > ユーザの設定 > 権限**にアクセスします。
2. ページの上部には、自分の組織が自分の組織ロールと一緒に表示されます。**アプリケーションの権限**の表には、組織内の各アプリケーションに対して割り当てられているロールが表示されます。
3. 各ロールの横にあるヘルプアイコンをクリックして、「詳細を表示」を選択すると、各レベルで使用可能なデータアクセスと操作について参照できます。

関連項目

- アプリケーションロール (953ページ)
- 組織ロール (955ページ)

プロジェクト

マニフェストで CLI を実行した後、または GitHub アカウントが Contrast の組織に接続した後に、関連するプロジェクトと検査結果が Contrast Web インターフェイスに表示されます。これにより、エージェント組み込みが不可能であったり、ソフトウェア開発ライフサイクル(SDLC)の中で対応するのでは遅すぎたりするオープンソフトウェアの脆弱性を、より早く広範囲を可視化することができます。

プロジェクトページで CLI の結果を表示するには、Contrast CLI 2.0 を npm でインストール (667ページ)する必要があります。

プロジェクトページの一覧にリポジトリのスキャン結果を表示するには、GitHub アカウントを Contrast に接続 (715ページ)します。

関連項目

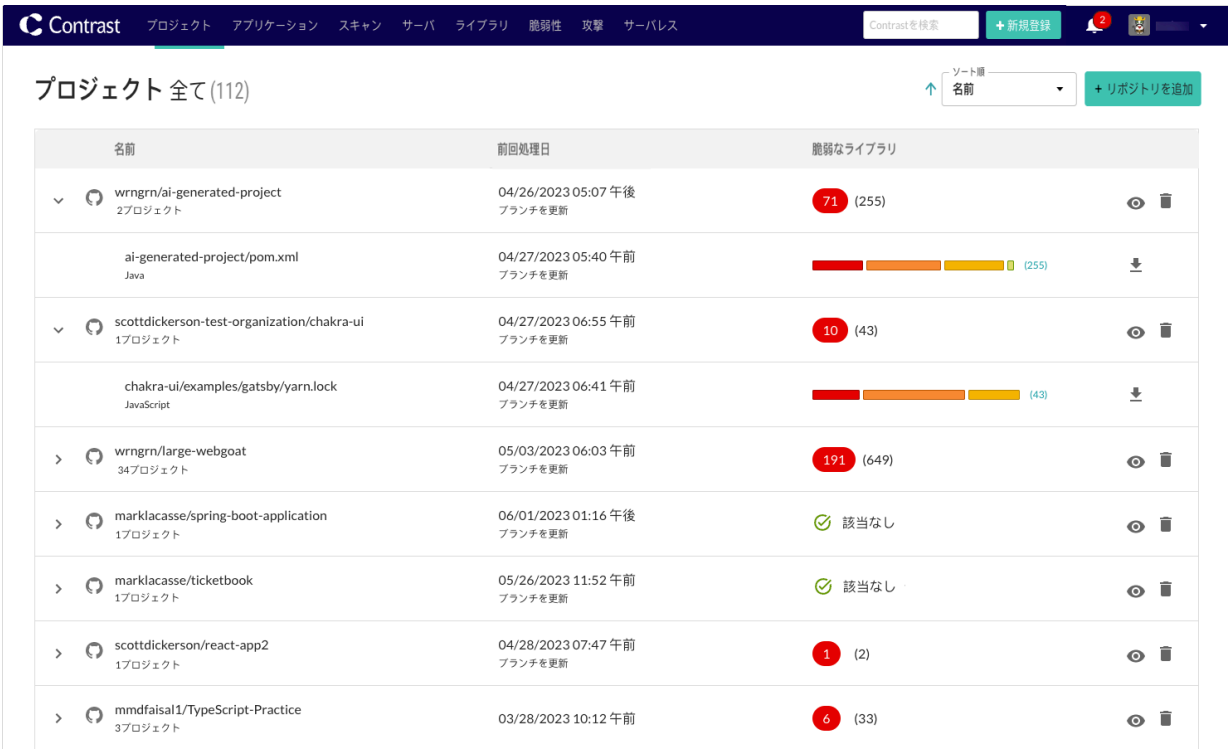
- プロジェクトの表示 (524ページ)
- Contrast CLI のサポート対象の言語とパッケージマネージャ (666ページ)
- Contrast Security GitHub アプリ (715ページ)

プロジェクトの表示

GitHub から Contrast に接続されているプロジェクトやリポジトリの情報を表示する方法は複数あります。

注記

組織内の権限によって、このページで操作を実行できる場合とできない場合があります。リポジトリを削除するためには、Admin 権限 (953ページ)を有効にする必要があります。



- Contrast Web インターフェイスのナビゲーションバーで**プロジェクト**を選択します。
- 一覧からプロジェクト名を選択し、展開できる場合は行を展開すると、接続されているプロジェクトの詳細情報が表示されます。各行には、最新のアクティビティと脆弱なライブラリの総数、および重大な脆弱性があるライブラリの総数が含まれています。

- プロジェクトページには以下が表示されます。
  - **名前**：これは、GitHub アカウントとリポジトリ名、または CLI 用にローカルに保存されたマニフェストのいずれかを含むプロジェクトの名前です。実行された解析によって、プロジェクトの種類が分類されます。Contrast CLI で解析されたプロジェクトは、このアイコン「」で識別され、Contrast Security の GitHub アプリで解析されたプロジェクトは、このアイコン「」で識別されます。
  - **前回処理日**：直近の活動(トリガーの理由)とそのタイムスタンプ。
  - **脆弱なライブラリ**：ここには、ライブラリの脆弱性(CVE)が確認されたプロジェクト内のライブラリ数が表示されます。ライブラリに少なくとも 1 つの重大な CVE がある場合、そのライブラリ数が赤で表示されます。選択可能な場合、行を展開すると、接続されているプロジェクトが表示されます。脆弱性の棒グラフにカーソルを合わせると、深刻度別の CVE 数が表示されます。棒グラフをクリックすると、詳細パネルが開きます。脆弱性が存在する場合、一覧で表示され、深刻度別に色分けされます。
    - 「*解析が完了していません。再試行するか、サポートにご連絡ください。*」のメッセージが表示される場合は、解析が進行中であるか、エラーが発生していることを意味します。再度、リポジトリに接続してみてください。失敗した場合は、サポートに連絡してください。
    - 「*該当なし*」のメッセージが表示されている場合は、脆弱なライブラリは検出されなかったことを意味します。
  - **処理**： ここでは、GitHub のリポジトリに移動したり、リポジトリを削除 (718ページ)したり、CLI プロジェクトのデータを CSV ファイルにエクスポート (525ページ)したりできます。
- 右上の *ソート順*ドロップダウンを使用すれば、前回処理日または名前で一覧を並べ替えできます。

プロジェクトの情報をエクスポート

プロジェクトの詳細情報は、CSV ファイルでエクスポート、または SBOM(ソフトウェア部品表)ファイルを生成してエクスポートできます。

CSV ファイルでエクスポートする場合、ファイルにはプロジェクトごとに、*Library Name*(ライブラリ名)、*Language*(言語)、*Version*(バージョン)、*Release Date*(リリース日)、*CVE Count*(CVE 数)、*Licenses*(ライセンス)などのデータフィールドが含まれます。

手順

プロジェクトの情報をエクスポートするには：

1. Contrast Web インターフェイスのナビゲーションバーで**プロジェクト**を選択します。
2. エクスポートするプロジェクトの行を探します。
3. 行の右端にある**エクスポート**アイコンを選択します。
4. *ダウンロード*より、出力形式を選択します。 SBOM オプションを選択した場合は、フォーマット標準を指定する必要があります。

ファイルがデスクトップにダウンロードされます。

アプリケーション

アプリケーションにエージェントを組み込んだら (46ページ)、Contrast Web インターフェイスでアプリケーションを確認 (526ページ)します。アプリケーションに関して、以下のような機能があります。

- ルートカバレッジ (538ページ)
- セッションメタデータ (535ページ)
- フローマップ (544ページ)
- アプリケーションのスコア (958ページ)

注記
これらの機能を使用するには、組織の管理者かアプリケーションの管理者(Admin ロールのあるアカウント)が、アプリケーションにライセンスを付与 (820ページ)する必要があります。

アプリケーションの表示

Contrast Web インターフェイスでアプリケーションの情報を表示するには、いくつかの方法があります。

手順

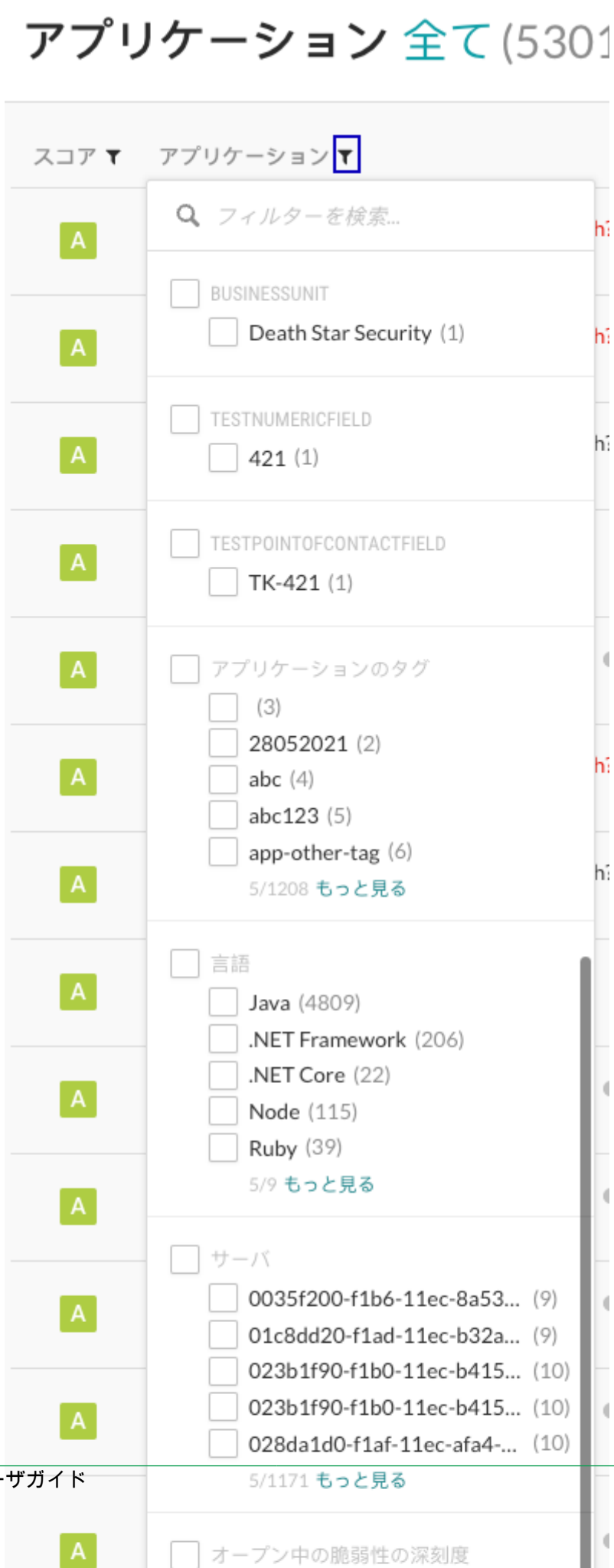
1. ナビゲーションバーで**アプリケーション**を選択すると、組織内で検出された全てのアプリケーションが一覧表示されます。
2. アプリケーションの一覧からアプリケーション名をクリックすると、そのアプリケーションの**概要**タブにアクセスできます。
3. アプリケーションのステータスに基づいて一覧にフィルタをかけるには、アプリケーションの一覧の最上部にある小さい三角形を選択します
   または、特定のアプリケーションを名前で検索するには、**虫眼鏡**アイコン( )を選択します。

フィルタには次のものがあります。

- **全て**：Contrast に追加したアプリケーション(アーカイブされたアプリケーションを除く)。
- **オンライン**： エージェントが過去 5 分以内に Contrast に接続したアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **オフライン**： エージェントが 5 分以上 Contrat に接続していないアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **マージ済**： マージされたアプリケーションの一部であるアプリケーションのみ(メインアプリケーションとそのコンポーネント)。
  アーカイブされたアプリケーションを除く。
- **ライセンスあり**： Contrast のライセンスが適用されているアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **ライセンスなし**： Contrast のライセンスが適用されていないアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **高リスク**： 深刻度が重大か高の脆弱性があり、ステータスが、、のアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **ポリシー違反**： このフィルタは、組織がコンプライアンスポリシーを有効にしている場合に利用できます。
  コンプライアンスポリシーに違反しているアプリケーションのみ。
  アーカイブされたアプリケーションを除く。
- **アーカイブ済**: 履歴として参照するために表示されるアプリケーションのみ。アーカイブされたアプリケーションのエージェントは、Contrast に脆弱性を報告しなくなります。

4. アプリケーションのスコアで表示を絞り込むには、「スコア」列のヘッダの横にある**フィルター**アイコン(▼)を選択し、1 つまたは複数のスコアを選択します。
   フィルターを解除するには、列のヘッダの横にある**クリア**を選択します。

5. アプリケーションの一覧で表示を絞り込むには、「アプリケーション」列のヘッダの横にある**フィルター**アイコン(▼)を選択します。フィルタには次のものがあります。
   - **アプリケーションメタデータ(設定した場合)：** アプリケーションに関連付けられたアプリケーションメタデータ。
   - **アプリケーションのタグ(作成した場合)：** アプリケーションに割り当てたタグ。
   - **言語：** アプリケーションで使用されている言語。
   - **サーバ：** アプリケーションが実行されているサーバ。
   - **オープン中の脆弱性の深刻度：** オープン中の脆弱性の深刻度。
   - **テクノロジ：** アプリケーションで使用されているテクノロジ。例えば、JSON や jQuery など。フィルターを解除するには、列のヘッダの横にある**クリア**を選択します。
   - **環境：** アプリケーションに関連付けられている環境(開発、QA、本番)。
   - **アプリケーションの重要度：** アプリケーションの設定で指定したアプリケーションの重要性。


アプリケーション 全て (5301
スコア
アプリケーション
フィルターを検索...
BUSINESSUNIT
Death Star Security (1)
TESTNUMERICFIELD
421 (1)
TESTPOINTOFCONTACTFIELD
TK-421 (1)
アプリケーションのタグ
(3)
28052021 (2)
abc (4)
abc123 (5)
app-other-tag (6)
5/1208 もっと見る
言語
Java (4809)
.NET Framework (206)
.NET Core (22)
Node (115)
Ruby (39)
5/9 もっと見る
サーバ
0035f200-f1b6-11ec-8a53... (9)
01c8dd20-f1ad-11ec-b32a... (9)
023b1f90-f1b0-11ec-b415... (10)
023b1f90-f1b0-11ec-b415... (10)
028da1d0-f1af-11ec-afa4-... (10)
5/1171 もっと見る
オープン中の脆弱性の深刻度

アプリケーションの詳細情報

アプリケーションの「概要」タブには、アプリケーションの設定とアクティビティに関する情報が表示されます。

ダッシュボード

タブ上部のダッシュボードには、以下の情報が表示されます。

- **スコア**：アプリケーションで検出された脆弱性の数と深刻度に基づいたレターグレードです。説明は、アプリケーションのスコアガイド (958ページ)を参照してください。
- **ライブラリ**:Contrast SCA で特定されたオープンソースライブラリの数と脆弱性のあるライブラリの数。
- **ルート疎通**：アプリケーションで疎通されたルートの数と、観測されたルートの数。
- **サーバ**：選択したアプリケーションに関連付けられているサーバの数。

環境の情報

アプリケーションにサーバが定義されている場合は、環境ごとに次の詳細が表示されます。

- **Protect と Assess の設定**：Protect または Assess がオンかオフになっているサーバの数がバーに表示されます。
  設定を管理するには、バーのセクションを選択します。選択すると、フィルターが適用されたサーバの表示 (589ページ)が開き、Assess と Protect の設定を管理できます。
- **サーバ**：アプリケーションに関連付けられているサーバの数。
- **脆弱性**：脆弱性の数。フィルタを使用すれば、深刻度、ステータス、種類別に表示を変更できます。
  脆弱性バーのセクションにカーソルを合わせると、詳細が表示されます。
- **脆弱性の傾向**： Contrast で確認された脆弱性の傾向と、これらの脆弱性が手動による検証または自動検証ポリシーによって対策された修復状況が表示されます。

アプリケーションの設定の編集

アプリケーションに関して、以下のような特定の情報を参照したり、編集することができます。

- **アプリケーション名**： 組織内の各アプリケーション には、一意の名前が必要です。アプリケーションの名前を変更するには、アプリケーションの一覧からアプリケーション名をクリックして、アプリケーションの**概要**ページに移動します。ページ上部の名前をクリックしてテキストを更新します。
  または、概要ページの右上にある**設定**アイコンを選択し、**アプリケーションの設定**画面で、アプリケーション名を更新します。
  システム管理者(SuperAdmin ロールのあるユーザ)の場合、ユーザメニューで **SuperAdmin** を選択して**アプリケーション**ページにアクセスし、一覧上で名前をクリックしてアプリケーション名を編集することもできます。
- **アプリケーション ID**： アプリケーション ID は、ブラウザの URL にある最後の URI セグメントです。アプリケーション ID を調べるには、アプリケーションの一覧からアプリケーションを選択します。URL の `applications/` の後のセグメントがアプリケーション ID です。
- **アプリケーションの重要性**： この値は、アプリケーションのメタデータとして表示され、組織のインテグレーションの設定にも使用できます。アプリケーションの重要度を設定するには、アプリケーション名を選択して、**概要**ページを表示し、**設定**アイコンを選択します。**アプリケーションの設定**画面で、**重要性**フィールドを使用してドロップダウンからレベルを選択します。

1. **アプリケーション**タブで、アプリケーションを選択します。選択したアプリケーションの概要が表示されます。
2. アプリケーションの概要で設定⚙アイコンを選択します。**アプリケーションの設定**画面が表示されます。
3. 必要に応じてフィールドを編集してください。
4. 更新したアプリケーションの設定を保存するには、**保存**をクリックします。

フィールドの説明

- **アプリケーション名**： 組織内の各アプリケーション には、一意の名前が必要です。アプリケーションの名前を変更するには、アプリケーションの一覧からアプリケーション名をクリックして、アプリケーションの**概要**ページに移動します。ページ上部の名前をクリックしてテキストを更新します。
  または、概要ページの右上にある**設定**アイコンを選択し、**アプリケーションの設定**画面で、アプリケーション名を更新します。
  システム管理者(SuperAdmin ロールのあるユーザ)の場合、ユーザメニューで **SuperAdmin** を選択して**アプリケーション**ページにアクセスし、一覧上で名前をクリックしてアプリケーション名を編集することもできます。
- **アプリケーションコード**：組織内部で識別するコードや番号で、アプリケーションやマイクロサービスに対して一意のものです。 これらのアプリケーションの一意の識別子を連携して Contrast で利用する場合は、このフィールドを使用します。このコードは、アプリケーション起動時に設定でき、エージェントの設定のアプリケーションプロパティのセクションで指定できます。
- **優先する URL**：脆弱性を再現するために使用される URL です。
  [en]
- **重要性**： この値は、アプリケーションのメタデータとして表示され、組織のインテグレーションの設定にも使用できます。アプリケーションの重要度を設定するには、アプリケーション名を選択して、**概要**ページを表示し、**設定**アイコンを選択します。**アプリケーションの設定**画面で、**重要性**フィールドを使用して、ドロップダウンメニューからレベルを選択します。

アプリケーションにタグを付ける

アプリケーションにタグを付けることで、アプリケーションが整理され、効率的な検索が可能になります。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. タグを付けるには：
   a. 1 つのアプリケーションにタグを付けるには、アプリケーションの行の最後にカーソルを合わせて、**タグ**アイコン(🏷)を選択します。
      もしくは、アプリケーションの「概要」ページに移動し、一覧の上部にある**タグ**アイコン(🏷)を選択します。
   b. 複数のアプリケーションにタグを付けるには、各アプリケーションの横にあるチェックマークをオンにし、一覧の下部にあるアクションメニューから**タグ**アイコンを選択します。

3. 「アプリケーションにタグを付ける」画面で、文字を入力すると、既存のタグの一覧が表示されます。使用するタグを選択するか、新しいタグを入力します。
   タグを追加すると、そのアプリケーションの「概要」タブでアプリケーション名の横にタグが表示されるようになります。
   タグを外すには、タグ名の **x** をクリックします。

4. タグを使用してフィルターをかけるには、アプリケーションの列の横にあるフィルターアイコン(▼)を選択し、「アプリケーションのタグ」から選択します。

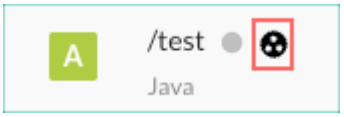
アプリケーションのマージとマージ解除

複数のアプリケーションをマージすることで、メインアプリケーションとなる 1 つのアプリケーションにまとめることができます。これは、組織の管理者がアプリケーションを管理する上で一般的な操作です。

マージを行う主な目的は、スコア付け、脆弱性の検出や修復などを単一のアプリケーションとして参照するためです。複数のモジュールから構成されているアプリケーションは、アプリケーションの一覧で個別に表示される場合があります。マージすることで、アプリケーションの全てのモジュールを論理的に Contrast で 1 つの実体としてまとめることができます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択し、チェックマークを使用してマージするアプリケーションを選択します。
2. ページ下部に表示されるメニューで**マージ**アイコン()を選択します。
3. 「アプリケーションをマージ」画面で、ドロップダウンにあるマージ対象のアプリケーションから、メインアプリケーションとなるアプリケーションを 1 つ選択します。
4. アプリケーションがマージされると、メインアプリケーションの名前の横にメインアプリケーションのアイコン()が表示されます。



5. マージされたアプリケーションにあるアプリケーションモジュールと、疎通されたルート情報を確認するには、アプリケーションの行でメインアプリケーションのアイコンを選択します。

6. メインアプリケーションから全てのアプリケーションモジュール、または特定のアプリケーションモジュールのマージを解除するには、アプリケーションの行または概要ページでメインアプリケーションアイコン(☢)を選択します。「アプリケーションモジュール」画面で、任意の数のモジュールを選択したら、**選択項目のマージを解除**を選択します。

アプリケーションのアーカイブとアーカイブ解除

アプリケーションで脆弱性の収集の必要がなくなっても履歴のために組織に残したい場合は、そのアプリケーションをアーカイブするのが最適な方法です。

アプリケーションをアーカイブすると、脆弱性やライブラリなど、過去のアプリケーションデータの整合性は維持されますが、エージェントは Contrast に脆弱性を報告しなくなります。

アーカイブされたアプリケーションは総合スコアに含まれないため、ポートフォリオ全体のスコアも改善します。

管理者は、アーカイブされたアプリケーションを復元できます。アプリケーションのアーカイブを解除すると、そのアプリケーションはデフォルト表示のアプリケーション一覧で表示されます。全ての脆弱性や問題は、すぐにスコアに反映されます。

開始する前に

- アプリケーションをアーカイブしても、Contrast のライセンスは解放されません。ライセンスを利用可能なライセンスプールに戻すには、アプリケーションをアーカイブして、完全に削除 (535ページ)する必要があります。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アーカイブするアプリケーションを検索します。
3. アプリケーションをアーカイブするには：
   a. 1 つのアプリケーションをアーカイブするには、アプリケーションの行の最後にカーソルを合わせて**アーカイブ**アイコン(🗃)を選択します。

b. 複数のアプリケーションをアーカイブするには、各アプリケーションの横にあるチェックマークをオンにし、一覧の下部にあるアクションメニューから**アーカイブ**アイコン(🗃)を選択します。

c. 表示された画面で、**アーカイブ**を選択し処理を確定します。

4. アプリケーションのアーカイブを解除するには：
   a. アプリケーション一覧の上部で、アプリケーションの見出しの横にある小さい三角形(˅)を選択します。
   b. **アーカイブ済**を選択します。
   c. 1 つのアプリケーションのアーカイブを解除するには、アプリケーションの行の最後にカーソルを合わせて**アーカイブを解除**のアイコン(📤)を選択します。
   d. 複数のアプリケーションのアーカイブを解除するには、各アプリケーションの横にあるチェックマークをオンにし、一覧の下部にあるアクションメニューから**アーカイブを解除**のアイコンを選択します。

   e. 表示された画面で、**アーカイブを解除**を選択して処理を確定します。

アプリケーションのリセット

アプリケーションをリセットすると、そのアプリケーションに関連付けられているデータが全て消去されますが、アプリケーションは削除されません。アプリケーションのリセットは、特定のアプリケーションに関する全ての履歴や検出結果を消去したい場合に便利です。

アプリケーションを削除する前にリセットを行い、関連する全ての脆弱性、URL、コンポーネントなどを適切に消去するのが一般的です。

リセットによる処理

- **基本的な処理：**
  アプリケーションをリセットすると、過去の脆弱性データ、ライブラリデータ、およびルートカバレッジのデータが失われます。ライセンスの関連付けとエントリは Contrast で保持され、サーバとの関連付けも保持されます。
- **アプリケーションまたはサーバがオンラインの場合：**
  オンラインのアプリケーション、またはサーバがオンラインのアプリケーションをリセットすると、脆弱性、ライブラリ、およびルートの数は 0 になります。アプリケーションのエンドポイントをいくつか閲覧すると、脆弱性、ライブラリおよびルートのデータが再度作成されます。ただし、ルート検索はアプリケーションの起動時に行われるため、検出されたルートは無いままとなります。
- **アプリケーションまたはサーバがオフラインの場合：**
  アプリケーションまたはサーバがオフラインの時にアプリケーションをリセットした場合、再起動後、全てのデータが想定通りに読み込まれます。

開始する前に

1 度にリセットできるのは、1 つのアプリケーションのみです。

> ⚠ **注意**
> アプリケーションをリセットした場合、消去されたデータを回復する方法はありません。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. 行の右端にカーソルを合わせ、**リセット**アイコン( )を選択します。
   あるいは、アプリケーションの「概要」タブで**設定**アイコン( )を選択し、**アプリケーション をリセット**を選択します。
3. 「アプリケーションをリセット」の画面で、**リセット**を選択します。

アプリケーションの削除

アプリケーションを削除すると、関連する検出結果(脆弱性やライブラリなど)が全て完全に削除されます。

アプリケーションに適用されたライセンスは、最大許容アプリケーション数に対して永続的にカウントされます。ライセンスが適用されたアプリケーションを削除しても、アプリケーションに許容されるライセンス数には影響はありません。

開始する前に

- **任意**：アプリケーションをアーカイブして削除する前にアプリケーションをリセット (534ページ)します。
  アプリケーションをリセットすると、そのアプリケーションに関連付けられているデータが全て消去されますが、アプリケーションは削除されません。アプリケーションをリセットして、特定のアプリケーションに関連する全ての履歴や結果を消去するのは、よくあるユーザ操作です。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. 削除するアプリケーションを検索します。
3. アプリケーションをアーカイブするには：
   a. 1 つのアプリケーションをアーカイブするには、アプリケーションの行の最後にカーソルを合わせて**アーカイブ**アイコン( )を選択します。
   b. 複数のアプリケーションをアーカイブするには、各アプリケーションの横にあるチェックマークをオンにし、一覧の下部にあるアクションメニューから**アーカイブ**アイコン( )を選択します。
   c. 表示された画面で、**アーカイブ**を選択します。
4. ページ上部のアプリケーションの見出しの横にある**ドロップダウン**アイコン( )を選択し、**アーカイブ済**を選択します。
5. アプリケーションを削除するには：
   a. 1 つのアプリケーションを削除するには、削除したいアプリケーションの行の最後にカーソルを合わせて**削除** ( )アイコンを選択します。
   b. 複数のアプリケーションを削除するには、各アプリケーションの横にあるチェックマークをオンにし、一覧の下部にあるアクションメニューから**削除**アイコン( )を選択します。

   c. 表示された画面で、**削除**を選択します。

セッションメタデータフィルタの使用

セッションメタデータを使用すると、特定のブランチ、ビルド、コミットしたユーザ、リポジトリなどによって、脆弱性やルート情報をフィルタリングできます。エージェントの設定ファイルにセッションメタデータとして必要な設定情報を追加 (537ページ)すると、標準の脆弱性データとともに追加の情報がエージェントによって Contrast に報告されます。

セッションには、エージェントの設定ファイルで設定したメタデータ値を組み合わせることができます。定義した値に応じて、各エージェントの実行を単独のセッションの 1 部にすることもできますし、全てのエージェントの実行に独自のセッション情報を持たせることもできます。Contrast を CI/CD パイプラインに組み込んでいる場合は、アプリケーションの新しいバージョンをデプロイするたびに、一意のセッションメタデータ値を少なくとも 1 つ送信するようします。例えば、コミットハッシュ(commitHash)やビルド番号(buildNumber)などの値は、アプリケーションのデプロイごとに変わる可能性が高いため、これらのメタデータがエージェントから送信されるように設定します。

特定のセッションメタデータフィルタを選択しない場合、脆弱性一覧の「セッション」列には、エージェントの設定ファイルで指定された値のうち最大 10 個までが表示されます。この制限は、Contrast Web インターフェイスで脆弱性とシンクグループのデータを正しく表示するためのものです。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. 一覧からアプリケーション名をクリックします。
3. **脆弱性**タブまたは**ルートカバレッジ**タブのいずれかを選択します。
4. 画面の右上にあるセッションメタデータのアイコン(🗗)を選択します。
   セッションメタデータフィルタを適用すると、「脆弱性」および「ルートカバレッジ」の一覧に影響します。
5. **最新のセッション**を選択し、**フィルタを適用**を選択すると、直近のセッションのデータが表示されます。
   脆弱性一覧では、「セッション」列と「〜で表示」(〜はプロパティ名)フィルタが非表示になり、直近のセッションの情報が、セッションメタデータのアイコンの上に表示されます。

6. 特定のセッションの情報を表示するには、**カスタムのセッション**を選択し**フィルタを適用**を選択します。
   a. **システムプロパティ**で、表示されているプロパティを 1 つ選択します。
   b. **値**で入力し始めると、選択したプロパティに対する値を検索できます。

7. セッションメタデータフィルタをクリアするには、以下のいずれかの方法を使用します。
   - セッションメタデータのアイコン(🗗)を選択し、**フィルタをクリア**を選択します。

- セッションメタデータのアイコン ( )の上に表示されている**クリア**を選択します。

脆弱性一覧では、セッションメタデータフィルタをクリアすると、「セッション」列が表示されます。

セッションメタデータの設定

アプリケーションのセッションメタデータを Contrast に送信するには、エージェントの設定ファイルにセッションメタデータのキーと値のペアを追加します。

エージェントが、メタデータとして Contrast サーバに報告できるのは、以下のビルドプロパティです。以下のプロパティの全てまたは一部を指定することができます。プロパティを設定ファイルに指定すると、メタデータが利用できるようになり、メタデータを各脆弱性の追加情報として参照したり、フィルターに使用することができます。

この設定は、システムプロパティ、環境設定、または YAML 設定ファイルのプロパティとして指定できます。

名前	設定
コミットハッシュ	`commitHash`
コミットしたユーザ	`committer`
ブランチ名	`branchName`
Git タグ	`gitTag`
リポジトリ	`repository`
テストラン	`testRun`
バージョン	`version`
ビルド番号	`buildNumber`

メタデータの文字列の形式は、RFC 2253 に準拠した文字列で、キー=値のペアで定義する必要があります。以下の文字は含めないでください。

- 文字列の先頭に空白またはハッシュ(#)文字
- 文字列の末尾に空白
- 特殊文字：カンマ(,)、プラス記号(+)、二重引用符(“)、 バックスラッシュ(\)、左山括弧(<)、右山括弧(>)、セミコロン(;)

以下の例で、セッションメタデータを設定する方法をいくつか紹介します。

- **Java のシステムプロパティを使用する場合：**javaagent フラグを指定している行に、追加のプロパティを含めます。この場合、contrast.application.session_metadata プロパティを設定し、実行したテストを識別するためのキーと値をペアにして指定(RFC 2253 に準拠)します。

```
-Dcontrast.application.session_metadata="branchName=feature/some-new-
thing,committer=Jane,repository=Contrast-Java"
```

- **.NET Framework で *app.config* または *web.config* を使用する場合：** 設定ファイルに項目を追加して、メタデータのプロパティを指定します。

```
<?xml version="1.0"?>
<configuration>
  <connectionStrings />
      <appSettings>
        <add key="contrast.application.session_metadata" \
value="branchName=feature/some-new-
thing,committer=Jane,repository=Contrast-DotNet" />
```

- **YAML 設定ファイルの場合：***contrast_security.yaml* ファイルに項目を追加して、メタデータのプロパティを指定します。

```
application:
 session_metadata: branchName=feature/some-new-
thing,committer=Jane,repository=Contrast-Ruby
```

- **CI(継続的インテグレーション)のビルドスクリプトの場合：** 環境変数を使用して値を指定できます。

```
-Dcontrast.application.session_metadata="branchName=feature/some-new-
thing,committer=Jane,repository=Contrast-Java,buildNumber=$BUILD_NUMBER"
```

```
-Dcontrast.application.session_metadata="branchName=$GIT_BRANCH,committer=
$GIT_COMMITTER_NAME,commitHash=$GIT_COMMIT_HASH,repository=$GIT_URL,buildN
umber=$BUILD_NUMBER"
```

ルートカバレッジ

ルートカバレッジにより、Assess ユーザは脆弱性と元の Web リクエストを関連付けることができます。

ルートカバレッジによって、どのルートが疎通されていて、どのルートが疎通されていないかなど、アプリケーションのコンポーネントに関する詳細な情報を確認できます。これは、どこにテストや修復を重点的に行うかを判断するのに役立ちます。

Web リクエストの例

Web リクエストは、Web アプリケーションの最も基本となるインターフェイスです。リクエストを処理する関数は、他のサービスやデータベース、ファイルなどと連携を行う後続の関数を呼び出す場合があります。

リクエストを処理するプロセスで、Contrast はアプリケーション全体のデータフローを監視し、脆弱性を特定します。1 つの Web リクエストが、複数の種類の攻撃に対して脆弱な場合があります。Contrast では、これらの脆弱性を元のリクエストに関連付けます。

以下は、Web リクエストの例です。

```
GET /users?active=true
Host: YourDomain.com
Accept: application/json
```

以下は、上記の Web リクエストがどのように関数で処理されるかの例です。

```
@Controller
public class UserController {
    @GetMapping("/users")
    public String users(@RequestParam(name="active", required=false, \
defaultValue=true) Bool active) {
        ...
    }
}
```

ルートカバレッジの仕組み

アプリケーションのルートは、次の 3 つの要素の組み合わせです。

- HTTP 動詞(例、GET)
- リソースのパス(例、/users)
- コントローラのメソッドシグネチャ(例、UserController.users(Bool active))
  対応するシグネチャが使用されていない場合には、route.jsp、route.xhtml、/route/id/{id} など、メソッドシグネチャの代わりにルートテンプレートが使用されます。

Contrast エージェントを起動すると、アプリケーション内の関数にエージェントが組み込まれ、アプリケーションを実行中に Web リクエストの脆弱性が評価されます。関数に Web リクエストを処理するためのフレームワークが実装されていれば、リクエストが処理される前に Contrast でルートが識別されます。そのようなルートは、Contrast では**検出済**というステータスになります。

アプリケーションがリクエストを処理すると、そのアクティビティは Contrast によって観察されて、**疎通済**ルートとなります。

フレームワーク とテクノロジ

Contrast では、以下のフレームワークのルート検出をサポートします。

- **Java**：現在サポート対象のテクノロジ (75ページ)
- **.NET Framework**：現在サポートされている全てのフレームワーク (167ページ)
- **.NET Core**：現在サポートされている全てのフレームワーク (225ページ)
- **Node.js**：現在サポートされている全てのフレームワーク (285ページ)
- **Python**：現在サポートされている全てのフレームワーク (360ページ)
- **Ruby**：現在サポートされている全てのフレームワーク (417ページ)
- **Go**： 現在サポートされている全てのフレームワーク (476ページ)

お使いのフレームワークがサポートされていない場合は、サポートまでお問い合わせください。サポート対象外のフレームワークの場合、Contrast では観測されたリクエストに基づいてルートを推測しようとうとしますが、ルートは検出されず Contrast には表示されません。

対象外(内蔵されたルートやアプリケーション)

Contrast のルートカバレッジには、一部の Web フレームワークやアプリケーションに内蔵されたルートは含まれません。 例えば、以下のような場合です。

- Java アプリケーションの Jersey フレームワークには、WADL ファイルを提供するため機能があり、そのルートが内蔵されています。このルートは、Contrast のルートカバレッジ には含まれません。他の Web フレームワークにも同様に、フレームワークに内蔵されているルートがあります。
- Contrast Java エージェントは、Tomcat の管理マネージャ(Tomcat Manager Application)などの備え付けのアプリケーションからのルートを報告しません。

ルート情報の表示

ルートカバレッジ (538ページ)は、脆弱性がアプリケーションの攻撃対象領域に対して、どのように関連付けられるかを理解するのに役立ちます。

ルートカバレッジの一覧からルートを削除しても(手順 8 および 9)、サーバの再起動時やアプリケーションの疎通時にルートがまだ存在する場合に、そのルートは再度カバレッジの対象に含まれることにな

ります。ルートを完全にカバレッジの対象外にする場合は、そのルートの行の右端にある**除外**アイコン(🚫) を選択してください。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーションの名前を選択します。
   アプリケーションの概要タブには、アプリケーションの合計ルート数に対する疎通済みのルート数が表示されます。
3. 概要タブで、疎通済みのルート数を選択するか、**ルートカバレッジ**タブを選択します。

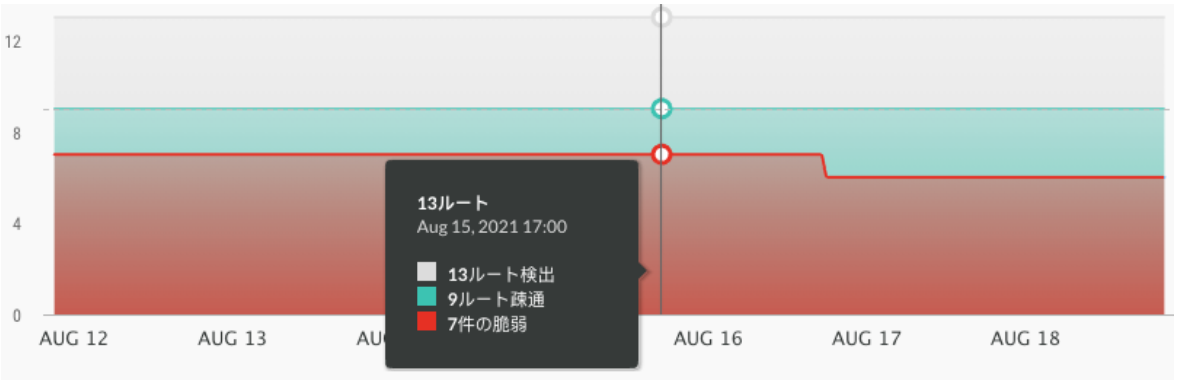
4. ルートカバレッジタブでは、「ルートカバレッジ概要」に次の情報が表示されます。

- **疎通済ルート**： このセクションには、以下の情報が表示されます。
  - 検出されたルートのうち、疎通されているルートの割合。
  - 疎通されたルートの数。
- **セッションメタデータ**： このセクションには、適用されたセッションメタデータのフィルタに基づいて、以下の情報が表示されます。

> **注記**
> セッションメタデータのフィルタを適用していない場合、値は表示されません。セッションメタデータの値を参照するには、**フィルタを適用**または**フィルタを編集**(現在のフィルタを変更する場合)を選択し、使用するフィルタを指定 (535ページ)します。

  - 適用されたフィルタの条件に一致する、疎通済みルートの割合
  - 適用されたフィルタの条件に一致する、疎通済みルートの数
  - 選択中のセッション
  - アプリケーションのリポジトリ
  - ビルド番号
  - ブランチ名
  - コミットしたユーザ
5. ルートカバレッジタブでは、セッションメタデータのフィルタを適用していない場合、ルートカバレッジのグラフが表示されます。

> **注記**
> セッションメタデータのフィルタを既に選択している場合は、このグラフは表示されません。グラフを表示するには、**フィルタをクリア**を選択します。



グラフには、ルートのステータスに基づいてルートの詳細が表示されます。
- Contrast で検出されたルート
- Contrast エージェントを使用して疎通されたルート
- 疎通されて脆弱性があると判明したルート

6. ルートカバレッジの一覧には、各ルートの詳細が表示されます。
   - **ルート**：Contrast で特定されたルート、または追跡中のルート。
   - **サーバ**：アプリケーションが稼働しているサーバ。
     デフォルトでは、サーバ列に表示されるのは 3 サーバまでです。3 サーバを超える場合に、サーバの全リストを表示するには、**全て(x サーバ)を表示**を選択します(x はサーバの合計数)。

> **注記**
> サーバを削除すると、サーバはグレー表示されるのではなく一覧から削除されます。

   - **脆弱性**：そのルートに関連付けられた脆弱性の数。
   - **最後のアクティビティ**： そのルートのアクティビティ期間。
   - **ステータス**：そのルートのステータス。
7. 一覧より各項目を選択すると、アプリケーションで特定された各ルートの詳細情報が表示されます。
   a. ルートの URL を表示するには、ルート名を選択します。
   b. 特定のルートの脆弱性の詳細を表示するには、脆弱性列の深刻度バーのセクションを選択します。各セクションには、深刻度(重大、高、中、低、注意)が表示されています。
   c. アクティビティ期間に基づいてルートを表示するには、「最後のアクティビティ」列の横にあるフィルタアイコン(▼)を選択します。
      期間を変更すると、ルートカバレッジのグラフの期間も変更されます。
      フィルタの選択を解除するには、列の見出しの横にある**クリア**を選択します。
   d. 特定のステータスのルートのみを表示するには、ステータス列の横にあるフィルタアイコン(▼)を選択します。ステータスには、以下があります。
      - **全て**：除外されていない全てのルートを表示します。

- **疎通済**：疎通されたルートのみを表示します。
- **未疎通**：検出されたルートで、まだ疎通されていないルートを表示します。
- **脆弱なもの**：脆弱性が関連付けられているルートのみを表示します。
- **対象外**：アプリケーションのスコア付けの計算から除外されるルートのみを表示します。

8. 一覧から 1 つのルートを削除するには：
   a. 行の右端にカーソルを合わせ、**削除**アイコン(🗑)をクリックします。
   b. ルートの削除の確認画面で、**削除**を選択します。
9. 一覧から複数のルートを削除するには：
   a. 1 つまたは複数のルートの行でチェックマークを選択します。また、全てのルートを選択する場合は、**ルート**の横にあるチェックマークを選択します。
   b. ページの下部に表示される一括アクションメニューで、**削除**アイコン(🗑)を選択します。
   c. ルートの削除の確認画面で、**削除**を選択します。
10. ルートの情報を Contrast の外部で参照・共有するには：
   a. 1 つまたは複数のルートの行でチェックマークを選択します。また、全てのルートを選択する場合は、**ルート**の横にあるチェックマークを選択します。
   b. ページの下部に表示される一括アクションメニューで、**エクスポート**アイコン(⬇)を選択します。
   この操作によって、ルート情報が CSV ファイルにエクスポートされます。ファイルは、デフォルトのダウンロード先にダウンロードされます。
   CSV ファイルには次のものが含まれます：
   - アプリケーションのルートのリスト
   - ルートが検出されたサーバの情報
   - ルートが最後に疎通された日付
   - 脆弱性の一覧、各脆弱性の深刻度およびステータス

関連項目

- ルートを除外する (543ページ)
- ルートを含める (543ページ)

除外ルートと含めるルート

セキュリティ上の理由から特定の割合のルートカバレッジが必要な場合、ルートカバレッジの計算から無関係なルートやアクセスできないルートを含めないよう除外するオプションが Contrast にはあります。

既に除外したルートは、再度対象に含めることもできます。

ルートを除外したり含めるには、Admin(管理者)ロールが必要です。

ルートを対象に含めない場合の影響

- Contrast では、除外されたルートの脆弱性データは収集されますが、このデータはアプリケーションのスコア付けの計算に含まれなくなります。
- ルートを除外すると、Contrast がそのルートを検出した全ての環境から除外されます。
- セッションメタデータを定義してアプリケーションで疎通やルート検出した場合、ルートを除外すると、このデータも除外されます。
- 監査ログには、除外された各ルートのエントリは含まれます。

ルートを対象に含める場合の影響

- 除外されていないルートのデータは、アプリケーションのスコア付けの計算に含まれます。
- ルートを対象に含めると、Contrast がそのルートを検出した全ての環境にそのルートが含まれるようになります。

- 監査ログには、対象に含めた各ルートのエントリが含まれます。

ルートを除外する

無関係なルートやアクセスできないルートを除外する (542ページ)ことで、アプリケーションのルートカバレッジの計算が正確になります。

ルートを確実に除外したままにするには、ルートを除外した後に、そのルートを削除しないことです。ルートを除外してから削除すると、アプリケーションサーバの起動時やアプリケーションの疎通後にそのルートが検出された場合に、ルートは再度カバレッジに含まれる対象となります。

開始する前に

- カバレッジから除外するルートを決めます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーション名を選択します。
3. **ルートカバレッジ**タブを選択します。
4. 1 つまたは複数のルートを除外します：
   a. 除外したいルートが 1 つの場合は、行の最後にカーソルを合わせ**除外**アイコン( )を選択します。
   b. 複数ルートを除外したい場合は、左側の列のチェックマークを使用して、ルートを選択します。次に、ページ下部にある一括アクションメニューから**除外**アイコンを選択します。

   c. 表示される画面でルートの除外を確定します。
   選択したルートのステータスが、**対象外**に変わります。

ルートを含める

除外されたルートは、再度カバレッジにルートを含める (542ページ)ことができます。除外されたルートは、ルートカバレッジのページで**対象外**のステータスになります。

開始する前に

- カバレッジに含めるルートを決めます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーション名を選択します。
3. **ルートカバレッジ**タブを選択します。
4. 一覧の上部にある**三角形** ( )を選択して**対象外**を選択すると、カバレッジから除外されているルートが表示されます。

5. 1 つまたは複数のルートをカバレッジに含めます：
   a. 除外されている 1 つのルートをカバレッジに含めるには、行の最後にカーソルを合わせ**再度含める**アイコン( )を選択します。
   b. 除外されている複数のルートを選択するには、左の列のチェックマークを使用してルートを選択します。次に、ページ下部にある一括アクションメニューから**再度含める**アイコンを選択します。

   c. 表示される画面で、ルートをカバレッジに含めることを確定します。

ステータスは、ルートを除外する前のステータスに戻ります。

フローマップ

アプリケーションのフローマップによって、組織内および組織外のデータとリソースの共有状況がインタラクティブに表示されます。

アプリケーションを実行するたびに、Contrast エージェントから報告されるデータを使用して、アプリケーション、アプリケーション内のテクノロジ層、およびアプリケーションが接続しているバックエンドシステムなどの構成を表すダイアグラムが Contrast で作成されます。組織内でより多くのアプリケーションが実行され、エージェントによってアプリケーションのバックエンドシステムが認識されると、現在参照中のアプリケーションに対して、共有バックエンドシステムで接続しているアプリケーションも認識されます。

フローマップを表示 (545ページ)すると、アプリケーションに関連付けられているシステムとリソースの全体像を確認できます。個々のシステムとアプリケーション間の接続に注目することで、組織内のユーザや接続されたアプリケーションが、現在のアプリケーションとアプリケーションに関連する可能性のある機密情報に対して、適切なアクセスがあるかを判断することもできます。詳細については、フローマップの理解 (545ページ)を参照してください。

エージェントは、文字列認証によりアプリケーションの照合を行います。計測された他のアプリケーションで、共通の文字列認証情報(例えば、REST エンドポイント、データベース接続、その他の一意のホストとポートの組み合わせなど)を共有しているものが、接続中のアプリケーションとして表示されます。

注記

接続中のアプリケーションの情報を表示する権限のない (822ページ)ユーザには、フローマップにそのアプリケーションは表示されません。

フローマップの表示

アプリケーションのフローマップによって、組織内および組織外のデータとリソースの共有状況がインタラクティブに表示されます。

アプリケーションのフローマップを表示するには：

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択して、アプリケーションの一覧よりアプリケーション名をクリックします。
2. **フローマップ**タブをクリックします。
3. フローマップタブには、3 つの関連するセクションがあります：**アプリケーションアーキテクチャ**、**バックエンドシステム**、**アプリケーション**です。詳細については、フローマップデータの理解 (545ページ)を参照してください。

フローマップの理解

アプリケーションのフローマップによって、組織内および組織外のデータとリソースの共有状況がインタラクティブに表示されます。

アプリケーションのフローマップ (545ページ)は、3 つの関連セクションで情報が編成されます。

- **アプリケーションアーキテクチャ**：このセクションでは、アプリケーションのフロントエンドをビュー、プレゼンテーション、サービスの階層に分類します。また、アプリケーションがデプロイされている環境、レターグレード、脆弱性ステータスおよび攻撃ステータスなどアプリケーションに関する基本的な情報も確認できます。このセクションの 3 つの層は、以下の通りです。
  - **ビュー**：ブラウザでの表示および処理を決定するテクノロジ層を表します。
  - **プレゼンテーション**：アプリケーションのビューを生成するライブラリ層を表します。
  - **サービス**：アプリケーションロジックを実行しているデータベース、LDAP ドライバ、バックエンドコードなどで構成される層を表します。

  リスト内の項目にカーソルを合わせると、アプリケーションで使用されている各タイプのライブラリのインスタンス数が表示されます。また、ライブラリをクリックするとライブラリのページに移動します。エージェントから脆弱性が報告されると、検出されたライブラリの横に警告アイコンが表示されます。アイコンにカーソルを合わせると脆弱性の**概要**ページへのリンクが表示されます。
- **バックエンドシステム**：アプリケーションが接続されている各システムを表示します。データベースのシリンダアイコン、URL のグローブアイコン、または LDAP データベースのプラグアイコンにカーソルを合わせると、各システムの詳細を参照できます。アイコンをクリックすると、ほかのアプリケーションへの接続が強調表示されます。鍵付きの実線は、接続が暗号化されていることを表します。破線は、接続が暗号化されていないか、暗号化の状態が不明であることを表します。
- **接続中のアプリケーション**：バックエンドシステムによってプライマリアプリケーションに接続されている各アプリケーションの一覧を表します。特定の条件を満たす接続中のアプリケーションを表示するには、ファネルアイコンをクリックし、ドロップダウンから、環境、アプリケーション言語、カスタムタグなどのフィルターを選択します。このメニューには、参照可能な場合、プライマリアプリケーション(接続されているアプリケーションではありません)のセッションメタデータフィールドも表示されます。 **フローマップを表示**のリンクをクリックすると、そのアプリケーションの**フローマップ**タブに移動します。

注記

現在のアプリケーションに対してエージェントから報告されているデータがない場合は、**バックエンドシステム**と**接続中のアプリケーション**のセクションは空欄のままです。

ヒント
フローマップを表示しているときに別のユーザがアプリケーションにアクセスしている場合は、**ブラウザ**タブが表示され、アクセスしているブラウザの一覧が表示されます。アイコンにカーソルを合わせると、ブラウザの種類やバージョンなどの詳細を参照できます。

スキャン

Contrast Scan は、静的アプリケーションセキュリティテスト(SAST)ツールで、コードをすぐにスキャンして、開発の初期段階で脆弱性を特定できます。

スキャンには以下の種類があります。

- **Java バイナリ**： Java の JAR ファイルや WAR ファイルをスキャンできます。
- **ソースコード**： ほとんどの言語 (554ページ)のアーティファクトをスキャンできます。
- **ローカルスキャンエンジン**: ローカルシステムでアーティファクトをスキャンできます。Contrast に結果は報告されますが、ローカルコードがアップロードされることはありません。

スキャンの操作

Contrast Scan では、次のことができます。

- スキャンプロジェクトを作成 (557ページ)
- スキャンプロジェクトをアーカイブ (583ページ)
- スキャンプロジェクトを削除 (559ページ)
- スキャンプロジェクトをアーカイブ (583ページ)
- スキャン結果を分析 (569ページ)
- 新規スキャンを開始 (560ページ)
- スキャンをキャンセル (561ページ)
- スキャン設定を変更 (582ページ)
- ローカルでスキャンを実行 (563ページ)
- SAST コンプライアンス対応レポートを作成 (562ページ)

関連項目

Contrast Scan のサポート対象テクノロジ (554ページ)

Contrast Scan リリース情報

スキャンエンジン・ローカルスキャンエンジン

Contrast Scan 1.0.4

リリース日： 2023 年 12 月 14 日

注記
このバージョンのローカルスキャンエンジンは、リクエストによってのみ利用可能です。今回、Contrast はチェックサム情報を公開しません。

このバージョンのローカルスキャンエンジンのアクセスをリクエストするには、Contrast の通常のサポートプロセスに従ってください。

新しいローカルスキャンエンジンは、近日中に、一般利用を可能にする予定です。

*[en] **New and improved:***

- *[en] Increased the memory that the multi-language source code scan engine uses to 2G to better support larger code bases. The minimum memory requirement when using the local scan engine is still 12GB.*
- *[en] Added a `--memory` parameter to the CLI that you can use to override the allocated memory for the multi-language source code scan engine.*
- *[en] Added additional logging to capture the parameters used when invoking the local scan engine. This logging captures the entire invocation command for the local scan engine (for example, `-r`, `-p` and so forth) for use when troubleshooting errors.*

重要
多言語ソースコードに対応したこの新しいスキャンエンジンは、バージョン 1.0.4 となっています。バージョン 1.0.0、1.0.1、1.0.2 は、多言語スキャンエンジンの内部テストおよびベータ版とみなし、Contrast のお客様がダウンロードすることはできません。

アプリケーションの署名の確認

ダウンロードしたローカルスキャンエンジンが、Contrast によって作成され署名されているかを確認するには、次のコマンドを実行します。

```
jarsigner -verify -verbose -certs sast-local-scan-runner-0.0.XX.jar
```

XX は、確認したいローカルスキャンエンジンのバージョンに置き換えてください。

Contrast Scan 1.0.4

リリース日： 2023 年 12 月 14 日

注記
このバージョンのローカルスキャンエンジンは、リクエストによってのみ利用可能です。今回、Contrast はチェックサム情報を公開しません。

このバージョンのローカルスキャンエンジンのアクセスをリクエストするには、Contrast の通常のサポートプロセスに従ってください。

新しいローカルスキャンエンジンは、近日中に、一般利用を可能にする予定です。

修正された不具合：

- VB.NET および Scala のソースコードが、多言語エンジンによって正しく識別、スキャンされない不具合を修正しました。

重要
多言語ソースコードに対応したこの新しいスキャンエンジンは、バージョン 1.0.4 となっています。バージョン 1.0.0、1.0.1、1.0.2 は、多言語スキャンエンジンの内部テストおよびベータ版とみなし、Contrast のお客様がダウンロードすることはできません。

アプリケーションの署名の確認

ダウンロードしたローカルスキャンエンジンが、Contrast によって作成され署名されているかを確認するには、次のコマンドを実行します。

```
jarsigner -verify -verbose -certs sast-local-scan-runner-0.0.XX.jar
```

XX は、確認したいローカルスキャンエンジンのバージョンに置き換えてください。

Contrast Scan 1.0.3

リリース日： 2023 年 11 月

> **注記**
> Contrast Scan ローカルエンジン 1.0.3 は、現在制限付きのリリースとなっています。そのため、現時点ではチェックサム情報は提供されません。
>
> このバージョンをご利用頂くには、サポートチケットを発行してリクエストしてください。ご不便をおかけして申し訳ございませんが、早急に対応させて頂きます。

新機能と改善点：

2023 年 11 月 29 日

- ロールベースのアクセス制御の認証の問題を修正しました。空のリソースグループにプロジェクトを割り当てようとした場合や、ユーザが複数のリソースグループにアクセスできるにも関わらずリソースグループを指定しなかった場合に、403 エラーがトリガーされる可能性がありました。
  ロールベースのアクセス制御が有効になっている場合、スキャンプロジェクトの作成時に Contrast CLI の `-r <>`オプションが必須になりました。

2023 年 11 月 8 日

- Contrast Scan ローカルエンジンは、25 以上の言語のソースコードのスキャンに対応するようになりました。対応する言語の一覧は、Contrast Scan のサポート対象言語 (554ページ)を参照して下さい。
- ローカルエンジンは、適切な JVM を実行している Windows 環境でネイティブに実行できるようになりました。
- スキャンするアーティファクトのパスにスペースを使用すると、致命的なスキャンエラーが発生する問題を修正しました。
- ローカルスキャンエンジンから不要なログを削除し、Java バイナリファイル(JAR ファイルまたは WAR ファイル)のスキャン時の全ディスク領域の使用率を削減しました。
- Alpine Linux で実行するとローカルスキャンエンジンが失敗する問題を修正しました。

> **重要**
> 多言語ソースコードに対応したこの新しいスキャンエンジンは、バージョン 1.0.3 となっています。バージョン 1.0.0、1.0.1、1.0.2 は、多言語スキャンエンジンの内部テストおよびベータ版とみなし、Contrast のお客様がダウンロードすることはできません。

アプリケーションの署名の確認

ダウンロードしたローカルスキャンエンジンが、Contrast によって作成され署名されているかを確認するには、次のコマンドを実行します。

```
jarsigner -verify -verbose -certs sast-local-scan-runner-0.0.XX.jar
```

XX は、確認したいローカルスキャンエンジンのバージョンに置き換えてください。

Contrast Scan 0.0.63

リリース日： 2023 年 7 月 24 日

修正された不具合：

- ローカルスキャナで、複数の JAR ファイル内に見つかった全ての脆弱性が報告されていなかったバグを修正しました。ZIP ファイル内で最後にスキャンされた JAR ファイルのみが報告されていました。

チェックサム：

- **MD5 チェックサム：**f57f9174d0643832f9e38b95998fe280
- **SHA チェックサム：** 8b2f5680111c5a4e5999a3449ee871bb822d27f6

注記

チェックサムの生成方法

- **MD5**: 以下のコマンドを使用します。

```
curl -L -H 'Accept: application/vnd.github.v3.raw' -s https://
$CONTRAST_GITHUB_PAT@maven.pkg.github.com/Contrast-Security-
Inc/sast-local-scan-runner/com.contrastsecurity.sast-local-
scan-runner/X.X.XX/sast-local-scan-runner-X.X.XX.jar.md5 -o \
sastXX.md5
```

- **SHA**: 以下のコマンドを使用します。

```
curl -L -H 'Accept: application/vnd.github.v3.raw' -s https://
$CONTRAST_GITHUB_PAT@maven.pkg.github.com/Contrast-Security-
Inc/sast-local-scan-runner/com.contrastsecurity.sast-local-
scan-runner/X.X.XX/sast-local-scan-runner-X.X.X.jar.sha1 -o \
sastXX.sha
```

どちらの種類のチェックサムも、`X.X.XX` の箇所をダウンロードしてチェックサムで検証するエンジンのバージョンに置き換えてください。例えば、エンジンのバージョンが 0.0.60 の場合、`X.X.XX` を `0.0.60` に置き換え、出力(sastXX.sha または sastXX.md5)には、`60` などの現在のバージョンを表す値を指定します。

アプリケーションの署名の確認

ダウンロードしたローカルスキャンエンジンが、Contrast によって作成され署名されているかを確認するには、次のコマンドを実行します。

```
jarsigner -verify -verbose -certs sast-local-scan-runner-0.0.XX.jar
```

XX は、確認したいローカルスキャンエンジンのバージョンに置き換えてください。

Contrast Scan 0.0.60

リリース日： 2023 年 5 月 22 日

新機能と改善点：

- プロジェクトを始めてスキャンする時に、ローカルスキャンエンジンのパラメータとして、リソースグループを指定できる機能が追加されました。
  この機能を利用するには、組織でロールベースのアクセス制御が有効になっており、新しいプロジェクトを作成するための十分な権限(Manage Project ロール以上)が必要です。
  リソースグループ名は、`-r` パラメータで指定します。

チェックサム：

- **MD5 チェックサム：** 0fa38c5c9e46e3b2c6bdb2d2ed3baa20
- **SHA チェックサム：** 76fe00f7d70d45176904a2b62a9d1083f0731a03

注記
チェックサムの生成方法

- **MD5** : 以下のコマンドを使用します。

```
curl -L -H 'Accept: application/vnd.github.v3.raw' -s https://
$CONTRAST_GITHUB_PAT@maven.pkg.github.com/Contrast-Security-
Inc/sast-local-scan-runner/com.contrastsecurity.sast-local-
scan-runner/X.X.XX/sast-local-scan-runner-X.X.XX.jar.md5 -o \
sastXX.md5
```

- **SHA**: 以下のコマンドを使用します。

```
curl -L -H 'Accept: application/vnd.github.v3.raw' -s https://
$CONTRAST_GITHUB_PAT@maven.pkg.github.com/Contrast-Security-
Inc/sast-local-scan-runner/com.contrastsecurity.sast-local-
scan-runner/X.X.XX/sast-local-scan-runner-X.X.X.jar.sha1 -o \
sastXX.sha
```

どちらの種類のチェックサムも、`X.X.XX` の箇所をダウンロードしてチェックサムで検証するエンジンのバージョンに置き換えてください。例えば、エンジンのバージョンが 0.0.60 の場合、`X.X.XX` を `0.0.60` に置き換え、出力(sastXX.sha または sastXX.md5)には、`60` などの現在のバージョンを表す値を指定します。

アプリケーションの署名の確認

ダウンロードしたローカルスキャンエンジンが、Contrast によって作成され署名されているかを確認するには、次のコマンドを実行します。

```
jarsigner -verify -verbose -certs sast-local-scan-runner-0.0.XX.jar
```

`XX` は、確認したいローカルスキャンエンジンのバージョンに置き換えてください。

Contrast Scan 0.0.56 - 0.0.59

リリース日： 2023 年 4 月 6 日

新機能と改善点：

- 複数 JAR のスキャンに対応
  このリリースでは、複数の JAR ファイルを 1 つのアーティファクトとしてスキャンできる機能が追加されました。複数の JAR ファイルを ZIP ファイルに追加し、1 つのアーティファクトとしてスキャンすることができます。

複数 JAR の ZIP ファイルをスキャンするには、最上位レベルで JAR ファイルを ZIP ファイルにパッケージし、通常通り Contrast Scan ローカルエンジンを使用してスキャンします。例：

```
multiple-jar-artifact.zip
-> artifact1.jar
-> artifact2.jar
-> artifact3.jar
```

スキャンが完了すると、Contrast Web インターフェイスでは、「スキャン」タブの下に 1 つのプロジェクトとして表示されます。

修正された不具合：

リリース 0.0.57 から 0.0.59 に、スキャンの動作やパフォーマンスに影響しない内部のバグ修正が含まれています。

スキャン Web インターフェイス

12 月リリース：スキャン Web インターフェイス

リリース日： 2023 年 12 月

新機能と改善点：

- 2023 年 12 月 14 日
  [en] ***New and improved;***
  - *[en] On the Scan project page in the Contrast web interface, you can view the languages that the multi-language source code scan engine detected.*
  - *[en] Added the ability to search for scan projects based on detected languages.*

  [en] ***Bug fixes:***
  - *[en] Fixed a bug in the CLI that suggested a scan had failed when it invoked the multi-language source code scan engine.*
  - *[en] Fixed a bug in the CLI that prevented the list of found vulnerabilities from being displayed in the CLI output once a scan completes.*

12 月リリース：スキャン Web インターフェイス

リリース日： 2023 年 12 月

新機能と改善点：

- 2023 年 12 月 14 日
  - **新機能：**「スキャンプロジェクト」ページおよびスキャンプロジェクトページの「脆弱性」タブから、スキャンプロジェクトのコンプライアンス対応レポートを作成できるようになりました。
  - ユーザが脆弱性のステータスを**修正完了**に変更できる機能を削除しました。このステータスは、後続のスキャンでソースコードに脆弱性がまだ存在するかどうかに基づいて、スキャンエンジンによって判定されます。
  - VB.NET および Scala のソースコードが、多言語エンジンによって正しく識別、スキャンされない不具合を修正しました。

11 月リリース：スキャン Web インターフェイス

リリース日： 2023 年 11 月

新機能と改善点：

- 2023 年 11 月 28 日
  - ロールベースのアクセス制御が有効になっている場合、スキャンプロジェクトの作成にリソースグループの指定が必要になりました。スキャンプロジェクトの作成画面には、プロジェクトを作成しているユーザに割り当てられているリソースグループの一覧を表示するドロップダウンがありま

す。ユーザのロールに割り当てられたリソースグループが 1 つの場合は、このリソースグループがデフォルトの選択になります。
また、ユーザには**プロジェクトの作成**アクションが含まれたロールが必要です。
スキャンプロジェクトの作成 (557ページ)にて、この新しい要件について説明しています。

- 2023 年 11 月 8 日
  - **新機能：** Contrast Scan で 2 種類のスキャンを提供するようになりました。Java ファイル用の Java バイナリスキャンと、その他の多くの言語やテクノロジ用のソースコードスキャンです。ソースコードスキャンを選択した場合、スキャンしたいソースコードが含まれる ZIP ファイルをアップロードします。
  - **新機能：** Contrast Scan のサポート対象言語とテクノロジ (554ページ)に記載されているように、ソースコードスキャンによってスキャンのサポート対象が拡張され、25 以上の言語とテクノロジが追加されました。このソースコードスキャンを使用するには、新しいプロジェクトを作成する際に、**ソースコード**のオプションを選択して下さい。
  - **SaaS 版ご利用の場合：** Contrast Scan では、ソースコードスキャンの多言語検出に対応するようになりました。ZIP ファイルをアップロードすると、ZIP ファイル内に存在する言語がスキャンエンジンによって判別されて、各ファイルがスキャンされます。結果は、Contrast で 1 つのスキャンプロジェクトで表示されます。
  - スキャンプロジェクトの作成時に言語を選択する必要がなくなりました。Contrast Scan 側で、アップロードするコードアーティファクトの種類を判別できるようになりました。複数の JAR やソースコードを含む ZIP ファイルだけでなく、単一の JAR や WAR ファイルも引き続きサポートされます。
  - ダウンロードできる CSV ファイルに 2 つのフィールドを追加しました。
    - **Language(言語)**：脆弱性の言語を示します。
    - **Comment(コメント)**：脆弱性に対する最後のコメントを表します。

    既存のプロジェクトに対して新たにスキャンを実行すると、これらのフィールドが CSV ファイルに入ります。

6 月リリース：スキャン Web インターフェイス

リリース日：2023 年 6 月

新機能と改善点

- 2023 年 6 月 30 日
  - 脆弱性の現在のステータスを変更することなく、脆弱性のステータスにコメントできる機能を追加しました。特定の脆弱性の「アクティビティ」タブで、コメントを追加できます。
- 2023 年 6 月 12 日
  - スキャンページとスキャンの詳細ページの上部にプロジェクト作成者の名前を表示することで、誰がプロジェクトを作成したかを確認できるようになりました。
  - プロジェクトの特定のスキャンを誰が実行したかを確認できるようになりました。
    スキャンページの「スキャン履歴」で、特定のスキャンを実行した人の名前が表示されるよう、名前の列を新規に追加しました。スキャンの詳細ページにもスキャンを実行した人が表示されます。

注記

上記の機能は、いずれも新規プロジェクトおよび新規スキャンに適用されます。既存のプロジェクトやスキャンには、これらの新情報は表示されません。

5 月リリース：スキャン Web インターフェイス

リリース日：2023 年 5 月

新機能と改善点：

- Java バイナリスキャナが複数 JAR のスキャンに対応するようになりました。
  SaaS 版の Java バイナリスキャナを使用する場合(Contrast CLI または Contrast Web インターフェイスを使用)、1 つの ZIP ファイルに複数の JAR ファイルを含めることができるようになりました。

アップロードできる ZIP ファイルのサイズの上限は、1GB です。

4 月リリース：スキャン Web インターフェイス

リリース日： 2023 年 4 月

新機能と改善点：

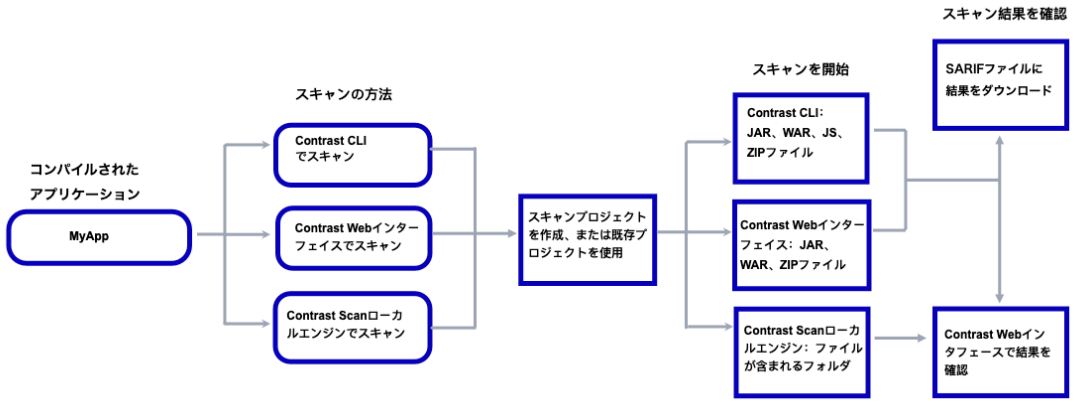
- プロジェクト内の脆弱性に対して行われたステータスの変更に関する情報を表示する、脆弱性のアクティビティタブを追加しました。
  このタブを表示するには、選択したスキャンプロジェクトの「脆弱性」タブを選択し、特定の脆弱性を選択します。
- プロジェクト内の脆弱性のステータスを変更する際に、コメントを追加できるようになりました。
- 全てのプロジェクトを管理できるロールのあるユーザに対して、Contrast Web インターフェイスでプロジェクトと関連する全てのデータを削除できる機能を追加しました。

スキャンのプロセス

本項では、Contrast Scan を使用する際のワークフローと、Contrast Scan がコードの解析に使用するプロセスについて説明します。

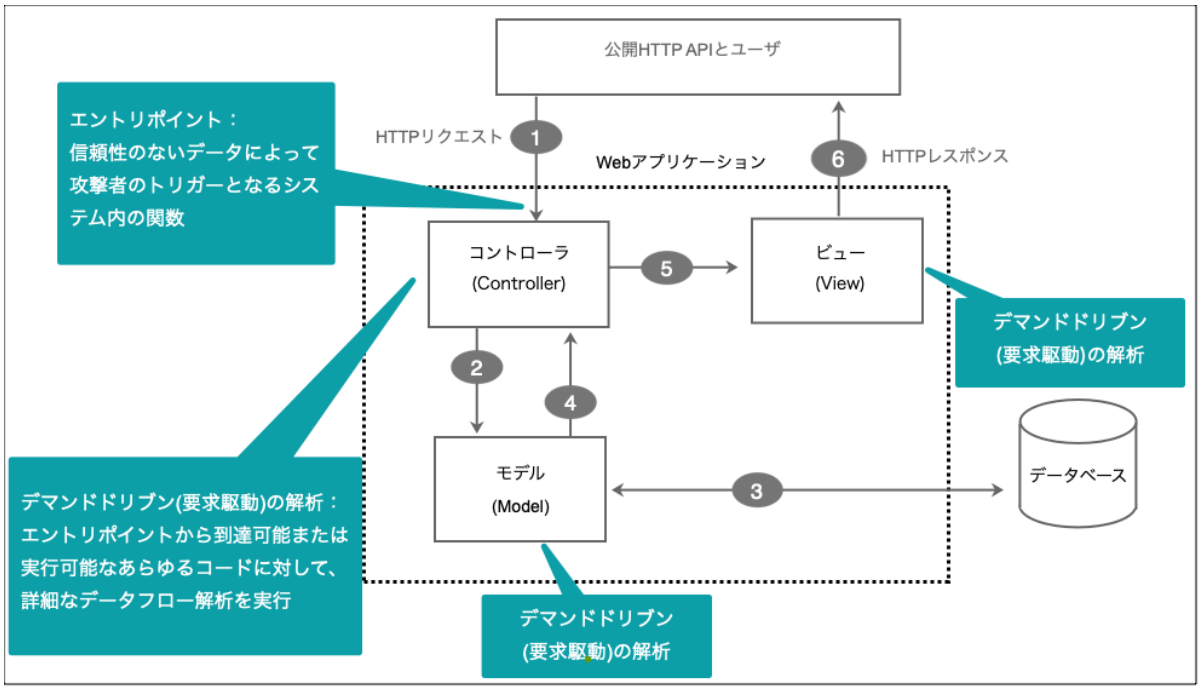
スキャンのワークフロー

次の図は、Contrast Scan を使用する場合のワークフローを示しています。

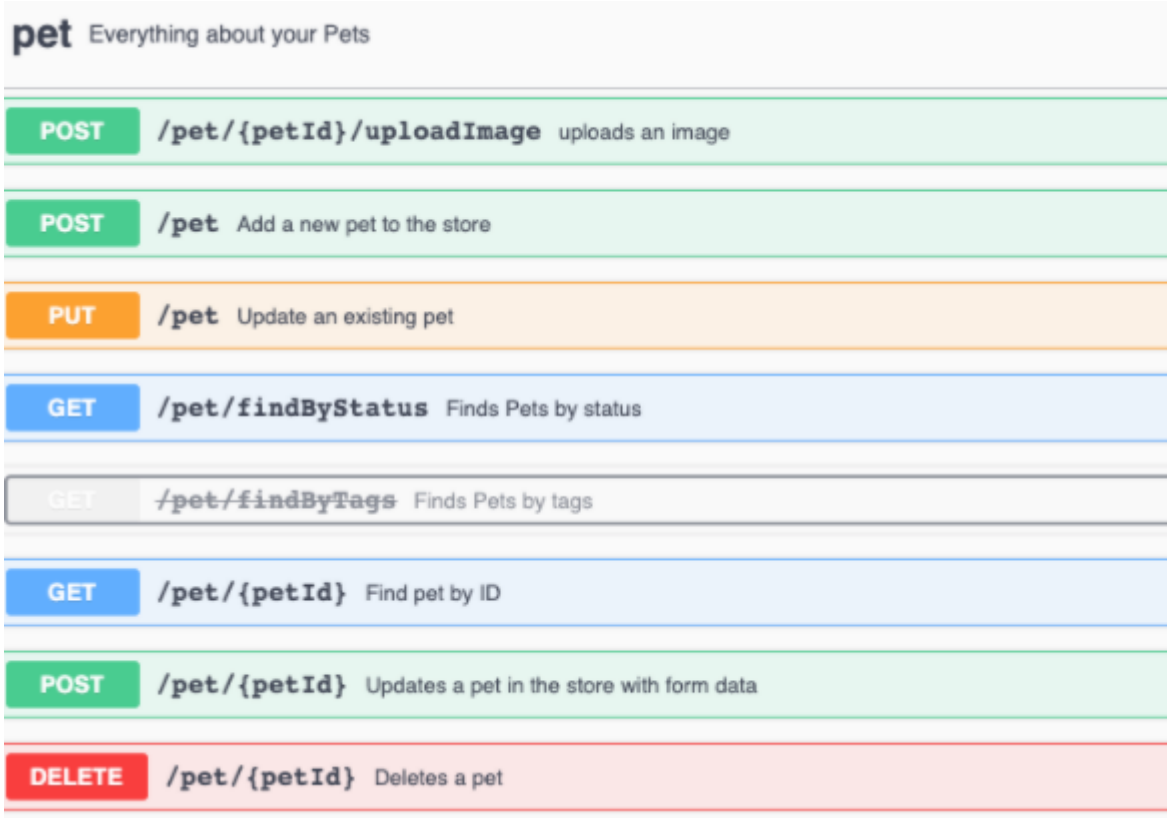


アプリケーションの参照

次の図は、データフローの解析を目的として、Contrast Scan が一般的な Web アプリケーションをどのように参照するかを示しています。



Contrast Scan は、データのエントリポイントを調べて、スキャン対象のコードを検索します。 Contrast Scan によって調査される代表的なデータエントリポイントの例は、以下のようになります。



Contrast Scan のサポート対象言語とテクノロジ

Contrast Scan は、以下の言語とテクノロジをサポートします。

テクノロジ	対応するファイル拡張子	アップロードするアーティファクト
Java バイナリスキャン		
Java(例 : J2EE、JSP、Spring MVC など)	java	JAR ファイルや WAR ファイル、または複数の JAR ファイルを含む ZIP ファイル。
ソースコードスキャン		
SAP ABAP	abap、bsp、asprog、aclass、aint、asfinc、asfugr、 appl、component	スキャンするファイルが含まれる ZIP ファイルやフォルダ。Contrast Scan では、自動的にファイルの言語が検出されます。
ActionScript	as	
ASP.NET	asax、ascx、ashx、asmx、aspx、master	
C#	cs、cshtml	
C	c、h、pc	
C++	h、hh、cpp、hpp、cc、pc	
COBOL	cob、cbl、cpy、pco	
Go	go	
HTML	htm、html、xhtml	
Informix	sql、4gl	
Java	java	
JavaScript/TypeScript	js、xsjs、ts、tsx	
JCL	jcl、prc	
JSP	jsp、jspx、xhtml	
Kotlin	kt、kts、ktm	
NATURAL	nls、nlp、nlh、nlm、nss、nsp、nsh	
Objective-C	h、m	
Oracle Forms	oforms	
PHP	php、php3、php4、php5、php6、phps、phtml	
PL-SQL	sql、sf、sps、spb、sp、fnc、spp、plsql、trg、st、prc、pks、pkb、pck	
PowerScript	sru、sra、srw、srf、srs、srm、srx	
Python	python、py	
RPG4	rpg、rpg3、rpg4、rpgle、dspf、mbr	
Scala	scala	
Swift	swift	
Transact-SQL	sql、tsql、sp	
TypeScript	js、xsjs、ts、tsx	
Visual Basic 6	bas、frm、cls	
VB.NET	bas、frm、cls	
XML	xml	

スキャンパッケージの準備
スキャンで最良の結果を得るには、スキャンするパッケージをアップロードする前に、以下のベストプラクティスを参考に準備してください。

アーティファクトの種類

- Java の場合、WAR または JAR パッケージのいずれかをアップロードしてください。
- JavaScript の場合、JS または ZIP パッケージをアップロードしてください。

1 つの ZIP パッケージに、複数の JAR ファイルを含めることができます。アップロードできる ZIP ファイルのサイズの上限は、1GB です。

クラスファイルと依存関係へのアクセス
パッケージ化されたファイルが、ここで説明しているものと異なる場合には、Contrast Scan はコードについて仮定を立てます。そのため、得られる結果が正確では無くなる可能性があります。検知漏れや誤検知になる可能性があります。

Contrast Scan がすべての必要なクラスファイルと依存関係にアクセスできれば、結果に擬似クラスは含まれることはありません。擬似クラスとは参照されるクラスですが、スキャンでそのバイトコードを検出できないか、もしくはスキャンで中間表現(IR)に逆コンパイルできないものを指します。

- Contrast Scan は、以下のファイルへのアクセスが必要です。
  - アプリケーションのクラスファイル
  - アプリケーションの依存関係にある JAR ファイルやクラスファイル
- Oracle Java™ の仕様に従って、アプリケーションおよび依存関係を WAR ファイルで構成してください。
- Spring Boot の JAR ファイルと同様に、アプリケーションおよび依存関係を JAR ファイルで構成してください。
  Spring Boot JAR ファイルは、アプリケーションと依存関係を既知の場所に配置します。
- 標準の JDK ファイルや一般的なサーブレットコンテナが提供する依存関係は含める必要はありません。Contrast Scan 側で、これらの依存関係を提供します。

フレームワーク
正確な結果を得るために、Web アプリケーションで使用されているフレームワークが Contrast Scan のサポート対象である必要があります。

- Angular 8 以降
- Jakarta EE 2.0-3.0
  - Angular 8 以降
  - J2EE
  - Jakarta EE 2.0-3.0
  - jQuery
  - React 16 以降
  - SpringBoot
  - Spring MVC
  - Vue.JS 2 以降

シン JAR ファイルの使用回避
シン JAR(thin JAR)ファイルは、アプリケーションのバイトコードのみを含むファイルです。これらのファイルには、依存関係に動的にアクセスするための特別な実行ローダーが必要です。シン JAR ファイルをアップロードした場合、アプリケーションコードのスキャンは実行されません。正確なスキャンのためのアプリケーションの依存関係へのアクセスができません。

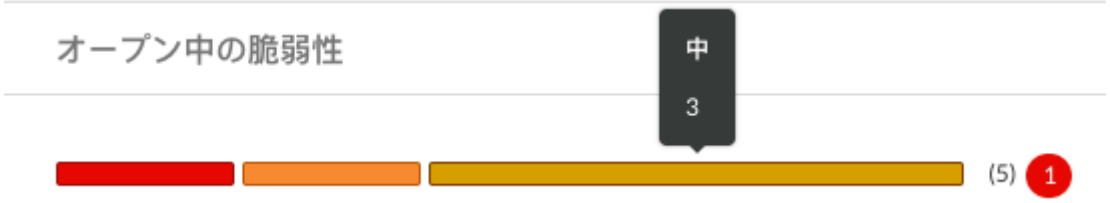
スキャンプロジェクトの表示
スキャンプロジェクトの一覧には、組織内の各スキャンプロジェクトの以下の情報が表示されます。

- **スコア**： プロジェクトの最新のスキャンに基づいて、アプリケーションの潜在的セキュリティリスクをレターグレードで表します。
  スキャンでは、Contrast ライブラリのスコア付け (610ページ)を使用して、スコアを採点します。
- **スキャンプロジェクト**： スキャンプロジェクトの名前。
- **オープン中の脆弱性**： Contrast で検出された脆弱性で未解決の脆弱性の数。
- **前回のスキャン**： 最後のスキャンが完了してからの経過時間。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. ビューにフィルタをかけるには、一覧の上部にある小さい三角形( )を選択すると、フィルタを選択できます。
   - **全て**： アーカイブされたものを除く全てのスキャンプロジェクトが表示されます。
   - **アーカイブ済**： アーカイブされたスキャンプロジェクトのみが表示されます。
3. 特定のスキャンプロジェクトを検索するには、虫眼鏡アイコン( )を選択し、検索ボックスにプロジェクト名の一部または全てを入力します。
4. ビューを並べ替えるには、ソートボックスを選択し、オプションを選択します。
   - **名前**： プロジェクト名でビューを並べ替えます。
   - **前回のスキャン**： 最後のスキャンが完了した日時に基づいて、ビューを並べ替えます。
   緑色の矢印を使用すると、昇順または降順で並べ替えることができます。

5. 特定のプロジェクトの詳細を参照するには、「スキャンプロジェクト」列でプロジェクト名を選択します。
6. 特定の深刻度の脆弱性の数を表示するには、脆弱性の棒グラフで該当部分にカーソルを合わせます。特定の深刻度の脆弱性を参照するには、棒グラフの該当部分を選択します。



スキャンプロジェクトの作成

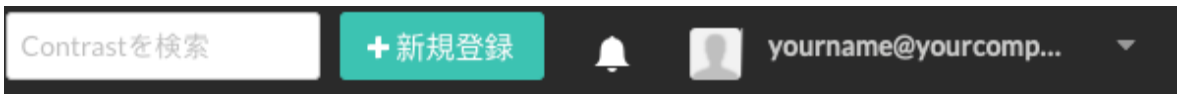
スキャンプロジェクトは、スキャンするファイルのためのコンテナです。

開始する前に

- スキャン用にアップロードするファイルを準備します。
  スキャンでは、プログラミング言語ごとに異なるファイル形式をサポートしています。例えば、Java の場合は、JAR ファイルまたは WAR ファイルをアップロードします。
- SaaS 版をご利用のお客様でロールベースのアクセス制御が有効になっている場合は、ロールに**プロジェクトの作成**アクションが含まれている必要があります。

手順

1. Contrast Web インターフェイスで、右上の**新規登録**を選択します。



2. **コード**のカードを選択します。

3. SaaS 版をご利用のお客様でロールベースのアクセス制御 (834ページ)が有効になっている場合、ドロップダウンからリソースグループを選択します。

- 1 つのリソースグループにしかアクセスできない場合は、リソースグループは選択できません。代わりにアクセス権のあるリソースグループの名前が表示されます。
- オンプレミス版をご利用のお客様、またはロールベースのアクセス制御が有効になっていない場合は、リソースグループのオプションは利用できません。

4. プロジェクトの名前を入力します。
スキャンプロジェクトの名前は 一意である必要があります。また、Contrast 内でスキャンプロジェクトを簡単に識別できるような名前を指定してください。
ファイルの名前と一致するようなプロジェクト名を付けることをお勧めします。例えば、ファイルが `webgoat.jar` であれば、プロジェクトの名前を `webgoat` や `webgoat.jar` などにします。
5. **スキャンプロジェクトを作成**を選択します。

注記
スキャンプロジェクトを作成する代わりに既存のスキャンプロジェクトを使用する場合は、**スキャンプロジェクトを表示**を選択します。この操作により、スキャンプロジェクトのタブが開き、既存のプロジェクトが表示されます。

次の手順

スキャンを開始 (560ページ)

スキャンプロジェクトの削除

スキャンプロジェクトを完全に削除したい場合、以下の手順で削除することができます。プロジェクトに関連付けられた全てのデータが完全に削除されます。

この操作は元に戻すことができません。

手順

[en]

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. 設定アイコン(⚙)を選択します。
4. 「スキャンプロジェクトの設定」の画面で、**削除**を選択します。

5. 「プロジェクトを削除」の画面で**削除**を選択して、プロジェクトを削除することを確定します。

スキャンの開始

次の様な作業を実行したい場合に、スキャンを開始します。

- 新しいアプリケーションの解析
- 以前にスキャンし、アプリケーションの脆弱性を修正したコードのテスト

開始する前に

- 再度スキャンするファイルを含むスキャンプロジェクトを選択するか、新たにプロジェクトを作成する

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. **新規スキャン**を選択します。

4. 表示された画面から、アップロードするファイルを選択し、**開く**または**アップロード**を選択します。
   スキャンプロジェクトに以前のスキャンがある場合は、以前にスキャンしたファイルの新しいバージョンを選択します。
   ファイルのアップロードが完了すると、自動的にスキャンが開始されます。

   注記
   Java バイナリスキャン： 複数の JAR ファイルを含む 1 つの ZIP ファイルをアップロードするか、単一の JAR ファイルか WAR ファイルをアップロードします。

   ソースコードスキャン： スキャンするファイルが含まれている 1 つの ZIP ファイルかフォルダをアップロードします。ZIP ファイルまたはフォルダには、異なる言語のファイルを含めることができます。Contrast Scan では、自動的にファイルの言語が検出されます。
5. スキャンの確認 (561ページ)でスキャンの進行状況を確認します。

スキャンのキャンセル

進行中のスキャンをキャンセルできます。

スキャンをキャンセルすると、スキャン履歴でのステータスが「キャンセル」に変わります。

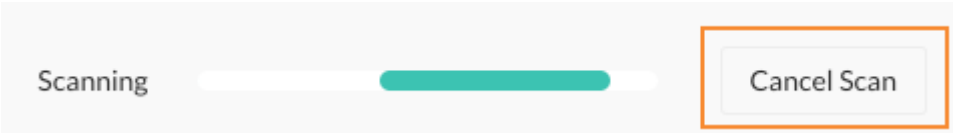
開始する前に

進行中のスキャンプロジェクトを検索します。

手順

スキャンをキャンセルするには：

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. 進行中のスキャンプロジェクトを選択します。
3. アクティビティバーの**スキャンをキャンセル**を選択します。



スキャンはすぐに停止します。

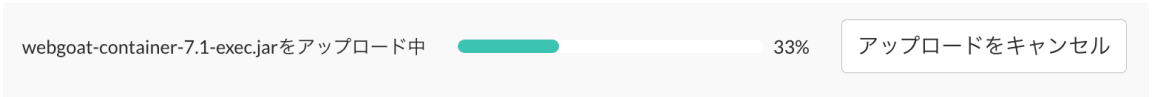
スキャンの確認

スキャンを開始した後、スキャンのどのタブでもファイルのアップロードとスキャンの進行状況を確認することができます。

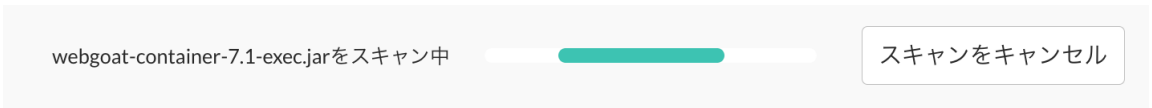
選択したプロジェクトのスキャン履歴は、概要タブに表示されます。

手順

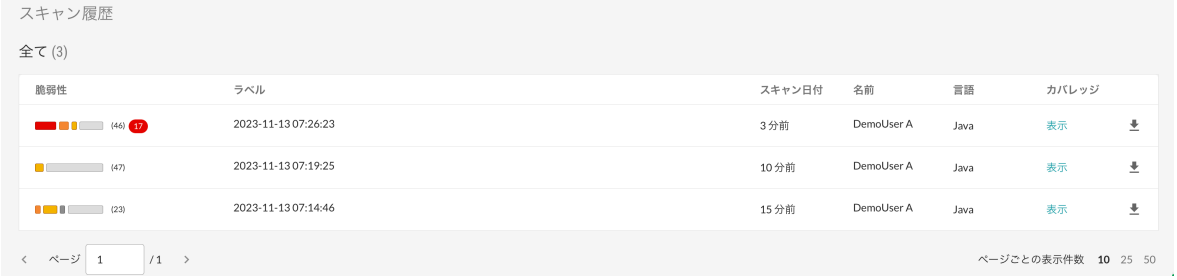
1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
   スキャンのページでは、スキャンに関する以下の情報が表示されます。
   - 各スキャンプロジェクトのスコア
   - スキャンプロジェクトの名前
   - オープン中の脆弱性の数とステータス
   - 最後のスキャンが完了した時間
2. スキャンプロジェクトを選択します。
3. スキャンの開始 (560ページ)
4. ファイルのアップロード中やスキャンの実行中は、スキャンページの上部で進行状況を確認することができます。
   - ファイルのアップロード中は、以下のようなプログレスバーが表示されます:



   - ファイルスキャン中は、以下のようなアクティビティバーが表示されます:



5. 選択したプロジェクトのスキャンの履歴を確認するには、スキャン画面で**概要タブ**を選択してスキャン履歴を表示します。



6. スキャンの詳細を表示するには、スキャンのラベル列の日時を選択するか、カバレッジ列の**表示**を選択します。

SAST コンプライアンス対応レポートの作成

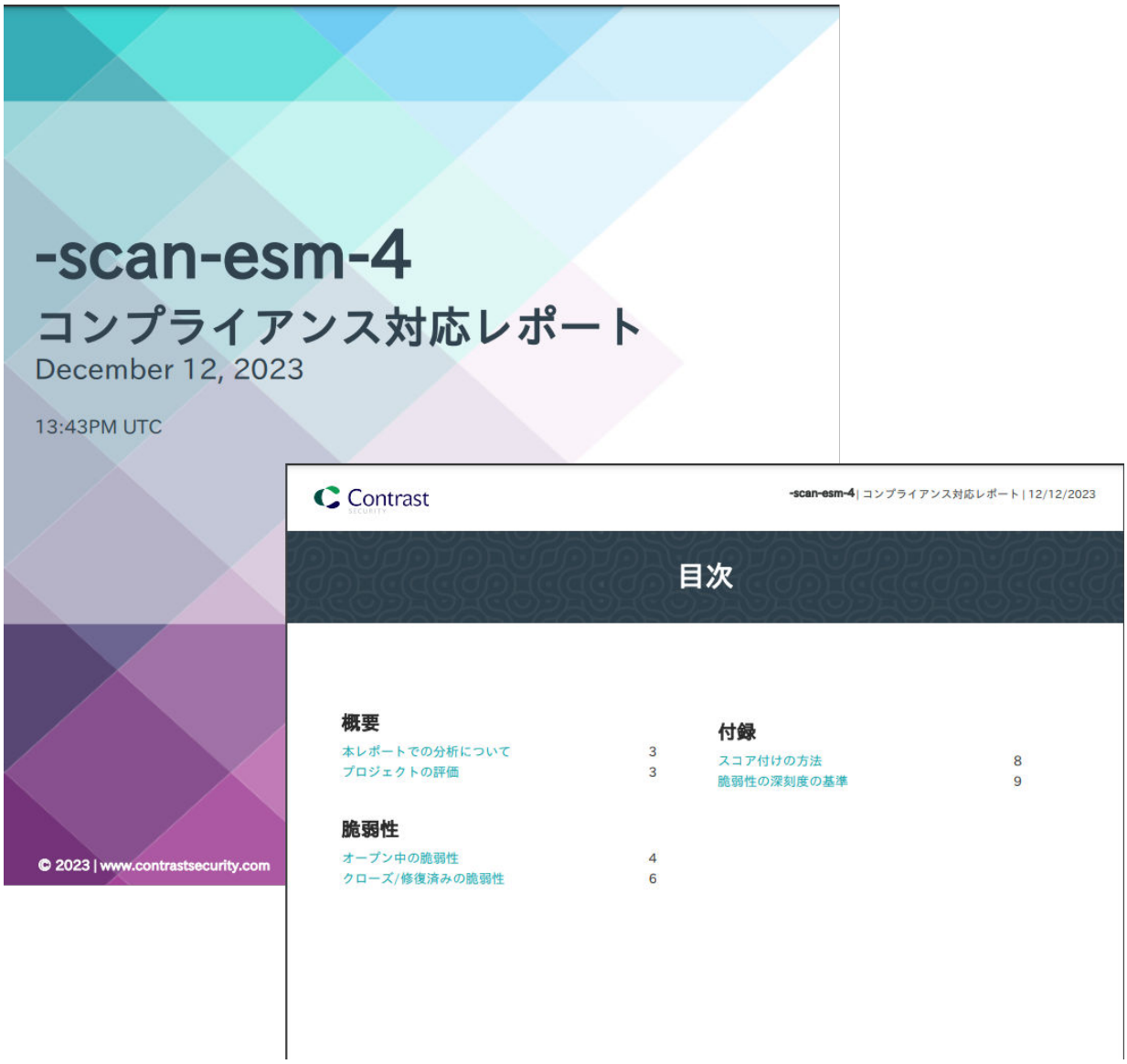
Contrast Scan の SAST コンプライアンス対応レポートは、最新のスキャン情報に基づいた脆弱性修復のエビデンスを提供します。レポートは PDF ファイル形式になります。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. レポートを作成したいスキャンプロジェクトを選択します。
3. スキャンプロジェクトページの右上にある**レポート**アイコン(📊)を選択します。
「脆弱性」タブからもレポートを作成することができます。
4. Contrast でレポートが生成されると、ダウンロードが可能になりますので、ダウンロード先を指定します。

SAST コンプライアンス対応レポートの内容

選択したプロジェクトの SAST コンプライアンス対応レポートには、プロジェクトのスコア、オープン中およびクローズ/修復済みの脆弱性に関するグラフと詳細が含まれます。



Contrast Scan ローカルエンジン

Contrast Scan ローカルエンジンを使用すると、Contrast CLI または Contrast Web インターフェイスの代わりに、Docker CLI コマンドまたは Java JAR ファイルを使用してアプリケーションをスキャンできます。スキャンが正常に完了すると、Contrast Scan ローカルエンジンによって結果が Contrast プラットフォームにアップロードされ、そこで結果を確認することができます。 アップロードされるのは以下のファイルです。

- 静的分析結果形式(SARIF)で記述されたスキャン結果の JSON ファイル
- スキャンの出力情報の LOG ファイル

この方法は、スキャンするファイルを Contrast プラットフォームにアップロードせずに、ローカルでスキャンしたい場合に便利です。

サポート対象プラットフォーム

Contrast Scan ローカルエンジンは、Linux システムと Docker コンテナでサポートされています。

ローカルスキャンでのプロキシサーバの設定

セキュリティ上の理由から、ローカルスキャンエンジンと Contrast プラットフォーム間の通信にプロキシサーバを使用している場合があります。ローカルスキャンを実行 (566ページ)する際にプロキシサーバを有効にするには、以下の環境変数を使用してください。

変数	説明
CONTRAST__API__PROXY__ENABLE	プロキシの設定を有効にします。
CONTRAST__API__PROXY__URL	**必須** プロキシサーバの URL(例、`http://host:port`)
CONTRAST__API__PROXY__TYPE	**必須** プロキシサーバの方式(例、BASIC)
CONTRAST__API__PROXY__USERNAME	**任意** プロキシサーバのユーザ名
CONTRAST__API__PROXY__PASSWORD	**任意** プロキシサーバのパスワード

スキャンの手順

Contrast Scan ローカルエンジンを使用するには：

1. ローカルスキャンの実行方法を選択します。
   - Contrast Security から、ローカルスキャナアプリケーションが含まれる Docker コンテナをダウンロード (564ページ)
   - Contrast Scan ローカルエンジンの JAR ファイルをダウンロード (564ページ)
2. 検査結果のアップロードにプロキシサーバを使用するかどうかを判断します。
3. ローカルシステムでスキャンを実行 (566ページ)します。
4. Contrast Web インタフェースで結果を確認 (569ページ)します。

Docker コンテナのダウンロード

Contrast ローカルスキャナを使用するには、Contrast の GitHub リポジトリから Docker コンテナをダウンロードします。

Docker コンテナの代わりに Java JAR ファイルを使用する場合は、JAR ファイル (564ページ)をダウンロードしてください。

開始する前に

- Docker CLI をインストールします。
- Docker に 12GB のメモリを割り当てるよう設定します。
- 個人用アクセストークン(PAT)を Contrast サポートから入手します。

手順

1. ローカルシステムで次の変数を設定します。

   ```
   export CONTRAST_PAT=<ContrastProvidedPAT>
   ```

   `<ContrastProvidedPAT>`は、Contrast サポートから入手した PAT に置き換えて下さい。
2. 次のコマンドを使用して、リポジトリにログインして Docker コンテナをダウンロードします。

   ```
   docker login ghcr.io/contrast-security-inc -u local-scanner -p \
   $CONTRAST_PAT
   docker pull ghcr.io/contrast-security-inc/contrast-sast-scanner-
   java:latest
   ```

Java JAR ファイルのダウンロード

Contrast ローカルスキャンエンジンの Java JAR ファイルをダウンロードするには、以下のいずれかを使用します。

- Maven からダウンロードする
- curl コマンドでダウンロードする

開始する前に

- 個人用アクセストークン(PAT)を Contrast サポートから入手します。

Maven からダウンロード

1. 以下の情報を含む POM(Project Object Model)ファイルを作成します。

```
<server>
  <username>local-scanner</username>
  <password>ContrastProvidedPAT</password>
  <id>github</id>
</server>

<repository>
  <id>github</id>
  <name>github</name>
  <url>https://maven.pkg.github.com/Contrast-Security-Inc/sast-local-
scan-runner</url>
</repository>

<dependency>
    <groupId>com.contrastsecurity</groupId>
    <artifactId>sast-local-scan-runner</artifactId>
    <version>latest</version>
</dependency>
```

   `ContrastProvidedPAT` は、Contrast サポートから入手した PAT に置き換えて下さい。
2. POM ファイルを右クリックして**実行(Run As) > Maven Install** を選択するか、`mvn install` コマンドを使用します。

curl コマンドでダウンロード

- Java JAR ファイルをダウンロードするには、次の curl コマンドを使用します。

```
curl -L -H 'Accept: application/vnd.github.v3.raw' -s https://
<ContrastProvidedPAT>@maven.pkg.github.com/Contrast-Security-Inc/sast-
local-scan-runner/com.contrastsecurity.sast-local-scan-runner/
<VersionNumber>/sast-local-scan-runner-<version number>.jar -o \
scanner.jar
```

  - `<ContrastProvidedPAT>`は、Contrast サポートから入手した PAT に置き換えて下さい。
  - `<VersionNumber>`を特定のバージョン番号に置き換えるか、`latest` を指定して最新バージョンをダウンロードして下さい。利用可能なバージョンについて、確認が必要であれば、Contrast の担当者までお問い合わせください。

ローカルスキャンエンジン環境変数

Contrast Scan ローカルエンジンで、以下の環境変数を使用できます。

変数	必須/任意	説明
CONTRAST__API__URL	必須	スキャン結果を報告する Contrast プラットフォームのアドレスを指定します。URL の後ろに`/api/sast` を追加してください。

変数	必須/任意	説明
CONTRAST__API__USER_NAME	必須	Contrast のユーザアカウントのユーザ名(通常、自分のログイン ID)
CONTRAST__API__API_KEY	必須	Contrast API キー
CONTRAST__API__SERVICE_KEY	必須	Contrast サービスキー
CONTRAST__API__ORGANIZATION	必須	Contrast の組織 ID
CONTRAST__API__PROXY__ENABLE	任意	プロキシの設定を有効にします。
CONTRAST__API__PROXY__URL	プロキシ設定が有効な場合は必須	プロキシサーバの URL(例、`http://host:port`)
CONTRAST__API__PROXY__TYPE	プロキシ設定が有効な場合は必須	プロキシサーバの方式(例、BASIC)
CONTRAST__API__PROXY__USERNAME	任意	プロキシサーバのユーザ名
CONTRAST__API__PROXY__PASSWORD	任意	プロキシサーバのパスワード
UMBRELLA_LAUNCHER_JAVA_PATH	任意	Umbrella Java のスキャナ jar ファイルのパス(例、`/app/contrast-scan-java-cli.jar`)

ローカルスキャンの実行

Contrast Scan ローカルエンジンは、Docker コンテナまたは Java JAR ファイルとして提供されます。スキャンを実行するために、ビルドした成果物が必要です。

開始する前に

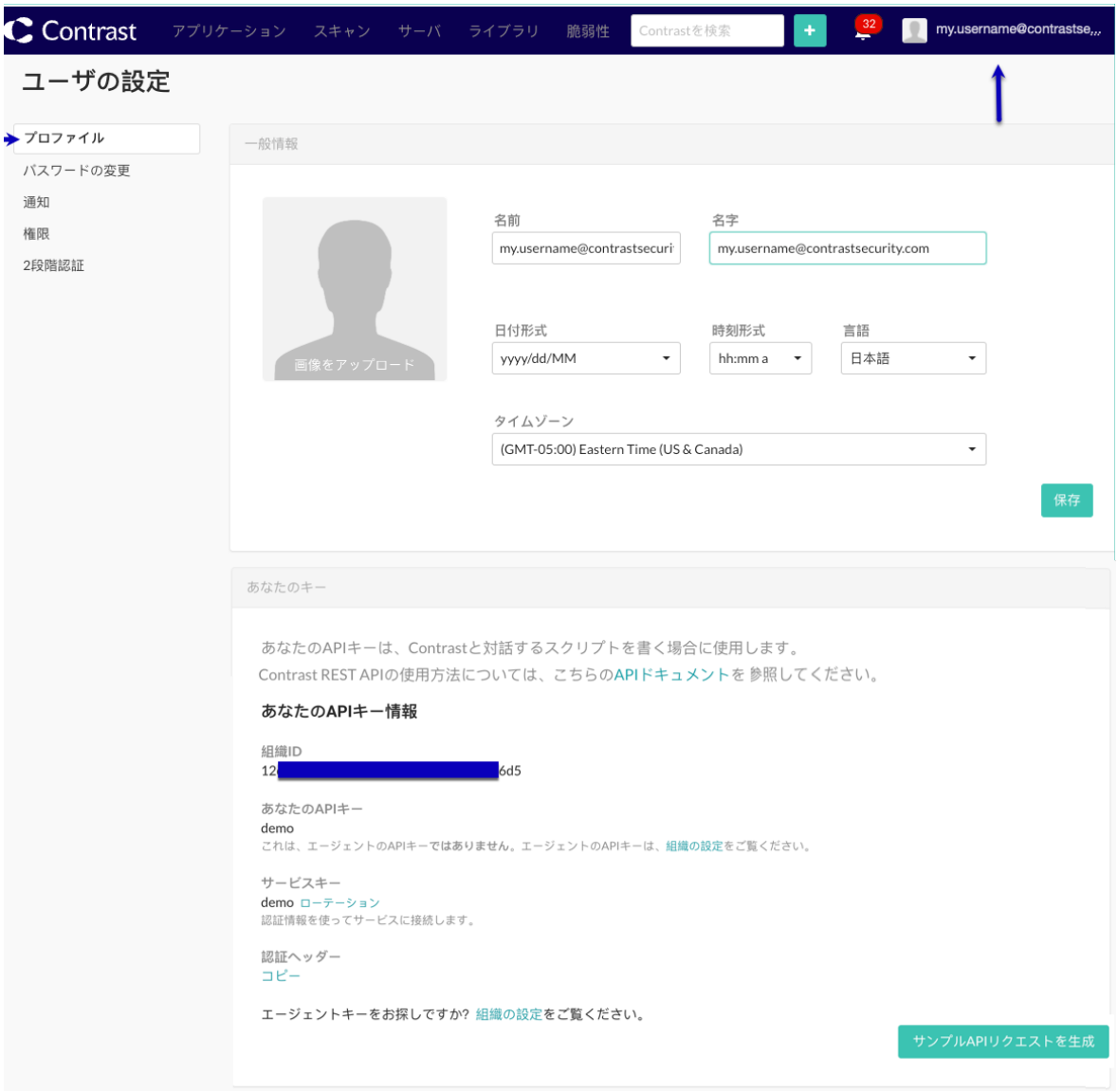
- スキャンしたいビルド済み成果物の場所を確認します。
- スキャン結果をローカルシステム上のどこに保存するかを決めます。
  出力結果のパスを指定しない場合、ローカル エンジンは結果を `results.sarif` という名前のファイルで現在の作業ディレクトリに書き込みます。
- Contrast Scan ローカルエンジンの JAR 版を使用する場合、以下のソフトウェアがシステムにインストールされていることを確認します。
  - Java 11
  - JavaScript のプロジェクトファイルをスキャンする場合、Semgrep App バージョン 0.114.0
- Contrast Scan ローカルエンジンがスキャン結果とスキャナ出力を Contrast にアップロードできるように、インターネットアクセスが必要です。
- 1CPU(Contrast Scan ローカルエンジンはシングルスレッド)、12GB の RAM があることを確認します。
- Contrast Scan ローカルエンジンを実行するディレクトリや指定の出力ディレクトリに対して、読み取りおよび書き込み権限があることを確認します。

重要
スキャンするファイルのパスにスペースが含まれていないことを確認してください。

手順

1. Contrast Web インタフェースにログインします。
2. ユーザメニューより、**ユーザの設定**を選択します 。
3. プロファイルにある、「あなたのキー」のセクションの以下の情報を取得します。
   - 組織 ID
   - あなたの API キー

- サービスキー
- Contrast URL



4. ローカルスキャナが Contrast と通信するための環境変数を設定します。

> **注記**
> ローカルスキャンエンジンと Contrast プラットフォーム間の通信にプロキシサーバを使用する場合は、プロキシサーバの環境変数 (563ページ)も指定します。

```
export CONTRAST__API__URL=<URL>
export CONTRAST__API__USER_NAME=<Username>
export CONTRAST__API__API_KEY=<APIKey>
export CONTRAST__API__SERVICE_KEY=<ServiceKey>
export CONTRAST__API__ORGANIZATION=<OrgId>
export LOCAL_ARTIFACT_LOCATION=<LocalArtifactLocation>
export LOCAL_OUTPUT_LOCATION=<LocalOutputLocation>
```

- <URL>は、スキャン結果を報告する Contrast の URL アドレスに置き換えます。 URL の後ろに `Contrast/api/sast` を追加してください。

```
export CONTRAST__API__URL=https://app.contrastsecurity.com/
Contrast/api/sast
```

- <Username> は、自分の Contrast アカウント(通常はログイン ID)に置き換えます。
- <APIKey>は、Contrast の API キーに置き換えます。
- <ServiceKey>は、Contrast のサービスキーに置き換えます。
- <OrgID>は、Contrast の組織 ID に置き換えます。
- <LocalArtifactLocation>は、Java のスキャンの場合、JAR ファイルや WAR ファイルのパスに置き換えます。JavaScript のスキャンの場合は、ソースコードが含まれるフォルダを指定します。
  オプション：変数を使用する代わりに、コマンドでパスを指定することもできます。
- <LocalOutputLocation>は、スキャン結果のファイルを保存するローカルシステムのパスに置き換えます。
  オプション：変数を使用する代わりに、コマンドでパスを指定することもできます。

5. スキャンを開始します。
   - **Docker 版**のローカルエンジンを使用する場合、次のコマンドでスキャンを開始します。

```
docker run -v $LOCAL_ARTIFACT_LOCATION:/app/artifacts \
    -v $LOCAL_OUTPUT_LOCATION:/app/results \
    -e CONTRAST__API__URL=${CONTRAST__API__URL} \
    -e CONTRAST__API__USER_NAME=${CONTRAST__API__USER_NAME} \
    -e CONTRAST__API__API_KEY=${CONTRAST__API__API_KEY}  \
    -e CONTRAST__API__SERVICE_KEY=${CONTRAST__API__SERVICE_KEY}  \
    -e CONTRAST__API__ORGANIZATION=${CONTRAST__API__ORGANIZATION} \
    ghcr.io/contrast-security-inc/contrast-sast-scanner-java:latest \
    --project-name "<ProjectName>-${CONTRAST__API__ORGANIZATION}" \
    --label "<LabelName>" \
    -r "<ResourceGroupName">\
    /app/artifacts/<FileName> \
    -o $LOCAL_TARGET_OUTPUT_LOCATION/results.sarif
```

   - <ProjectName>は、スキャンプロジェクトの名前に置き換えます。
   - <LabelName>は、スキャンのラベルに置き換えます。例えば、ビルド番号を"build:1.0.1"のように指定できます。
   - *[en] Replace <ResourceGroupName> with the name of the resource group where you want to add the project.*
   - <FileName>は、スキャンしたいファイルの名前に置き換えます。このファイルは、$LOCAL_ARTIFACT_LOCATION で指定したディレクトリに存在する必要があります。
   - **Java JAR 版**のローカルエンジンを使用する場合、次のコマンドでスキャンを開始します。

```
java -jar sast-local-scan-runner.jar <ScanArtifact> --project- \
name <ProjectName> --label <LabelName>
```

   - <ScanArtifact>は、スキャンする JAR、 WAR、または ZIP ファイルのパスに置き換えます。また、フォルダを指定することもできます。
   - <ProjectName>は、スキャンプロジェクトの名前に置き換えます。例："my project name"
   - <LabelName>は、スキャンのラベルに置き換えます。例えば、ビルド番号を"build:1.0.1"のように指定できます。

6. スキャン完了後、数分待つと、Contrast Web インタフェースに結果が表示されます。アップロードと処理時間の関係で、結果はすぐに表示されません。

コマンドのオプション

Docker コンテナまたは Java JAR ファイルで、次のコマンドオプションが使用できます。

オプション	説明
`-o, --output-results`	出力結果の保存場所を指定します。 指定しない場合、ローカルスキャナは現在の作業ディレクトリに結果を書き込みます。
`-V, --version`	バージョン情報を表示します。
`--project-name <ProjectName>`	スキャンプロジェクトの名前を指定します。 既に存在するプロジェクト名を指定した場合、ローカルエンジンはそのプロジェクトにスキャンを追加します。存在しない場合は、指定された名前の新しいプロジェクトを作成します。 プロジェクト名にスペースが含まれる場合は、二重引用符(")で囲んでください。例 : "My Scan Project" プロジェクト ID を使用しない場合、このオプションは必須です。
`--project-ID`	既存のプロジェクトの ID を指定します。 プロジェクト名を使用しない場合、このオプションは必須です。
`-r <ResourceGroup>`	プロジェクトを追加するリソースグループを指定します(オプション)。 指定したリソースグループにプロジェクトが既に存在する場合は、何も処理は行われません。
`--label <label>`	このスキャンのラベルを指定します。

終了コード

ローカルエンジンは、スキャンが完了すると以下の終了コードを返します。

終了コード	説明
0	スキャンは正常に終了し、結果を Contrast にアップロードしました。
1	入力値の検証エラーです。
2	Contrast API サーバへの接続エラーです。
3	Contrast API サーバから エラーが返されました。
4	ローカルエンジンがエラーを返しました、詳細はログファイルにあります。
5	予期せぬエラーが発生しました、詳細はログファイルにあります。

ローカルスキャン結果の確認

Contrast Web インターフェイスでローカルスキャンの詳細と結果が確認できます。

出力結果の場所を指定した場合、スキャンによってローカルシステムに作成された SARIF ファイルも見ることができます。

手順

1. Contrast Web インタフェースにログインします。
2. **スキャン**タブを選択します。
3. リストからローカルスキャンのスキャンプロジェクトを選択します。
4. ローカルスキャンの結果を表示するには、概要、脆弱性、ポリシーのタブを選択してください。
   Contrast ドキュメントには、スキャン結果の分析に関する詳細情報があります。

スキャン結果の分析

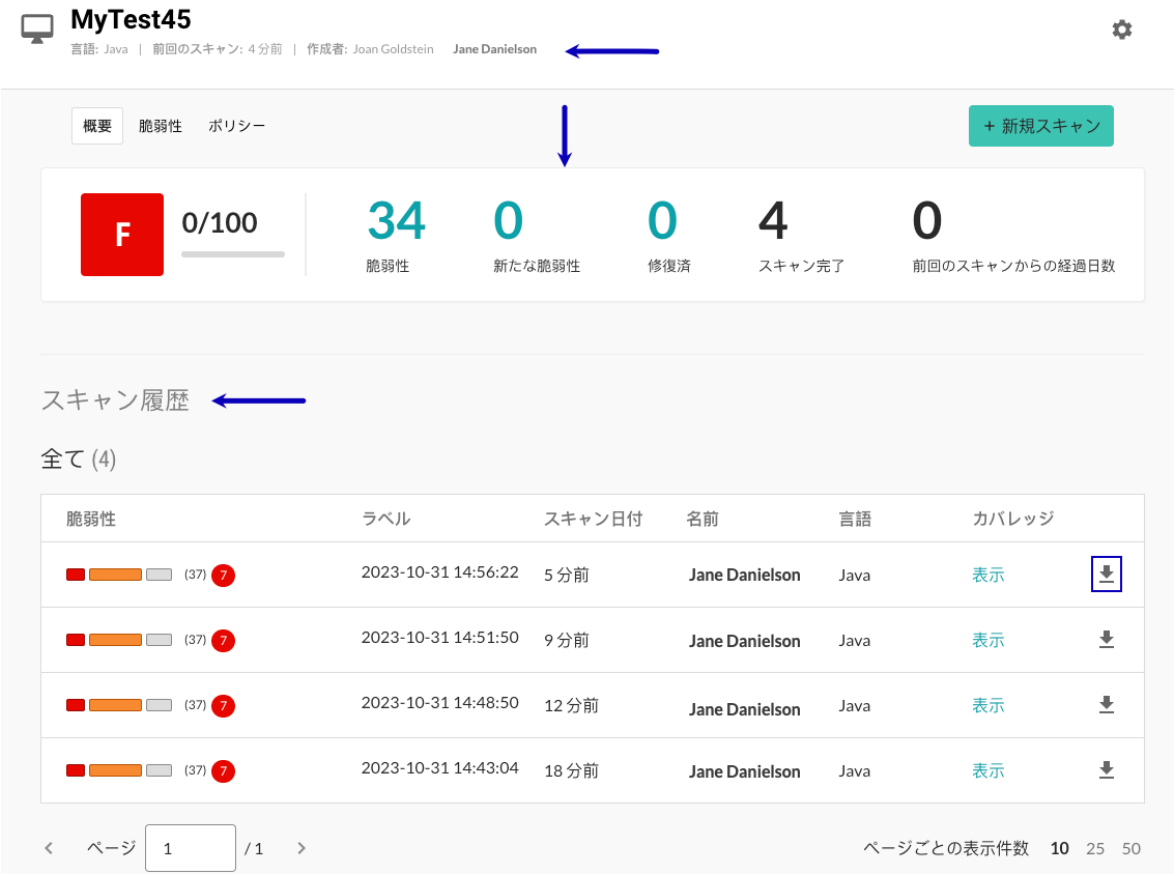
スキャンは、アプリケーション内のデータフローを観察し、検出した脆弱性を報告します。

結果を調査した後、アプリケーションのコードを修正して再度スキャンを実行し、脆弱性が修正されているか確認します。

手順

スキャンが完了したら、脆弱性の情報、プロジェクト作成者、各スキャンを実行したユーザを確認できます。

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
   スキャンページに、スキャンプロジェクトの一覧が表示されます。
2. スキャンプロジェクトを選択します。
3. 概要タブでは、スキャン結果の概要とプロジェクト内のスキャンの一覧が表示されます。



概要セクションには、次の情報が表示されます。

- **スコア**：プロジェクトの最新のスキャンに基づいて、アプリケーションの潜在的セキュリティリスクをレターグレードで表します。
  スキャンでは、Contrast ライブラリのスコア付け (610ページ)を使用して、スコアを採点します。
- **脆弱性**：最新のスキャンで検出された脆弱性の数。検出された脆弱性の詳細を表示するには、数字を選択します。
- **新たな脆弱性**：最新のスキャンで検出された新たな脆弱性の数。この値には、以前のスキャンで検出され、修正されていない脆弱性は含まれません。
  例：
  「スキャン 1」で 3 件の脆弱性が検出された場合：
  - 脆弱性の数は 3 です。
  - 新たな脆弱性の数も 3 です。

  アプリケーションのコード変更により新たな脆弱性が 1 件発生し、既存の脆弱性が修正されていない場合に「スキャン 2」を実行した場合：
  - 脆弱性の数(検出された全ての脆弱性)は、4 になります。
  - 新たな脆弱性の数(スキャン 2 で新たに検出されたもの)は、1 になります。

  新たな脆弱性の詳細を表示するには、数字を選択します。
- **修復済**：アプリケーションのソースコードや設定ファイルを変更することによって修正された脆弱性の数。
  修復済の脆弱性の詳細を表示するには、数字を選択します。
- **スキャン完了**：プロジェクトで完了したスキャンの数。
  完了したスキャンの詳細を表示するには、数字を選択します。

- **前回のスキャンからの経過日数：** 最後のスキャンが完了してからの日数。

このタブで、スキャンの行の末尾にある**ダウンロード**アイコン( )を選択すると、結果が SARIF ファイルにエクスポートされます。

4. 脆弱性タブで、脆弱性をステータスや深刻度でソートするには、ステータスまたは深刻度の列の横にある**フィルタ**アイコン( )を選択し、1 つまたは複数の項目を選択します。

深刻度のフィルタ：

ステータスのフィルタ：

フィルタをクリアするには、深刻度またはステータス列の横にある**クリア**を選択します。

5. 脆弱性に関するその他の詳細情報は、**備考**タブを選択すると、次のような情報が表示されます。
   - 脆弱性の検出時間

   a. 脆弱性の詳細やアプリケーションコード内での脆弱性の場所を参照するには、**詳細**タブを選択すると次の情報が表示されます。
      - 脆弱性が存在するメソッド
      - スキャンによって脆弱性が検出されたファイル
      - スキャンによって脆弱性が検出されたアプリケーションコードの最初の行

   b. コードの修正方法を参照するには、**修正方法**タブを選択します。

   c. 脆弱性に関するその他の詳細情報は、**備考**タブを選択すると、次のような情報が表示されます。
      - 脆弱性の検出時間
      - 脆弱性が検出されたコードモジュール
      - 脆弱性の種類(インジェクションなど)
      - 深刻度

- リスクの信頼性
- 脆弱性に適用されるセキュリティ基準

d. 脆弱性に関するアクティビティを参照したい場合、**アクティビティ**タブを選択すると、次のような情報が表示されます。
   - 変更を行ったユーザ
   - 脆弱性のステータスの変更
   - コメント

スキャン情報の表示

スキャン情報には、以下が含まれます。

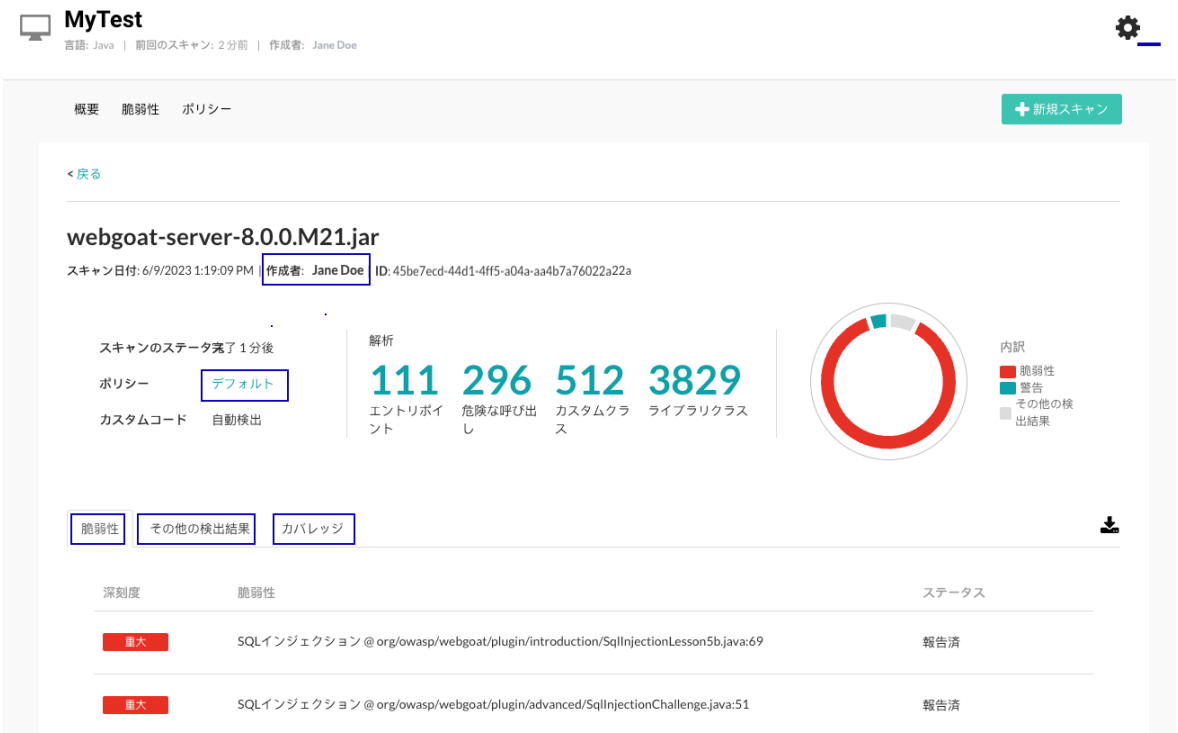
- スキャン結果の概要
- スキャンのカバレッジ詳細

開始する前に

- 情報を表示したいスキャン(スキャンプロジェクト)を決めます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. 「スキャン履歴」で、参照するスキャンの「ラベル」列のリンクを選択するか、「カバレッジ」列で**表示**を選択します。
4. スキャンの詳細が表示されます。



- 一覧の上部に、スキャンの詳細の概要が表示されます。
  概要の上部で、スキャンを実行したユーザ名を確認できます。

> **注記**
> スキャン実行者を表示する機能は、2023 年 6 月 12 日以降に作成されたスキャンプロジェクトで利用できます。

スキャンで使用されたルールを表示するには、「ポリシー」の横の**デフォルト**を選択します。
- 「脆弱性」タブに、スキャンで検出された脆弱性が表示されます。
  この一覧にある脆弱性は、対策が必要であると確信されるものです。
  脆弱性のデータをファイルにダウンロードするには、脆弱性の一覧の上部にある**ダウンロード**アイコン(⬇)を選択します。
- 「その他の検出結果」タブに、スキャンで検出されたその他の脆弱性が表示されます。
  これらの脆弱性の報告時にスキャンで行われた仮定により、この一覧にある脆弱性への対策の必要性はより低くなります。
- 「カバレッジ」タブに、スキャンに含まれたクラスとスキャンの対象外となったクラスが表示されます。

スキャンの脆弱性ステータスの編集

Contrast でスキャン中に脆弱性が検出されると、脆弱性に**報告済**のステータスが割り当てられます。このステータスは、脆弱性が悪用される可能性があることを示します。

このステータスは、脆弱性の管理方法に応じて、次のいずれかの値に変更できます。

- **確認済**：ソースコードをレビューする、脆弱性を悪用してみることなどで、この脆弱性が真の判定であると確認した場合のステータスです。
- **疑わしい**：脆弱性は、Contrast から提供された情報に基づくと真の判定のように見えますが、その有効性を判断するにはさらに調査が必要な場合のステータスです。
- **問題無し**：この脆弱性には、コードの変更を必要としないと判断された場合のステータスです。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. **脆弱性**タブを選択します。
4. ステータスを変更：
   a. 脆弱性のページで、ステータス列にあるステータスを選択します。

または、脆弱性の一覧から脆弱性を選択し、ビューの右側にあるステータスを選択します。

b. 必要に応じて、ステータス変更の理由を入力できます。理由の説明をコメントとして入力して**上書き**を選択します。

5. ステータスを変更せずに脆弱性にコメントを追加：

a. 脆弱性タブで、脆弱性を選択します。
b. **アクティビティ**タブを選択します。
c. コメントを入力して、**コメントを追加**を選択します。

スキャン結果のダウンロード

スキャンが完了したら、結果を SARIF(Static Analysis Results Interchange Format)ファイルでダウンロードできます。これは静的解析データを出力するための標準的な JSON 形式のファイルです。

ストレージの使用量を最適化するため、ダウンロードが可能なのは、スキャン完了後 5 日間です。それより古いスキャンはダウンロードできません。

開始する前に

- 結果をダウンロードするスキャンを決めます

手順

1. Contrast Web インターフェイスのナビゲーションバーでスキャンを選択します。
2. スキャンプロジェクトを選択します。
3. スキャン履歴からスキャン結果をダウンロードするには、該当スキャンの行の最後にカーソルを合わせ、ダウンロードアイコン( )を選択します。
4. スキャンの詳細ページからスキャン結果をダウンロードするには：
   a. スキャン履歴でスキャンを選択するか、カバレッジ列の**表示**を選択します。
   b. 概要タブまたは脆弱性タブで、ダウンロードアイコン( )を選択します。

SARIF ファイルのデータ

スキャンの結果は、SARIF ファイルでダウンロードできます。

SARIF は、静的解析結果を記述する標準的なデータモデルでシリアライズ化された出力形式です。SARIF ファイルのデータを理解することで、スキャンの結果をより深く考察できます。

SARIF ファイルには、次のような情報が含まれます。

- Contrast が使用するスキャナに関する情報
- スキャン対象とスキャン構成に関するデータ
- 脆弱性の検出結果に関するデータ
- スキャン中に適切に処理された エラーや通知内容
- スキャンのカバレッジ情報

スキャナの情報

Contrast で使用したスキャナに関する情報は、以下の例のようになります。

```
  "tool" : {
    "driver" : {
      "name" : "Contrast Scan",
      "organization" : "Contrast Security, Inc.",
      "version" : "pkg: 2.0.0-SNAPSHOT, engine: 2.0.0-SNAPSHOT, policy: 2.0.0-SNAPSHOT",
      "informationUri" : "https://www.contrastsecurity.com"
    }
```

脆弱性の情報

ここでは、1 つの脆弱性のデータを例として使用します。スキャンによる脆弱性の検出結果は、`results` セクションにあります。

- **オブジェクト**
  以下は、SQL インジェクションの脆弱性についてのスキャン結果の例です。`threadFlows` セクションにあるオブジェクトは、ソースからシンクまでのスキャン情報を示します。

```
  "artifacts" : [ ],
  "results" : [ {
    "ruleId" : "sql-injection",
    "level" : "error",
    "message" : {
      "text" : "sql-injection in User.fetch() reachable from LoginController.login()"
    },
    "locations" : [ {
      "physicalLocation" : {
        "artifactLocation" : {
          "uri" : "file:///github/workspace/src/main/java/com/scalesec/vulnado/User.java"
        },
        "region" : {
          "startLine" : 49,
          "startColumn" : 1,
          "snippet" : {
            "text" : "public static User fetch(String un) {\n  ...\n  rs = stmt.executeQuery(query);    // Java line
            49\n  ...\n}"
          }
        },
        "contextRegion" : {
          "snippet" : { }
        }
      }
    } ],
    "partialFingerprints" : {
      "GITHUB_SOURCECODE_LSH/v1" :
      "e86a7a0c38715f4a:1-158823731d54d12e:1-6b692237789f9a2a:1-64ed8e6526b79cf4:1-13db8c734146ff90:1"
    },
    "codeFlows" : [ {
      "message" : {
        "text" : "Untrusted data flow from LoginController.java:19 to User.java:49 via variable `query`"
      },
      "threadFlows" : [ {
        "locations" : [ {
          "location" : {
            "physicalLocation" : {
              "artifactLocation" : {
                "uri" : "com/scalesec/vulnado/LoginController.java"
              },
              "region" : {
                "startLine" : 19,
                "snippet" : {
                  "rendered" : {
                    "text" : "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"}, method={\"POST\"},
                    produces={\"application/json\"}, consumes={\"application/json\"})\nLoginResponse login(**@RequestBody
                    LoginRequest input**) {\n  ...\n}"
                  }
                },
                "properties" : {
                  "ir" : [ "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"}, method={\"POST\"},
                  produces={\"application/json\"}, consumes={\"application/json\"})\nLoginResponse login(**@RequestBody
                  LoginRequest input**) {", "  ...", "}" ]
                }
```

- **シンクの場所**
  脆弱性の問題がある場所やシンクの例です。

```
"locations" : [ {
  "physicalLocation" : {
    "artifactLocation" : {
      "uri" : "file:///github/workspace/src/main/java/com/scalesec/vulnado/User.java"
    },
    "region" : {
      "startLine" : 49,
      "startColumn" : 1,
      "snippet" : {
        "text" : "public static User fetch(String un) {\n  ...\n  rs = stmt.executeQuery(query);    // Java line
        49\n  ...\n}"
      }
    },
    "contextRegion" : {
      "snippet" : { }
    }
  }
```

- **スレッドフローのステップ**
  データフローの実行ステップの例です。

```
"threadFlows" : [ {
  "locations" : [ {
    "location" : {
      "physicalLocation" : {
        "artifactLocation" : {
          "uri" : "com/scalesec/vulnado/LoginController.java"
        },
        "region" : {
          "startLine" : 19,
          "snippet" : {
            "rendered" : {
              "text" : "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"}, method={\"POST\"},
              produces={\"application/json\"}, consumes={\"application/json\"})\nLoginResponse login(**@RequestBody
              LoginRequest input**) {\n  ...\n}"
            }
          },
          "properties" : {
            "ir" : [ "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"}, method={\"POST\"},
            produces={\"application/json\"}, consumes={\"application/json\"})\nLoginResponse login(**@RequestBody
            LoginRequest input**) {", "  ...", "}" ]
          }
        }
      },
      "logicalLocations" : [ {
        "name" : "LoginController.login()",
        "fullyQualifiedName" : "com.scalesec.vulnado.LoginController.login(com.scalesec.vulnado.LoginRequest)"
      } ]
    }
  }, {
    "location" : {
      "physicalLocation" : {
        "artifactLocation" : {
          "uri" : "com/scalesec/vulnado/LoginController.java"
        },
        "region" : {
          "startLine" : 20,
          "snippet" : {
            "rendered" : {
              "text" : "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"}, method={\"POST\"},
              produces={\"application/json\"}, consumes={\"application/json\"})\nLoginResponse login(@RequestBody
              LoginRequest input) {\n  ...\n  $stack0 = input.username;    // Java line 20\n  user = User.fetch
              ($stack0);    // Java line 20\n  ...\n}"
            }
          },
          "properties" : {
            "ir" : [ "$stack0 = input.<com.scalesec.vulnado.LoginRequest:java.lang.String username>;    // Java
            line 20", "user = staticinvoke <com.scalesec.vulnado.User: com.scalesec.vulnado.User fetch(java.lang.
            String)>($stack0);    // Java line 20" ]
          }
        }
      },
      "logicalLocations" : [ {
        "name" : "LoginController.login()",
        "fullyQualifiedName" : "com.scalesec.vulnado.LoginController.login(com.scalesec.vulnado.LoginRequest)"
      } ],
      "message" : {
        "text" : "un"
      }
    },
    "state" : {
      "un" : {
        "text" : "tainted",
        "properties" : {
          "taintTags" : [ "untrusted", "cross-site" ]
        }
```


- **脆弱性の物理的および論理的な場所**
  ここでの例は、脆弱性がある物理的および論理的な場所を示しています。
  このセクションには、中間表現(IR)データのコードスニペットやレンダリングデータを含めた実行ステップの情報があります(通常はユーザコードは表示されません)。Contrast はこのデータを使用して、スキャンで確認する内容を解析します。
  1. 中間表現(IR)の実行ステートメントの例です。
  2. アプリケーションで実行ステップが発生する場所を示す例です。

```
44          "threadFlows" : [ {
45            "locations" : [ {
46              "location" : {
47                "physicalLocation" : {
48                  "artifactLocation" : {
49                    "uri" : "com/scalesec/vulnado/LoginController.java"
50                  },
51                  "region" : {
52                    "startLine" : 19,
53                    "snippet" : {
54                      "rendered" : {
55                        "text" : "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"},
                          method={\"POST\"}, produces={\"application/json\"}, consumes={\"application/
                          json\"})\nLoginResponse login(**@RequestBody LoginRequest input**) {\n  ...\n}"
56                      }
57                    },
58                    "properties" : {
59                      "ir" : [ "@CrossOrigin(origins={\"*\"})\n@RequestMapping(value={\"/login\"},
                        method={\"POST\"}, produces={\"application/json\"}, consumes={\"application/json\"}
                        )\nLoginResponse login(**@RequestBody LoginRequest input**) {", "  ...", "}" ]
60                    }
61                  }
62                },
63                "logicalLocations" : [ {
64                  "name" : "LoginController.login()",
65                  "fullyQualifiedName" : "com.scalesec.vulnado.LoginController.login(com.scalesec.
                    vulnado.LoginRequest)"
66                } ]
67              }
```

- **信頼できないデータの所在**
  ここでの例は、スキャンで検出された信頼できないデータを示しています。
  スキャンは、コードソースからデータシンクまでの実行ステップを調べます。スキャン結果には、信頼できないデータに触れる全ての実行ステップが含まれます。
  `importance`(重要性)の結果が `essential`(必要)な場合は、このコードの部分に脆弱性がないかを確認する必要があります。
  1. スキャンが追跡する汚染データの場所を示す例です。
  2. 追跡対象のデータが実行ステップで扱われる場所を示す例です。スキャン結果は、`importance`(重要性)が `essential`(必要)であることが分かります。このコードでは脆弱性を確認する必要があります。

```
"location" : {
  "physicalLocation" : {
    "artifactLocation" : {
      "uri" : "com/scalesec/vulnado/User.java"
    },
    "region" : {
      "startLine" : 47,
      "snippet" : {
        "rendered" : {
          "text" : "public static User fetch(String un) {\n  ...\n  $stack0 = new
          StringBuilder();    // Java line 47\n  $stack1 = \"select * from users where
          username = '\";    // Java line 47\n  $stack0 = $stack0.append($stack1);    //
          Java line 47\n  $stack0 = $stack0.append(un);    // Java line 47\n  $stack1 =
          \"' limit 1\";    // Java line 47\n  $stack0 = $stack0.append($stack1);    //
          Java line 47\n  query = $stack0.toString();    // Java line 47\n  ...\n}"
        }
      },
      "message" : {
        "text" : "Propagation event of untrusted data occurred at 'User.java:47' in the
        method 'User.fetch()' to variable 'query'"
      },
      "properties" : {
        "ir" : [ "$stack0 = new java.lang.StringBuilder;    // Java line 47", "$stack1 =
        \"select * from users where username = '\";    // Java line 47", "$stack0 =
        virtualinvoke $stack0.<java.lang.StringBuilder: java.lang.StringBuilder append
        (java.lang.String)>($stack1);    // Java line 47", "$stack0 = virtualinvoke
        $stack0.<java.lang.StringBuilder: java.lang.StringBuilder append(java.lang.String)>
        (un);    // Java line 47", "$stack1 = \"' limit 1\";    // Java line 47", "$stack0
        = virtualinvoke $stack0.<java.lang.StringBuilder: java.lang.StringBuilder append
        (java.lang.String)>($stack1);    // Java line 47", "query = virtualinvoke $stack0.
        <java.lang.StringBuilder: java.lang.String toString()>();    // Java line 47" ]
      }
    }
  },
  "logicalLocations" : [ {
    "name" : "User.fetch()",
    "fullyQualifiedName" : "com.scalesec.vulnado.User.fetch(java.lang.String)"
  } ],
  "message" : {
    "text" : "query"
  }
},
"state" : {
  "query" : {
    "text" : "tainted",
    "properties" : {
      "taintTags" : [ "untrusted", "cross-site" ]
    }
  }
},
"importance" : "essential"
```


スキャン結果の分析

ここでは、SARIF ファイルの内容でスキャン結果の分析に役立つ情報について、いくつか例を用いて説明します。

- **データの追跡に使用されたクラス**
  これらのクラスは、スキャンの実行時間に影響します。

```
"scannedData" : {
  "scannedBodyClasses" : [ "com.scalesec.vulnado.ServerError",
    "com.scalesec.vulnado.LoginResponse",
    "org.springframework.lang.UsesSunMisc",
    "com.scalesec.vulnado.Postgres",
    "com.scalesec.vulnado.LinksController",
    "com.scalesec.vulnado.BadRequest",
    "com.scalesec.vulnado.Unauthorized",
    "com.scalesec.vulnado.Comment",
    "org.springframework.lang.NonNullFields",
    "org.springframework.lang.NonNull",
    "org.springframework.lang.UsesJava7",
    "org.springframework.lang.UsesJava8",
    "com.scalesec.vulnado.CowController",
    "com.scalesec.vulnado.LoginController",
    "org.springframework.lang.NonNullApi",
    "org.springframework.lang.Nullable",
    "org.springframework.lang.UsesSunHttpServer",
    "com.scalesec.vulnado.CommentsController",
    "com.scalesec.vulnado.Cowsay",
    "com.scalesec.vulnado.VulnadoApplication",
    "com.scalesec.vulnado.LoginRequest",
    "com.scalesec.vulnado.LinkLister",
    "com.scalesec.vulnado.User",
    "java.util.Optional",
    "com.scalesec.vulnado.CommentRequest" ],
```


- **型階層の解決に使用したクラス**
  ここに表示されるクラスは、スキャンで型の階層を解決するためだけに使用されています。
  ライブラリのクラスは、セキュリティの問題に関連していないか、または関連する API に対する特定のポリシーが Contrast にあります。
  このセクションに表示されるクラスがカスタムコードに関連している場合、スキャン結果に検知漏れや誤検知の可能性があります。

```
2134        "nonScannedBodyClasses" : [
2135            "java.nio.file.WatchEvent$Modifier",
2136            "java.awt.Color", "java.awt.peer.WindowPeer",
2137            "java.util.function.IntUnaryOperator",
2138            "sun.awt.datatransfer.DataTransferer",
2139            "java.awt.JobAttributes$MultipleDocumentHandlingType",
2140            "java.lang.Integer",
2141            "java.awt.image.SampleModel",
2142            "javax.swing.border.Border",
2143            "java.awt.peer.ScrollbarPeer",
2144            "java.util.Vector",
2145            "java.sql.DriverAction",
2146            "java.sql.SQLType",
2147            "org.springframework.core.ParameterizedTypeReference$1",
2148            "java.nio.file.Path",
2149            "java.nio.channels.Channel",
2150            "org.springframework.core.io.Resource",
2151            "java.awt.peer.ContainerPeer",
2152            "javax.swing.KeyStroke",
2153             "java.lang.CharSequence",
2154             "java.awt.dnd.DropTargetDragEvent",
2155             "java.time.temporal.TemporalField",
2156             "org.springframework.beans.factory.InjectionPoint",
2157             "java.util.logging.ErrorManager",
2158             "java.awt.Scrollbar",
2159             "java.io.Serializable",
2160             "java.util.concurrent.CompletionStage",
2161             "java.lang.LayerInstantiationException",
2162             "org.jsoup.parser.Token$EndTag",
2163             "java.lang.invoke.BoundMethodHandle$Specializer$Factory",
2164              "java.lang.invoke.MemberName",
2165              "java.util.function.ToDoubleFunction",
2166              "java.io.ObjectInputStream$GetField",
```

- **擬似クラス**
  擬似(phantom)クラスとは、参照されるクラスですが、スキャンでそのバイトコードを検出できないか、もしくはスキャンでコードを中間表現(IR)に逆コンパイルできないものを指します。
  スキャン結果に擬似クラスが含まれないことが理想的です。スキャン結果に擬似クラスが含まれる場合、より正確な結果を提供するための情報をスキャンで検出できなかったことを意味します。このセクションにアプリケーションコードまたはライブラリが表示されている場合は、コードを調べて問題が存在するかどうかを確認してください。

```
2400        "phantomClasses" : [ "BOOT-INF.classes.com.scalesec.vulnado.LoginController",
2401             "javax.annotation.Nonnull",
2402             "groovy.lang.Closure",
2403             "javax.annotation.meta.TypeQualifierDefault",
```

- **信頼できないデータのルート検出**
  ここでの例は、信頼できないデータがアプリケーションに入るルートが検出されたことを示しています。
  スキャンは、このセクションに表示されている関数からのデータフローの動きのみを調べます。脆弱な暗号化など、データフローに関連する他の機能については解析されません。

```
      "routesDiscovered" : [ {
        "routeSignature" : "com.scalesec.vulnado.LoginController.login(com.scalesec.vulnado.
        LoginRequest)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.CowController.cowsay(java.lang.String)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.CowController.cowsay2(java.lang.String)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.LinksController.links(java.lang.String)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.LinksController.linksV2(java.lang.String)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.CommentsController.comments(java.lang.String)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.CommentsController.createComment(java.lang.String,com.
        scalesec.vulnado.CommentRequest)"
      }, {
        "routeSignature" : "com.scalesec.vulnado.CommentsController.deleteComment(java.lang.String,
        java.lang.String)"
      } ]
    },
    "invocations" : [ {
      "commandLine" : "java -XX:MaxRAMPercentage=80 -jar /app/contrast-scan-java-cli.jar
      --prescan-metadata /tmp/
      cb9757b4-4fdf-4de9-bdf1-7769058307eb_e1a6403d-be7c-46ee-82aa-6b7e47ce97d2_metadata.json -o /tmp/
      results.sarif.json /tmp/
      cb9757b4-4fdf-4de9-bdf1-7769058307eb_e1a6403d-be7c-46ee-82aa-6b7e47ce97d2_vulnado-0.0.1-SNAPSHOT.
      jar",
      "toolExecutionNotifications" : [ ],
      "executionSuccessful" : true
```


スキャンポリシーの表示

アプリケーションコード内のどの脆弱性を Contrast で検査するかは、スキャンのポリシーによって決まります。 現時点では、ポリシーの編集や追加はサポートされていません。

注記

ポリシーの表示は、Java バイナリスキャンのみに適用されます。

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. **ポリシー**を選択します。
   ポリシーの一覧に、Java バイナリスキャンのポリシーが表示されます。
4. ポリシータブでは、検索ボックスに 1 文字以上の文字を入力して特定のポリシーを検索できます。

スキャン設定の変更

スキャン設定では、スキャンプロジェクトの名前を変更することができます。

開始する前に

- 管理者権限(Admin ロール)が必要です。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。

3. 画面の右上にある**設定**アイコン(⚙)を選択します。
4. プロジェクトの新しい名前を入力します。
5. **保存**を選択します。

スキャンプロジェクトのアーカイブ

特定のスキャンプロジェクトをスキャンプロジェクトの一覧から外したい場合(例えば、そのプロジェクトを使用しなくなった場合など)、スキャンプロジェクトをアーカイブすることができます。

アーカイブされたプロジェクトに関連するデータは、Contrast で保持されます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンプロジェクトを選択します。
3. 設定アイコン(⚙)を選択します。
4. 「スキャンプロジェクトの設定」の画面で、**アーカイブ**を選択します。

5. 「プロジェクトをアーカイブ」の画面で、**アーカイブ**を選択します。
   フィルタで「アーカイブ済」を選択しない限り、アーカイブされたプロジェクトは一覧に表示されなくなります。

スキャンプロジェクトのアーカイブ解除

アーカイブされたスキャンプロジェクトの情報を表示・使用するには、アーカイブを解除します。

開始する前に

- 組織の Admin ロールが必要です。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**スキャン**を選択します。
2. スキャンの一覧の上部にある小さい三角形(˅)を選択し、**アーカイブ済**を選択して、アーカイブされたプロジェクトを表示します。
3. プロジェクトの行の最後にカーソルを合わせ、**アーカイブを解除**のアイコン(⏏)を選択します。
4. 「プロジェクトのアーカイブを解除」の画面で、**アーカイブを解除**を選択します。

ビルドパイプラインでのインテグレーション

Contrast CLI (674ページ)には、Contrast Web インターフェイスを使用せずにスキャンを実行できるコマンドがあります。

本項では、Contrast CLI を使用してビルドパイプラインにスキャンを組み込む手順について説明します。

Contrast Maven プラグイン (772ページ)を使用して、プロジェクトの Maven ビルドに Contrast Scan を組み込むこともできます。

開始する前に

- Contrast Web インターフェイスのユーザメニューより**ユーザの設定 > プロファイル**を選択し、以下の情報を取得してください。
  - API キー
  - 組織 ID
  - Contrast URL
  - 認証ヘッダー
- WAR ファイルまたは JAR ファイルがアクセス可能な場所にあることを確認してください。

手順

1. お使いのビルドパイプラインのワークフローに、最新バージョンの Contrast CLI をダウンロードするコマンドを追加します。

   ```
   npm install --location=global @contrast/contrast@2
   ```
2. API キー、組織 ID、Contrast URL、認証ヘッダーの環境変数を設定します。
   以下は、GitHub のシークレットを使用して環境変数を設定する例です。ご利用の環境に適した方法をご使用ください。

   ```
   CT_API_KEY: ${{ secrets.CONTRAST__API__API_KEY }}
   CT_AUTH_TOKEN: ${{ secrets.CONTRAST__API__AUTH_TOKEN }}
   ORG_ID: ${{ secrets.CONTRAST__API__ORGANIZATION_ID }}
   URL: ${{ secrets.CONTRAST__API__URL }}
   ```
3. スキャンを開始するために、以下のようなコマンドを追加します。

   ```
   contrast --scan ../scan-cli-testing/java/apps/param.war \
   --api_key $CT_API_KEY \
   --authorization $CT_AUTH_TOKEN \
   --organization_id $ORG_ID \
   --host $URL \
   --project_name MY-Project \
   --language JAVA --wait_for_scan
   ```

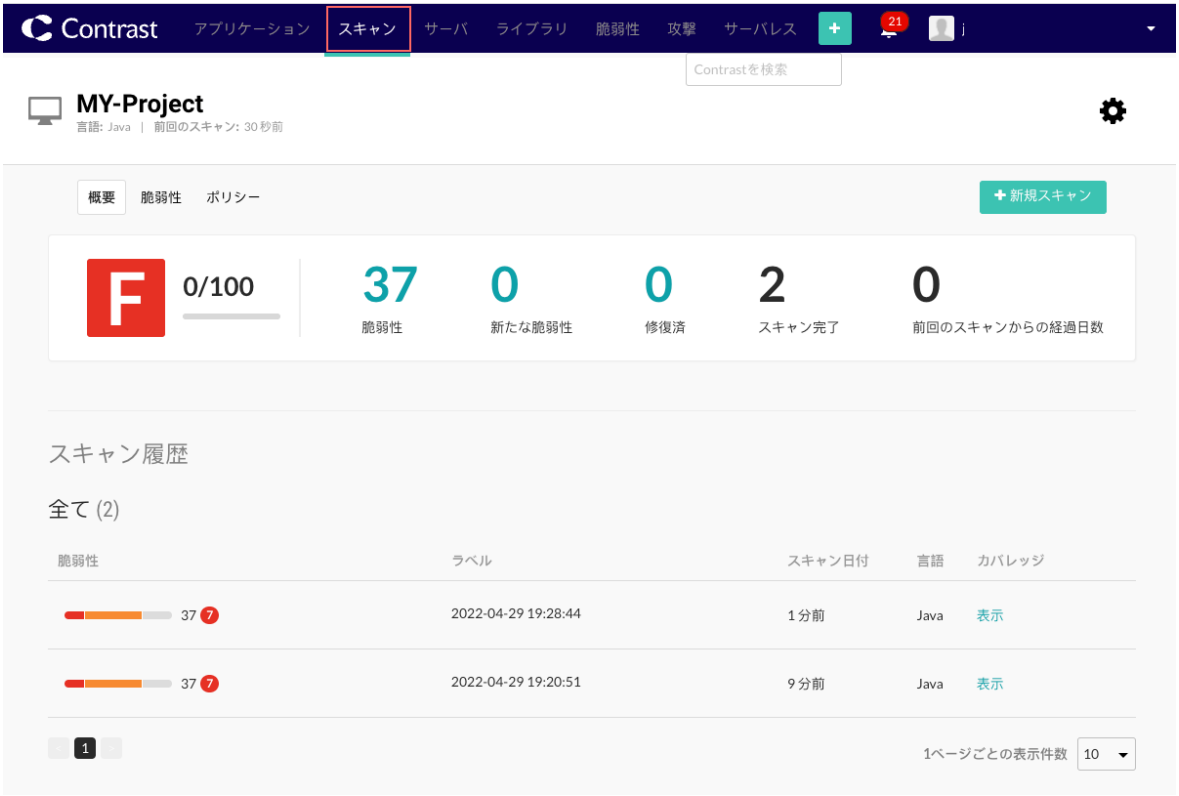
   このコマンドを初めて実行すると、`--project_name` オプションで指定した名前でプロジェクトが作成されます。
   コマンドの出力は、次のサンプルのようになります。

   ```
   project created ID is 788f9734-b933-4f05-b391-c130931baf88
    Uploaded file successfully.
    Response:  {
     id: '5091d134-93ea-4873-8110-8cf99d14606e',
     organizationId: '74f4cd04-6ca9-4eb7-a7a7-78909c2101cc',
     projectId: '788f9734-b933-4f05-b391-c130931baf88',
     filename: 'param.war',
     createdTime: '2022-04-04T10:06:16.952+00:00'
   }
   Timeout set to 5 minutes
   Waiting for results...
   New Results:  5
   Fixed Results: 0
   Total Results:  5
   ```

次回、同じプロジェクトに対してこのコマンドを実行すると、アップロードするファイルがそのプロジェクトに追加されます。コマンドの出力は、次のサンプルのようになります。

```
project already exists with this name. Getting ID...
 project ID is 788f9734-b933-4f05-b391-c130931baf88
 Uploaded file successfully.
 Response:  {
  id: '94b4e065-0e0f-46bb-b1d8-9f85bd03c602',
  organizationId: '74f4cd04-6ca9-4eb7-a7a7-78909c2101cc',
  projectId: '788f9734-b933-4f05-b391-c130931baf88',
  filename: 'param.war',
  createdTime: '2022-04-04T10:07:01.230+00:00'
}
Timeout set to 5 minutes
Waiting for results...
New Results:  5
Fixed Results: 0
Total Results:  5
```

4. スキャン完了後、Contrast Web インターフェースにスキャンプロジェクトの詳細と結果が表示されます。



インテグレーションの例

- GitHub にスキャンを組み込む (585ページ)
- Jenkins にスキャンを組み込む (586ページ)
- GitLab にスキャンを組み込む (586ページ)

例：GitHub にスキャンを組み込む

GitHub に Contrast Scan を組み込む前にスキャンのインテグレーション手順 (583ページ)について確認してください。

以下は、GitHub のワークフローに Contrast Scan を設定する例です。

```
- name: Set up contrast-cli
  run: |
    npm install --location=global @contrast/contrast@2.0.0
- name: Scan file
  env:
    CT_API_KEY: ${{ secrets.CONTRAST__API__API_KEY }}
    CT_AUTH_TOKEN: ${{ secrets.CONTRAST__API__AUTH_TOKEN }}
    ORG_ID: ${{ secrets.CONTRAST__API__ORGANIZATION_ID }}
    URL: ${{ secrets.CONTRAST__API__URL }}
  run: |
    contrast-cli --scan ./target/${{ inputs.SERVICE_NAME }}-${{ steps.build-
service.outputs.version }}.jar \
      --api_key $CT_API_KEY \
      --authorization $CT_AUTH_TOKEN \
      --organization_id $ORG_ID \
      --host $URL \
      --project_name MY-Project \
      --language JAVA --wait_for_scan
```

例：Jenkins にスキャンを組み込む

Jenkins に Contrast Scan を組み込む前にスキャンのインテグレーション手順 (583ページ)について確認してください。

Contrast Security は、Jenkins パイプラインに Contrast Scan を組み込むためのスクリプトを提供しています(これらのスクリプトにアクセスするには、サポートにお問い合せください)。

- **Jenkins パイプラインスクリプト：** `Jenkins_Script_SCAN` スクリプトでは、Contrast Scan ローカルエンジンの JAR ファイルを使用します。プロジェクトの JAR ファイルが、GitHub リポジトリにあることを前提としています。
- **Docker コンテナで使用する Jenkins パイプラインスクリプト：** `Jenkins_Script_SCAN_GH_Container_Image` スクリプトでは、パイプラインスクリプトで Docker コンテナ (564ページ)を使用します。

インテグレーション手順

この例では、Contrast Scan を Jenkins に組み込むための設定方法について説明します。

1. ご利用のローカル環境に Jenkins インスタンスをセットアップします(Jenkins のドキュメントを参照ください)。
   Jenkins のインスタンスが既にある場合は、この手順をスキップしてください。
2. 以下のソフトウェアをインストールします(まだインストールしていない場合) :
   - Java 11
   - 使用している環境用のプラグイン(例えば、Docker プラグイン)
3. 新しいパイプラインを作成し、Contrast のスクリプトをコピーします。
4. Contrast の認証情報をグローバル変数または環境変数に設定します。
   例：`URL`、`USER_NAME`、`API_KEY`、`SERVER_KEY`、`ORGANIZATION`、`CONTRAST_PAT`
   - `CONTRAST_PAT` には、Contrast サポートからお客様に提供される個人用アクセストークン(PAT)を指定します。PAT のアクセス権により、GitHub にログインし、Contrast Scan ローカルエンジンのファイルをダウンロードできます。
   - Jenkins に認証情報を追加するには、**Jenkins の管理 > Manage Credentials > 認証情報の追加 (Secret Text として)**を選択します。
5. お使いのパイプラインスクリプトで、全ての認証情報と変数を参照してください。

例：GitLab にスキャンを組み込む

GitLab に Contrast Scan を組み込む前にスキャンのインテグレーション手順 (583ページ)について確認してください。

ここでは、GitLab パイプラインを設定して以下の処理を行う例をご紹介します。

- リポジトリからコードをプル(pull)
  GitLab をコードリポジトリとして使用している場合は、この処理は必要がない場合があります。
- コードをビルド
- 生成された JAR ファイルをスキャン

パイプラインの設定例

以下は、GitLab パイプラインを設定するためのサンプルの YAML ファイルです。

```
stages:          # List of stages for jobs, and their order of execution
  - pull
  - build
  - scan
#   - deploy

pull:
  stage: pull
  artifacts:
    paths:
    - WebGoat
  script:
    - git clone -b main https://github.com/WebGoat/WebGoat.git

build:
  stage: build
  image: maven:3.8.1-openjdk-17-slim
  artifacts:
    paths:
    - $CI_PROJECT_DIR
  dependencies:
    - pull
  script:
    - apt-get update && apt-get install -y jq
    - VERSION=$(echo "$(curl --fail --silent "https://search.maven.org/
solrsearch/select?q=g:"com.contrastsecurity"&a:"contrast-
agent"&rows=20&wt=json" | jq -r
'.response.docs[0].latestVersion')") && curl --silent https://
repo1.maven.org/maven2/com/contrastsecurity/contrast-agent/${VERSION}/
contrast-agent-${VERSION}.jar -o
 /tmp/contrast.jar
    - ls -l /tmp
    - cd WebGoat
    - export MAVEN_OPTS="-javaagent:/tmp/contrast.jar -
Dcontrast.api.url=$AGENT_URL -Dcontrast.api.api_key=$AGENT_API_KEY -
Dcontrast.api.service_key=$AGENT_SERVICE_KEY -
Dcontrast.api.user_name=$AGENT_USER_NAME -
Dcontrast.agent.java.standalone_app_name=Webgoat-gitlab  -
Dcontrast.server.name=Gitlab -
Dcontrast.application.version=$CI_COMMIT_SHORT_SHA"
    - mvn -DskipTests clean install

scan:         # This is the step for Contrast Scan
  stage: scan
  image: node:18.19-slim
  dependencies:
```

```
    - build
  script:
    - ls -la
    - npm install -g @contrast/contrast@2
    - contrast version
    - contrast auth --api-key $API_KEY --authorization $AUTH --organization-
id $ORG_ID --host $URL
    - contrast scan -f $CI_PROJECT_DIR/WebGoat/target/webgoat-2023.7.jar --
fail --severity high

deploy: #TODO
```

サーバ

Contrast Web インターフェイスのサーバページで、サーバに関する情報を表示して、各環境(開発、テストおよび本番)ごとにサーバを設定することができます。そして、各環境でコードを実行して環境間の違いを比較できます。Contrast では、サーバを指定するためのシェルが用意されます。シェルが設定されると、Contrast で脆弱性の検知が始まります。

サーバの設定

Contrast での各サーバエントリは、アプリケーションに組み込んだ Contrast エージェントに設定した値によって表されます。各エージェントで以下を設定すると、新規に一意のサーバエントリが作成されます。

サーバのカスタム設定を定義するには、エージェントの設定ファイルで以下のオプションを使用します。

- **サーバ名(server name)**：デフォルト値は、ホスト名です。
- **サーバパス(server path)**：エージェントのプロセスが実行されているパス。
- **サーバタイプ(server type)**：アプリケーションをホストしているサーバの種類。

これらの設定にデフォルトの値ではなくカスタム値を使用することで、サーバエントリの重複を避けることができます。

以下の設定でサーバの環境を指定します。

- **サーバの環境(server environment)**：このサーバを使用する環境。
  有効な値は、`DEVELOPMENT`、`QA`、`PRODUCTION` です。これらの値では、大文字と小文字を区別しません。デフォルト値は、`DEVELOPMENT` です。

Contrast エージェントは、サポート対象となる全てのサーバタイプを自動的に認識します。サーバタイプが自動的に認識されない場合は、サポート対象外のテクノロジでエージェントが実行されている可能性があります。各エージェント (45ページ)の *サポート対象テクノロジ*のセクションをご確認ください。サポート対象外の環境での実行は、ルートカバレッジなどの一部の機能に影響が出る可能性があります。

設定ファイルの定義

以下は、サーバに関するプロパティで、設定ファイルでカスタムの値を指定できます。

```
# server
# Use the settings in this section to set
# metadata for the server hosting this agent.

server:

  # Override the reported server name.
  name: localhost
```

```
# Override the reported server path.
path: NEEDS_TO_BE_SET

# Override the reported server type.
type: NEEDS_TO_BE_SET

# Override the reported server environment.
# environment: DEVELOPMENT
```

エージェントの設定手順

サーバのカスタム設定を定義する際は、使用しているエージェントの各設定手順を参照してください。

- .NET Core エージェントの設定 (247ページ)
- .NET Framework エージェントの設定 (181ページ)
- Go エージェントの設定 (480ページ)
- Java エージェントの設定 (103ページ)
- Node.js エージェントの設定 (304ページ)
- Python エージェントの設定 (362ページ)
- Ruby エージェントの設定 (419ページ)

オプションの設定

Contrast Web インターフェイスには、サーバを設定するためのその他のオプションがあります。

- サーバの設定 (593ページ)
- Syslog への出力 (596ページ)

サーバの表示

サーバの一覧には、組織内のサーバに関する情報が表示されます。

- **名前**：サーバの名前。
- **前回のオンライン**：サーバに関連付けられているアプリケーションのエージェントが Contrast サーバにアクティビティを報告した最後の時間。
- **環境**：サーバがデプロイされている環境(開発、QA、本番)。
- **アプリケーション**：サーバに関連付けられているアプリケーション。

サーバの一覧から、各サーバの Assess と Protect の設定を管理することもできます。

手順

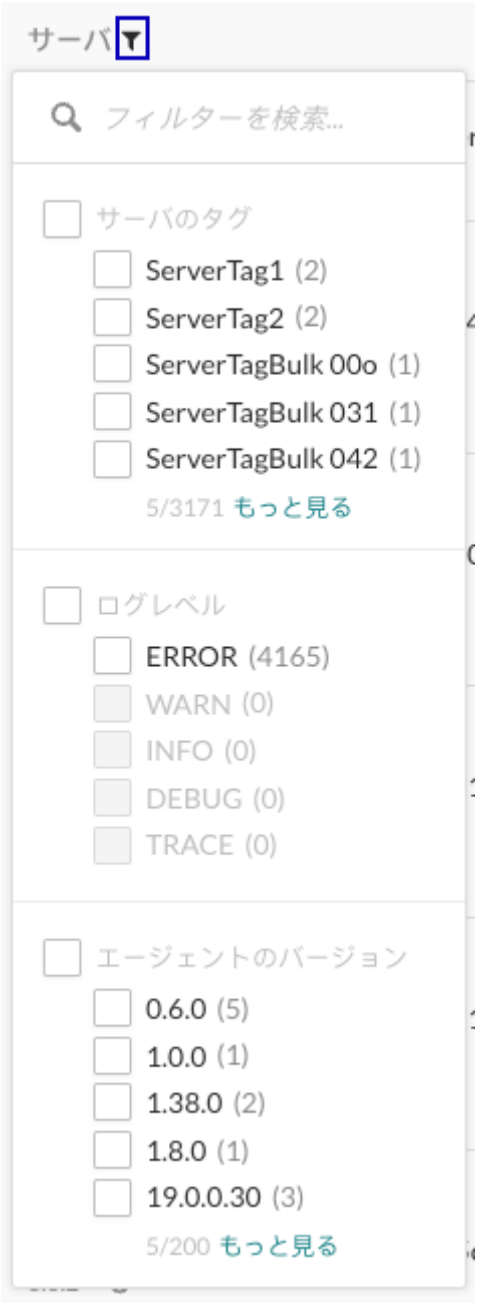
1. ナビゲーションバーで**サーバ**を選択し、組織内にある全てのサーバの一覧を表示します。
2. サーバのステータスで一覧にフィルタをかけるには、一覧の最上部にある小さい三角形( )を選択します。

   また、虫眼鏡のアイコン( )を選択して、特定のサーバを名前で検索することもできます。

フィルタには次のものがあります：

- **全て**：組織内の全てのサーバを表示します。
- **Protect あり**：Protect がオン(有効)になっているサーバを表示します。
- **Protect なし**：Protect がオフ(無効)になっているサーバを表示します。
- **オンライン**：Contrast がアクセスできるサーバを表示します。
- **オフライン**：Contrast がアクセスできないサーバを表示します。
- **バージョン遅れ**：サーバに関連付けられているアプリケーションのエージェントのバージョンが古いサーバを表示します。エージェントを新しいバージョンに更新することを検討してください。
- **Protect 部分的にあり**：Protect がオンになっていますが、再起動が必要なサーバが表示されます。
  サーバに関連付けらているアプリケーションのエージェントの設定を変更した場合、Protect を有効にするためにサーバの再起動が必要になることがあります。

3. サーバの一覧で表示を絞り込むには、「サーバ」列のヘッダの横にあるフィルターアイコン(▼)を選択します。



フィルタには次のものがあります：

- **サーバのタグ**：サーバに割り当てたタグ。
  サーバにタグを割り当てるには、該当するサーバの行の最後にカーソルを合わせ、タグ(🏷)アイコンを選択します。
- **ログレベル**：サーバに割り当てたログレベル (960ページ)。
- **エージェントのバージョン**：サーバに関連付けられているアプリケーションのエージェントのバージョン。

4. 「環境」列のヘッダの横にあるフィルターアイコン(▼)を選択すると、環境で一覧を絞り込むことができます。
   フィルタ：開発環境、QA、本番環境
5. Assess (813ページ)と Protect (815ページ)列のトグルボタンを使用して、オンまたはオフに切り替えることができます。
   - Assess と Protect のオン/オフの切り替えに Contrast Web インターフェイスのみを使用する場合、特定のサーバに対する設定は、オンなら緑色、オフなら灰色になります。この設定は、Contrast Web インターフェイスで変更できます。

- Contrast Web インターフェイス以外の方法で Assess または Protect の設定を指定した場合(エージェント設定ファイルなど)、オンなら緑色で使用不可になり、オフなら灰色で使用不可になります。この場合は、Contrast Web インターフェイスから変更できません。

- Contrast Web インターフェイスからの設定が使用不可になっている場合は、その設定にカーソルを合わせて、設定されている箇所を確認できます。Contrast でどの設定を有効な設定として使用するかは、優先順位 (62ページ)によって決まります。

6. 特定のサーバに関する詳細 (592ページ)を表示するには、サーバ名を選択します。

サーバの詳細情報

サーバの一覧からサーバ名を選択して「概要」タブを表示すると、サーバの設定とサーバに関連付けられたアプリケーションに関する情報が参照できます。

また、Protect と Assess の設定を管理することができます。

サマリー

概要タブの上部には、サマリーとして次の値が表示されます。

- **Assess と Protect の設定**
  Assess と Protect の利用を設定するには、設定ファイル、変数、または Contrast Web インターフェイスを使用することができます。設定した方法により、Contrast Web インターフェイス上で設定を変更できるかどうかが決まります。
  - Assess または Protect のオン/オフの切り替えに Contrast Web インターフェイスのみを使用する場合、トグルボタンが、オンなら緑色、オフなら灰色になります。この設定は、Contrast Web インターフェイスで変更できます。

  - Contrast Web インターフェイス以外の方法で Assess または Protect の設定を指定した場合(エージェント設定ファイルなど)、オンなら緑色で使用不可になり、オフなら灰色で使用不可になります。この場合は、Contrast Web インターフェイスから変更できません。

  Contrast Web インターフェイスからの設定が使用不可になっている場合は、その設定にカーソルを合わせて、設定されている箇所を確認できます。Contrast でどの設定を有効な設定として使用するかは、優先順位 (62ページ)によって決まります。
- **エージェントのバージョン**：このサーバに関連付けられたエージェントのバージョン。
- **ライブラリ**：選択したサーバに関連付けられたアプリケーションで確認されたオープンソースライブラリの数。脆弱なライブラリの数も表示されます。
  表示されている数字を選択すると、ライブラリの一覧が表示されます。
- **前回の起動から経過**：サーバが最後に起動してから経過した時間。
- **前回のオンラインから経過**：このサーバに関連付けられたエージェントが Contrast サーバにアクティビティを報告してから経過した時間。

統計
統計のセクションには、以下の情報が表示されます。

- **脆弱性**：Assess がオンになっている場合、このサーバに関連付けられているアプリケーションで検出された脆弱性の数。
  フィルターを使用すれば、表示を変更できます。脆弱性のバーにカーソルを合わせると、詳細が表示されます。
- **攻撃**：Protect がオンになっている場合、このサーバのアプリケーションで検知された攻撃の数。
  フィルターを使用すれば、表示を変更できます。攻撃のバーにカーソルを合わせると、詳細が表示されます。
- **アプリケーション**：このサーバに関連付けられているアプリケーション。
  アプリケーションのリンクを選択すれば、アプリケーションの詳細が表示されます。

アクティビティ
アクティビティのグラフには、選択した期間で Contrast サーバが受信したエージェントのレポートの集計が表示されます。

サーバの設定
サーバの設定画面より、各環境(開発、テスト、本番)でサーバがどのように機能するかを設定できます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで、**サーバ**を選択します。
2. 以下のいずれかの方法で、設定を変更するサーバを検索します。
   - サーバ列の上部にあるフィルターアイコン(▼)を選択
   - 虫眼鏡のアイコン(🔍)を使用して検索
3. 以下のいずれかの方法で、「サーバの設定」にアクセスします。
   - サーバの行の最後にカーソルを合わせ、**設定**アイコン(⚙)を選択
   - サーバの名前を選択してドリルダウンして、画面右上の**設定**アイコン(⚙)を選択
4. 必要に応じて、設定を変更します。
   - サーバ名を変更する。
   - サーバが稼働する環境(Development、QA、Production)を指定する。
   - 「サーバログファイル」のフィールドに、優先するパスを入力して、既存のサーバログファイルのパスを上書きする。

     注記
     サーバのログファイルは、ファイル形式が.LOG または.TXT のみに制限されます。

   - サーバのログレベル (960ページ)を設定する。
   - ボットのブロックを設定する。
     ボットをブロックすることで、スクレーパー、攻撃ツール、その他の自動化からの不要なトラフィックをブロックできます。
     ブロックされたポッドのアクティビティを表示するには、**攻撃 >攻撃イベント**で、フィルターに**自動化**を選択します。
     サポート対象の言語： Java、.NET Framework、.NET Core、Ruby、Python

     注記
     ボットのブロックは、各言語(Java、.NET Framework、.NET Core、Ruby、Python)のエージェントのYAML 設定ファイル (63ページ)で指定できます。

- **パフォーマンス向上のためサンプリングを有効にする (594ページ)**を選択する。
  この設定は、Assess が有効な時に利用できます。
  サンプリングには、以下を設定します。
  - **基準**：サンプリングが完了するまでに、Contrast で URL を解析する回数。デフォルトの設定は、**5** です。
  - **頻度**：基準のサンプル回数を取得後、毎回 N 番目のリクエストのみを解析します。頻度には、N の値を指定します。デフォルトの設定は、**10** です。
  - **サンプル保持画面** ：基準に戻る前に、Contrast でサンプルを保持する秒数。指定した秒数が経過すると、サンプリングはリセットされ基準サンプルが再度行われます。デフォルトの設定は、**180** です。

  例：

```
contrast.assess.sampling.request_frequency    25
contrast.assess.sampling.window_ms    360_000
contrast.assess.sampling.baseline    1
```

- **Syslog へ Protect イベントの出力を有効にする (596ページ)**を選択する。
  この設定は、Protect が有効な時に利用できます。
  サーバが syslog に出力する syslog メッセージの重要度レベルを選択します。Contrast で提供される syslog メッセージの重要度は、syslog RFC 3164 の仕様に従って分類されます。

アプリケーションのサンプリング

Contrast エージェントをアプリケーションに組み込んだ後にパフォーマンスの問題が発生した場合は、アプリケーションのサンプリングを有効にすることを検討してください。

サンプリングを有効にすると、一意に識別されたリクエストをもとに、抜粋してエージェントを短時間オフにすることができます。アプリケーションが同じリクエストに頻繁に応答している場合、エージェントはそれを何度も解析する必要はありません。サンプリングによって、エージェントはコードが異なるコンテキストにある場合にのみコードを解析するようになります。

ベストプラクティス：テスト環境で長時間のテストを実行する場合に、サンプリングを有効にすることを検討してください。この状況では高レベルのアクティビティが発生する可能性が高いため、サンプリングを有効にすることでアプリケーションのパフォーマンスを改善できます。

ご利用の環境がサンプリングに適しているのであれば、サーバの設定 (593ページ)にて有効にできます。

サンプリングの仕組み

1. Contrast エージェントは、同じ URL が複数回呼び出されているのを確認すると、「基準」に指定された回数に基づいて URL を解析します。
2. その後、Contrast エージェントが同じ URL を引き続き確認した場合、「頻度」の設定に基づいてのみ URL をチェックします。
3. サンプルは、「サンプル保持」の設定で指定した秒数だけ保持されます。「サンプル保持」の設定で指定した時間が経過すると、「基準」の設定に従って再度 URL が解析されます。

自動診断データ収集の使用

自動の診断データ収集により、Contrast で診断情報を手動で検索またはアップロードすることなく収集できます。自動の診断データ収集を使用することで、技術的な問題のトラブルシューティング時に Contrast サポートとの連携が容易になります。

注記
現在、この機能はプレビューモードとなっています。トラブルシューティング中にこの機能を有効にする必要がある場合は、Contrast サポートから連絡があります。

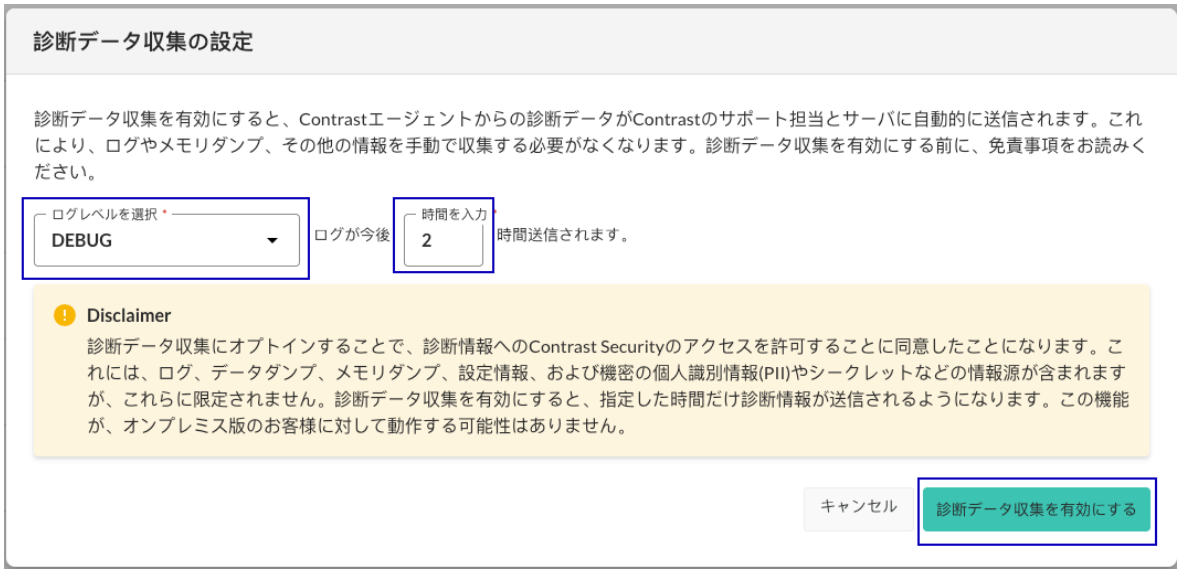
この機能を有効にすると、ログ、システムデータ、その他の診断情報が Contrast エージェントから収集され、Contrast サーバに送信されます。Contrast のサポート担当者は、トラブルシューティング作業中にこの情報にアクセスします。

開始する前に

- 現在、診断データ収集をサポートしているのは Java エージェントのみです。
- 診断データを収集する時間は、指定した値に基づきますが、1 時間から 25 時間までです。

手順

1. Contrast Web インターフェイスのナビゲーションバーで、**サーバ**を選択します。
2. サーバ行の端にカーソルを合わせ、診断( )アイコンを選択します。
3. 診断データ収集の設定で、Contrast サポートにアクセスさせたいログのログレベルを選択します。
   - **TRACE** ログ：1 つまたは複数のセッション中に生成されたトレースメッセージが含まれます。
   - **DEBUG** ログ：バグやその他の問題を特定するのに役立つ情報が含まれます。



4. 収集する時間を 1 時間から 25 時間の間で指定します。トラブルシューティングの作業に必要な時間については、Contrast サポートがアドバイスします。
5. **診断データ収集を有効にする**を選択します。
   サーバ行の端にカーソルを合わせると、診断アイコンの色が緑( )に変わります。
6. 診断アイコン( )を選択し、診断データ収集の設定に表示さているキーをコピーします。このキーをサポート担当者に渡してください。

7. 診断データ収集を無効にするには、手順 1 と 2 に従い、診断データ収集の設定を開きます。そして、**データ収集を停止**を選択します。

Syslog への出力

Contrast では、Contrast Security のログだけでなく、リモートの syslog サーバにセキュリティログを送信できます。 Syslog データは、共通イベント形式(CEF)で、ほとんどの SIEM(Security Incident Event Management)ソフトウェアで解析できます。

重要

Syslog 出力を有効にするサーバには、Protect ライセンスを適用する必要があります。

Syslog が外部メッセージを受信できるように、リモートログを有効にしなければならない場合があります。

サーバの Syslog メッセージは、エージェントによって送信されます。

Syslog 出力は TCP ではサポートされません。

1. 組織レベルでサーバのデフォルト設定 (828ページ)を指定する場合、 **Syslog へ Protect イベントの出力を有効にする**のチェックボックスをオンにします。追加のフィールドが表示されるので、必要な設定を入力します。
2. Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択すると、サーバの一覧が表示されます。Syslog 出力の設定は、個々のサーバまたは同時に複数のサーバ に有効にすることができます。Syslog のデフォルト値がすでに組織レベルで設定 (828ページ)されている場合は、サーバレベルでの設定時に値が事前に入力されます。
   - **個々のサーバ**：個々のサーバで syslog を有効にするには、該当サーバの行にカーソルを合わせ、**サーバの設定**アイコンを選択します。
   - **複数のサーバ**：チェックマークを使用して複数のサーバを選択し、ページの下部に表示される一括アクションメニューで**サーバの設定**アイコンを選択します。

   注記

   選択した 1 つ以上のサーバで syslog を有効にできない場合は、対象となるサーバでのみ syslog が有効になります。
3. **サーバの設定**画面で、**Syslog へ Protect イベントの出力を有効にする**のチェックボックスをオンにします(複数サーバの場合、最初にチェックボックス横の**編集**をクリックする必要があります)。

   注記

   選択した対象のサーバが異なる環境にある場合は、該当するサーバのデフォルトの設定を使用するか、全てのサーバの設定を手動で構成するかを選択できます。

4. **Syslog サーバホスト**を入力します。ここには、完全修飾ドメイン名(ホスト名だけでなく)または IP アドレスを指定します。例：*email.mydomainname.com* や 38.124.154.50
5. **ポート**を入力します。

6. **機能**を入力します。
7. **Syslog メッセージの重要度**を選択します。
8. 設定を**保存**すると、サーバで syslog が有効になります。
9. Syslog が有効になると、サーバの一覧でサーバ名の横に灰色の矢印アイコンが表示されます。アイコンにカーソルを合わせると、Protect イベントの出力場所を参照できます。
   サーバの設定を編集するには、上記のステップを繰り返して画面の必要な値を更新し、変更を保存してください。

ライブラリ

アプリケーションで使用されるライブラリのセキュリティは、アプリケーション全体のセキュリティに影響を及ぼします。

ライブラリには、公開または内製のものがあります。公開ライブラリは、Maven (Java)、NuGet (.NET)、npm (Node.js)、RubyGems (Ruby)、PyPI (Python)、pkg.go (Go)、Composer (PHP)から提供されるオープンソースのライブラリで、Contrast では A から F のスコア (610ページ)が付けられます。内製のライブラリは、サードパーティ製の商用ライブラリや独自に作成されたカスタムライブラリです。 内製のライブラリは、Contrast でスコア付けされません。

Contrast エージェントは、アプリケーションに含まれるオープンソースライブラリを自動的に識別します。ライブラリで検出された脆弱性を特定し、ライブラリがランタイムで使用されているかどうかも確認します。

そのため、Contrast でライブラリファイルのハッシュ値が作成されます。このハッシュ値を使用して、ファイルの内容が既知のライブラリファイルのデータベースと比較されます。データベースにハッシュがある場合は、ライブラリにスコア (610ページ)が割り当てられ、ライブラリのバージョン情報やライブラリで検出された全ての脆弱性情報(CVE)が Contrast サーバに報告されます。

注記

ライブラリがカスタムファイルの場合、ハッシュがデータベースに無いため、このライブラリは Contrast エージェントによって「不明(Unknown)」として報告されます。これは、最新のライブラリがリリースされたタイミングや、Contrast EOP(オンプレミス版)をエアギャップ環境で利用中でライブラリデータの更新 (904ページ)をしていない場合にも発生することがあります。

Java クライアントの場合、WebSphere では実行時にライブラリが再パッケージ化されるため、SHA-1 ハッシュが Contrast で認識されるものとは異なります。デプロイ時に SHA-1 を保持するには、JVM システムプロパティ `org.eclipse.jst.j2ee.commonarchivecore.ignore.web.fragment` を"true"に設定します。

また、全ての `wsadmin` の呼出しに同じパラメータが必要になります。

```
wsadmin -javaoption "-
Dorg.eclipse.jst.j2ee.commonarchivecore.ignore.web.fragment=true
"
```

Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択すると、ポートフォリオ内の全てのライブラリの概要を参照でき、ライブラリを一括管理できます。

- ライブラリの表示 (600ページ)
- オープンソースライセンスの参照 (609ページ)

- ライブラリのスコアについて (610ページ)

関連項目

CVE 検索 (611ページ)

SCA リリース情報

9 月のリリース

リリース日： 2023 年 9 月

新機能と改善点：

- 静的 SCA に Go 言語のライブラリ解析のサポートを追加しました。
- エクスポート (607ページ)機能に新しいレポート形式(XLSX)を追加しました。このレポート形式では 2 つのタブが作成されます。1 つ目のタブには標準の CSV レポートと同じ情報が含まれ、2 つ目のタブには全てのライブラリ、そのライブラリに影響する CVE、CVSS スコアおよび Contrast での深刻度がリストされます。

8 月のリリース

リリース日： 2023 年 8 月

新機能と改善点：

- Contrast Web インターフェイスにて、特定の脆弱性の深刻度を指定してライブラリをフィルタできる機能を追加しました。組織レベルでも個々のアプリケーションレベルでも行うことができます。
- 脆弱性カードに、NVD(脆弱性データベース)の CVE 項目へのハイパーリンクを追加しました。これにより、特定の CVE に関する最新の情報を参照することができます。

ライブラリの表示

ライブラリの情報を表示する方法はいくつかあります。

- Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択すると、組織で使用されている全てのライブラリが一覧で表示されます。より詳細な情報は、一覧でライブラリ名をクリックします。
- 個々のアプリケーションやサーバの詳細ページからも、ライブラリの情報を確認できます。
  - Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択し、アプリケーション名をクリックするとアプリケーションの詳細ページが表示されます。**ライブラリ**タブを選択します。
  - Contrast Web インターフェイスのナビゲーションバーで**サーバ**を選択し、サーバ名をクリックするとサーバの詳細ページが表示されます。**ライブラリ**タブを選択します。
- ライブラリ一覧の上部にある小さい三角形を選択すると、ライブラリの絞り込みができます。また、虫眼鏡のアイコンをクリックして、特定のライブラリを検索することもできます。

フィルタには次のものがあります。

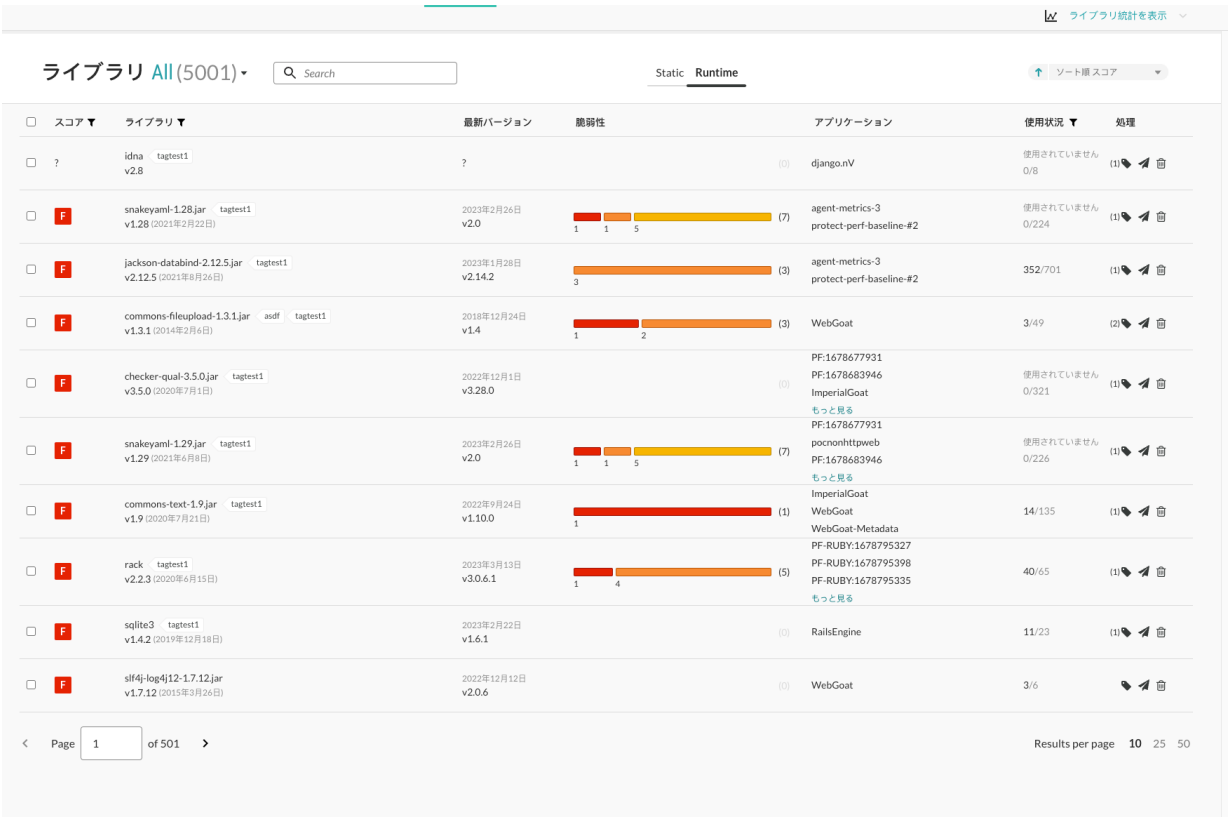
- **全て**：全てのライブラリを表示します。
- **脆弱なもの**：CVE があると識別されたライブラリのみを表示します。
- **ポリシー違反**： ライブラリポリシーに違反しているライブラリのみを表示します。
- **内製**：コード内で検出された商用のサードパーティ製ライブラリまたはカスタムビルドライブラリのみを表示します。
- **オープンソース**：コード内で検出されたオープンソースのライブラリのみを表示します。
- **高リスク**：スコア (610ページ)が C 以下のライブラリのみを表示します。
- **修復済**：修復済のステータスのライブラリが表示されます。

- ライブラリページの上部にある**ライブラリ統計を表示**を選択すると、組織のライブラリデータの分析を参照できます。各図には、高リスクとなる年数やライブラリのスコアなど、各カテゴリの統計的な平均値と内訳が表示されます。
  ライブラリは、スコア (610ページ)の評価が C 以下の場合、リスクが高いとみなされます。

静的タブとランタイムタブ

Contrast でのライブラリ情報は、2 つのタブに分かれます。

- **静的**： Contrast CLI (675ページ)で解析された マニフェスト(*package.json* や *pom.xml* など)からの検出結果が表示されます。
- **ランタイム**： 実行時に解析されたアプリケーションの検出結果が表示されます。



一覧でフィルター付きの列ヘッダーを使用して、スコア、ライブラリ、深刻度、プロジェクト別にライブラリを絞り込むこともできます。ライブラリー覧には、以下の項目が表示されます。

- **スコア**：*ランタイムタブでのみ表示されます。*スコアガイド (610ページ)に基づいて評価されたライブラリのスコアがレターグレードで表示されます。
- **深刻度**： *静的タブでのみ表示されます。*ここには、ライブラリに存在するすべての脆弱性(CVE)で最も重大な深刻度が表示されます。 フィルタを使用して、深刻度に基づいてライブラリを検索できます。なお、フィルタの **Other** オプションは、最も重大な深刻度が無し(CVSS スコアが 0)で、CVE が無く、かつ内製か未知のライブラリであるものが検索されます。

- **ライブラリ**： 一覧でライブラリ名をクリックすると、そのライブラリの詳細ページに移動します。詳細ページには、ライブラリが存在するアプリケーションとサーバのリスト、およびライブラリ内で Contrast が検出した既知の脆弱性(CVE)が表示されます。
  フィルタを使用して、結果を絞り込むことができます。
  - **言語**： 特定の言語で脆弱なライブラリを検索
  - **ライセンス**： ライセンス毎にライブラリを表示
  - **環境**： 環境ごとにライブラリを表示、本番環境の脆弱なライブラリを見つけるのに役立つ
  - **サーバ**： サーバタイプで脆弱なライブラリを検索
- **最新バージョン**： ライブラリの最新バージョン。

> **注記**
> **.NET のライブラリに関して：最新バージョン**の値は、パッケージのアップグレードが可能な推奨バージョンに関連しています。ライブラリのバージョンとハッシュは、Contrast エージェントが検出したファイルによって決定されます。ハッシュはライブラリファイルのバージョンを表し、最新バージョンに表示されるアップグレードバージョンは、パッケージのバージョンを表します。

- **脆弱性**： ライブラリで検出された CVE の情報。修復時の優先順位付けの目安となります。脆弱性の棒グラフにカーソルを合わせると、深刻度別の CVE 数が表示されます。棒グラフをクリックすると、詳細パネルが開きます。

  脆弱性が存在する場合、一覧で表示され、深刻度別に色分けされます。深刻度が重大な脆弱性は、一覧の一番上に赤色で表示されます。
- **アプリケーション**：*ランタイムタブでのみ表示されます。*ライブラリを使用しているアプリケーションの一覧。
- **使用状況**： *ランタイムタブでのみ表示されます。*ライブラリ内のクラスの総数のうち、ランタイムで使用されているクラスの総数が表示されます。ランタイムで使用されているクラスがない場合は、この列には「使用されていません」と表示されます。アプリケーションでクラスがロードされる時に、Contrast エージェントが使用状況を判断します。今まで使用されていなかったクラスの場合は、使用数は減少します。数字をクリックすると、ライブラリの使用状況を分析 (605ページ)できます。そこでは、ロードされクラスの情報だけでなく、ライブラリに関連するリスクやポリシー違反の情報も確認することができます。
- **処理**： *ランタイムタブでのみ表示されます。*ここで、ライブラリにタグ (603ページ)を付けたり、送信や削除 (603ページ)を行うことができます。
- **ステータス**： *ランタイムタブでのみ表示されます。ステータスを変更するには組織の Edit (955ページ)ロールが最低限必要です*(ステータス列が表示されていない場合は、弊社サポートに連絡して、この列を有効にするよう依頼してください)。**アプリケーション** > **アプリケーション名** > **ライブラリ**タブにアクセスすると参照できます。表示/適用できるステータスには、3 種類あります。
  - ***問題無し***: このライブラリにある脆弱性は認識済みでリスクは許容できる、またはこのライブラリは使用されていない状況。
  - ***修復済***：脆弱なライブラリに対応・対策済である状況。
  - ***報告済***： Contrast で脆弱性のあるライブラリが検出された状況。
- **プロジェクト**： *静的タブでのみ表示されます。*ライブラリを使用しているプロジェクトが一覧表示されます。

ライブラリの検出と削除

> **注記**
> ランタイムタブでのみ表示されます。静的に解析されたライブラリでは使用できません。

ライブラリは、ライブラリを使用するアプリケーションと、これらのアプリケーションがデプロイされているサーバに関連付けられています。

エージェントから送信されるライブラリ情報に、これまでに報告されたライブラリが含まれなくなると、そのライブラリにはそのサーバと関連付けがなくなります。

ライブラリを報告している全てのサーバからそのライブラリの情報が送信されなくなるか、そのライブラリを報告しているサーバが削除されると、アプリケーションからライブラリは削除されます。(サーバの削除は、サーバの 設定 (828ページ)に従います)。

ライブラリは、アプリケーションから手動で削除することもできます。

ライブラリを削除するには：

1. Contrast Web インターフェイスのナビゲーションバーで、**ライブラリ**を選択し、ライブラリ一覧から削除するライブラリの行を探します。
2. **処理**列にある**削除**アイコンをクリックします。このアイコンは、ライブラリの詳細ページの右上にもあります。
   複数のライブラリを一括で削除するには、左側の列でチェックマークを使用し、削除するライブラリを選択してから、ページの下部に表示される一括アクションバーから**削除**アイコンを選択します。

3. 表示される画面で、**削除**を選択して処理を確定します。処理を確定すると、ライブラリが削除されて一覧に表示されなくなります。エージェントが、以前に削除されたライブラリを報告した場合、そのライブラリはアプリケーションに含まれるため、ライブラリの一覧に再度追加されます。

ライブラリへのタグの追加

> **注記**
> ランタイムタブでのみ表示されます。静的に解析されたライブラリでは使用できません。

ライブラリにタグを追加するには：

1. Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択して、タグを付けるライブラリの行を探します。

2. **処理**列にある**タグ**アイコンを選択します。このオプションは、ライブラリの詳細ページの右上からもアクセスできます。

3. 表示される画面で入力を始めると、既に作成済みのタグの一覧が表示されます。既存のタグを使用する場合は、ドロップダウンから 1 つ以上のタグを選択します。もしくは、新規にタグを作成する場合は、フィールドに新しいタグを入力します。タグを外すには、タグ名の横にある **X** をクリックします。変更を保存するには、**保存**を選択します。
4. 複数のライブラリにタグを付けるには、ライブラリー覧の左側の列にあるチェックマークを使用してライブラリを選択します。ページの下部に表示される一括アクションメニューで、**タグ**アイコンを選択します。
5. タグごとに絞り込むには、ライブラリー覧で**ライブラリ**列の横にあるフィルターを選択して、絞り込むタグを選択します。

6. タグは、ライブラリの詳細ページでもライブラリ名の横に表示されます。タグの横にある **X** を選択すると、タグを外すことができます。

ライブラリ情報の送信

注記
ランタイムタブでのみ表示されます。静的に解析されたライブラリでは使用できません。

脆弱なライブラリを追跡するために、電子メールアドレスや連携しているバグ管理システム(開発担当者のためにチケットを作成)に、ライブラリ情報を送信できます。

選択した各ライブラリに対して次のデータを、メールアドレスやバグ管理システムに Contrast から送信できます。

- ライブラリ名
- 使用中のバージョン
- 脆弱性(CVE)の情報
- 影響のあるアプリケーションとサーバ
- バージョンの遅れ(現在のバージョンと最新バージョンとの比較)
- 使用状況(現在、Java および.NET でのみサポート)
- スコア (610ページ)

ライブラリの情報を送信するには：

1. Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択し、追跡したいライブラリの行を探します。
2. **処理**列にある**送信**アイコンをクリックしたら、**メールで送信**または**バグ管理システムへ送信**を選択します。
   このオプションは、ライブラリの詳細ページの右上からもアクセスできます。

3. 表示する画面で、使用するバグ管理システム (726ページ)を選択します。オプションを選択し、**送信**を選択するとチケットが作成されます。メールで送信する場合は、E メールアドレスを入力し、**送信**をクリックします。
4. 複数のライブラリの情報を送信する場合は、ライブラリ一覧の左側の列にあるチェックマークを使用して、ライブラリを選択します。ページの下部に表示される一括アクションメニューで、**送信**アイコンを選択します。選択した全てのライブラリには、共通のアプリケーションが少なくとも 1 つ必要です。

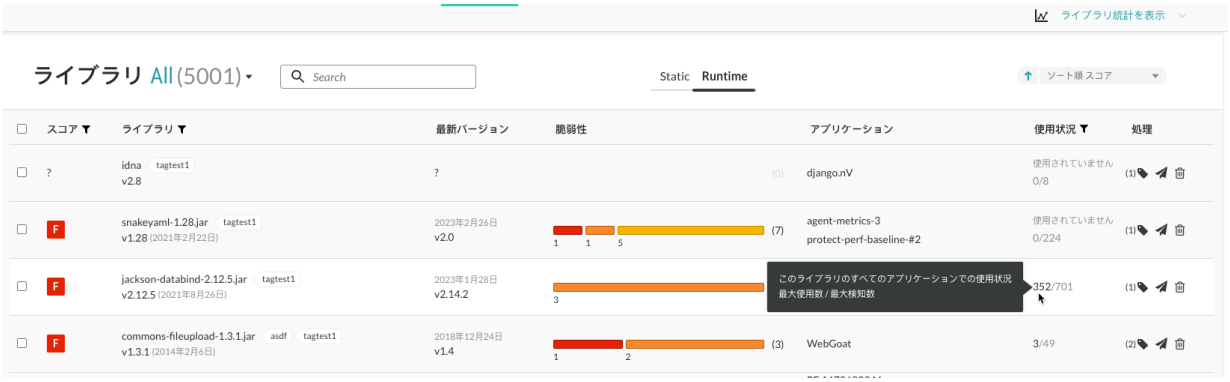
実行時のライブラリの使用状況の分析

実行時のライブラリの使用状況を確認することで、ライブラリのどの部分がアプリケーションで実際に使用されているかを知ることができます。そして、ライブラリがアプリケーションに与える影響を把握できるため、脆弱性(CVE)に関する調査時間を短縮できます。また、セキュリティ担当者は、アプリケーションが実行時に脆弱なライブラリを使用していることを開発担当者と一緒に確認できるので、作業の効率も向上します。

注記

Contrast SCA ライセンス (24ページ)のある組織のみで、使用状況の完全な情報を参照できます。詳細については、弊社営業担当の **JPNsales@contrastsecurity.com** にお問い合わせください。

Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択します。**実行時**をクリックして**使用状況**列を参照すると、実行時にライブラリが使用されているかどうか、およびどのくらい使用されているかを確認することができます。使用数は、そのライブラリで使用可能であることが判明している項目の合計数のうち、エージェントを組み込んだアプリケーションで使用されている項目の数を表します。

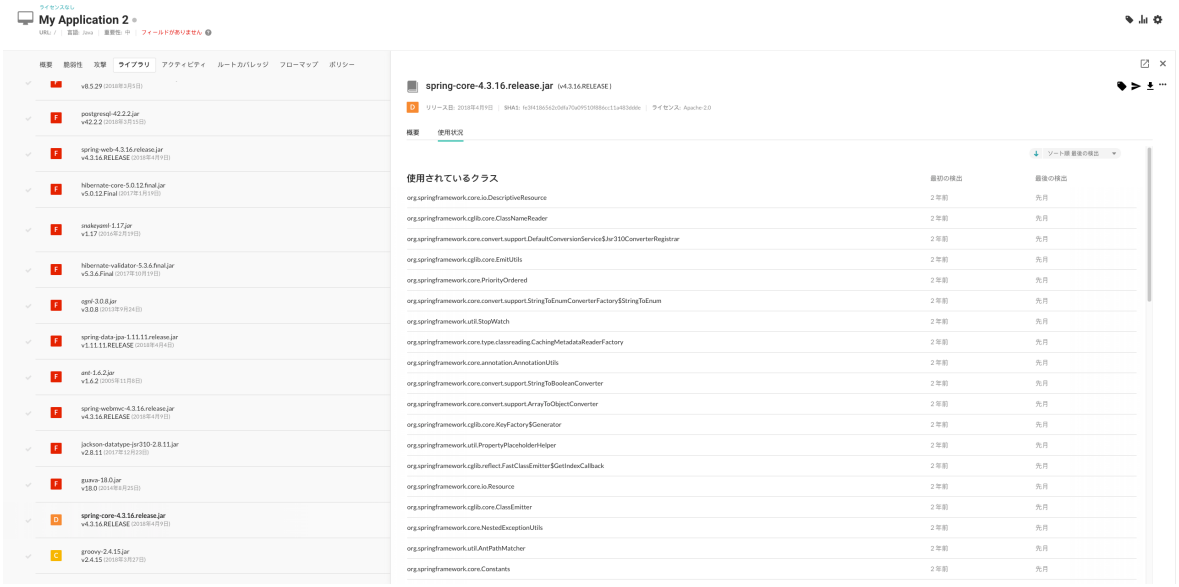


ここでの項目は、このライブラリを使用するアプリケーションの言語に応じて異なり、クラス、ファイル、または関数となります。メインアプリケーションに同じライブラリを使用する複数のアプリケーションがマージされている場合、使用中の項目はマージアプリケーションを表すものになります。

アプリケーションがライブラリを使用している場合、Contrast エージェントはライブラリ内で使用された項目を報告します。アプリケーションがライブラリ内のより多くの項目を使用すると、使用数も増加します。

適切なライセンスをお持ちの場合、以下の手順を行うことで、特定のアプリケーションにおけるライブラリの使用状況の詳細も確認できます。

1. **アプリケーション**のページで、詳細を確認するアプリケーションを選択します。
2. アプリケーションの**ライブラリ**タブを選択します。
3. 確認したいライブラリの使用数をクリックすると、概要と使用状況のパネルが表示されます。



4. **使用状況**タブをクリックすると、使用されてい各クラス、ファイル、関数を確認できます。特定のクラスを検索するには、検索アイコンをクリックします。また、Contrast でその使用が観測された最初と最後の時刻を確認することもできます。ライブラリのエクスポート (607ページ)にも、使用状況の全データを含めることができます。**概要**タブをクリックすると、*何が起こったか?*(問題の説明)と*どんなリスクであるか?*(深刻度、CVSS のスコア、CVE のタイトル、ポリシー違反)が表示されます。

> **注記**
> マージされたアプリケーションの場合、**最後の検出**と**最初の検出**の列に対する情報がある全てのアプリケーションについて、いつクラスが最初と最後に検出されたかが表示されます。

5. ライブラリにタグ (603ページ)を付けたり、ライブラリに関する情報を送信 (605ページ)することもできます。
6. パネル右上にある**さらに**(…)をクリックすると、パッケージ詳細やリポジトリ情報の参照、ライブラリの削除 (603ページ)、使用状況を CSV 形式でエクスポートすることができます。
7. 詳細パネルを閉じるには、**X** を選択します。

ライブラリ情報のエクスポート

エクスポート機能を使用して、ライブラリ情報をダウンロードできます。

ライブラリ情報をエクスポートするには：

1. Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択し、ライブラリの一覧の左側の列にあるチェックマークを使用してエクスポートしたいライブラリ(1 つまたは複数)を選択します。また、個々のライブラリ名をクリックして、詳細パネルを開きます。
2. ページの下部に表示される一括アクションメニュー、または詳細パネルの右上より、**エクスポート**アイコンを選択し、エクスポートしたい形式を選択します。

エクスポートファイルは、デスクトップにダウンロードされます。なお、XLSX 形式のエクスポートの場合は、エクスポートファイル内に **Libraries**(ライブラリ)と **Vulnerabilities**(脆弱性)という 2 つのタブが作られます。

データフィールド

エクスポートされたファイルには、ライブラリごとに以下のデータフィールドが含まれます。

- Library Name(ライブラリ名)
- Language(言語)
- Version(バージョン)
- Release Date(リリース日)
- Latest Version(最新バージョン)
- Grade(スコア)
- SHA1
- CVE Count(CVE 数)
- Application Count (アプリケーション数)
- Server Count (サーバ数)
- Number of Classes (クラス数)
- Number of Used Classes (使用中のクラス数)
- Licenses(ライセンス)
- App Name(アプリケーション名)
- Server Name(サーバ名)
- Server Environment(サーバの環境)
- Policy Violation(ポリシー違反)
- Severity (深刻度)
- Tags(タグ)

上記に加えて、XLSX 形式のエクスポートには、「Vulnerabilities 」タブに、Hash(ハッシュ)、Library Name(ライブラリ名)、CVE Name(CVE 名)、Severity(深刻度)、Severity Code(深刻度コード)のフィールドが含まれます。

ヒント

アプリケーションに関するより複雑なカスタム SCA(ソフトウェアコンポジション解析)レポートを作成する場合は、ライブラリ API を使用して Contrast のライブラリデータにアクセスできます。また、手動による方法で追加のライブラリ情報を調べることもできます。

例えば、以下の curl リクエストはライブラリの一覧を取得するものですが、各ライブラリにそのライブラリを使用しているアプリケーション一覧を含めています。カスタムレポートで使用するために、jq ツールでデータを CSV 形式にしています。

```
$ curl -H "Authorization: $(echo -n $username:$servicekey
base64)" -H "API-Key: $apikey" https://app.contrastsecurity.com/
Contrast/api/ng/$org_id/libraries/filter?expand=apps
jq -r '.libraries[]
{name: .file_name, app_name: .apps[].name}
[.name, .app_name]
@csv'
```

オープンソースライセンスの参照

重要
Contrast SCA をご利用のお客様のみ、オープンソースライセンスを表示できます。Contrast SCA を有効にするには、組織の管理者にご連絡ください。

オープンソースライブラリのライセンス情報(SPDX 形式)を表示するには、Contrast Web インターフェイスのナビゲーションバーで**ライブラリ**を選択し、ライブラリー覧を表示します。ライセンス情報を表示するには、次のような方法があります。

- 個々のライブラリのライセンスを表示するには、ライブラリー覧でライブラリ名にカーソルを合わせます。
- 特定のライセンスのライブラリを検索するには、**ライブラリ**列の見出しの横にある**フィルター**アイコンを選択して、絞り込むライセンスを選択します。
- ライブラリ名を選択して、そのライブラリの詳細ページを表示すると、ページの上部にもライセンス情報があります。
- **パッケージ詳細**を選択すると、そのパッケージのタイトル、バージョン、作成者などを参照できます(**パッケージ詳細**は、ライブラリー覧でライブラリ名にカーソルを合わせて表示されるヒントから、またはライブラリの詳細ページの右上の情報アイコンを選択すると表示されます)。
- 各ライブラリのライセンス情報は、Contrast でエクスポート (607ページ)できる CSV または XML ファイルにも含まれます。

依存関係ツリーの表示

アプリケーションにオープンソースライブラリを追加すると、そのライブラリの依存関係もすべて継承されます。このような推移的依存関係の中には、アプリケーションに脆弱なコードをもたらす可能性のあるものがあります。Contrast CLI (685ページ)を使用すると、すべてのライブラリの依存関係が識別され、そのデータが Contrast に送信されます。そのデータによって、Contrast でこれらのライブラリを階層的な依存関係ツリーとして可視化することができます。

アプリケーションのライブラリを階層で表示するには、Contrast がコンパイル前のアプリケーションコードにアクセスする必要があります。これは、ソフトウェア開発ライフサイクル(SDLC)の中で、Contrast エージェントがデータを収集するステージとは異なります。コンパイル前のデータ収集を行うために、アプリケーションに Contrast CLI (685ページ)をインストールして実行しておく必要があります。

アプリケーションのライブラリの依存関係ツリーを表示するには：

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーションを選択します。
3. アプリケーションの**概要**ページが表示されたら、**ライブラリ**タブを選択します。
4. 右上の**依存関係ツリー**アイコン を選択すると、アプリケーションのライブラリの依存関係が階層で表示されます。

このビューに表示されるライブラリの依存関係ツリーは、Contrast CLI (685ページ)によって収集されたデータに基づいています。

- クイックビューメニューを使用すると、脆弱性のあるライブラリのみを表示できます。デフォルトでは、全てのライブラリが表示されます。右矢印を使用して個々のセクションを展開して詳細を表示したり、「全て開く」オプションを選択して全ての情報を一度に表示することができます。
- また、既知の脆弱性があるライブラリには、ライブラリ名の横に警告アイコン⚠が表示されます。警告アイコンをクリックすると脆弱性の情報が表示されます。

- 特定のライブラリを検索するには、検索アイコンをクリックします。
- カスタム日付を選択して、依存関係ツリーの履歴を表示することもできます。
- フィルターアイコンをクリックすると、試験・開発環境で利用するライブラリや本番環境で利用するライブラリを選択でき、その依存関係を表示できます。デフォルトでは、本番用ライブラリを表示するオプションが選択されています。
- マージされた (532ページ)アプリケーションには**アプリケーション**のドロップダウンが表示されるので、マージされたアプリケーションで、脆弱なライブラリがどのように使用されているかを確認することができます。依存関係ツリーは、親アプリケーションや子アプリケーションごとに表示できます。

ライブラリのスコアガイド

Contrast では、アプリケーションのライブラリの安全性を文字で表したグレードで提供し、分析時の目安として使用することができます。以下のとおり、スコアをグレード(A から F の文字)に置き換えます。

- A： 90 - 100
- B：80 - 89
- C：70 - 79
- D：60 - 69
- F：35 - 59

スコアは、次の 3 つのペナルティ要素に基づきます。

- **時間**: ライブラリの古さです。アプリケーションで使用されているバージョンのリリース日から最新バージョンのリリース日までの年数に、2.5 を掛けた数が減算されます。
- **ステータス**: ステータスは、ライブラリの日付以降のバージョン数です。アプリケーションの現在のライブラリ以降にリリースされたバージョンの数に、10 を掛けた数が減算されます。
- **セキュリティ**: ライブラリに影響を与える CVE 数です。ライブラリの CVE ペナルティは、このライブラリの全ての既知の CVE の中で最も高い深刻度に、10 を掛けた数が減算されます。

注記

組織の管理者は、セキュリティ要素のみを対象とするようスコアの設定方法を調整 (834ページ)できます。

ヒント
例：

2010 年 1 月にリリースされたライブラリを使用していて、最新版が 2013 年 9 月にリリースされた場合は、経過した年数は 2 となります。そのため、時間のペナルティは次のようになります。

```
2 x 2.5 = 5
```

バージョン 1.1.1 を使用しているが、バージョン 1.1.2 と 1.1.3 がリリースされている場合は、ペナルティは次のようになります。

```
2 x 10 = 20
```

セキュリティのスコアに 2.4 と 2.2 があるライブラリがある場合、ペナルティは次のようになります。

```
2.4 x 10 = 24
```

ライブラリの最終的なスコアは、3 つのペナルティそれぞれの値を 100 から引くことによって計算されます。

```
100 - 5 - 20 - 24 = 51
```

51 のスコアには、「F」という文字が評価として割り当てられます。

CVE 検索

ライブラリ (599ページ)内の特定の CVE を探すには、検索バーを使用します。

- 検索バーで CVE 番号を入力すると、CVE が検出されているライブラリが検索結果に自動表示されます。

- 該当するライブラリを選択すると、ライブラリの詳細が表示されます。
- 組織 (810ページ)、アプリケーション (526ページ)、サーバ (588ページ)のライブラリが検索可能です。

Contrast Serverless

Contrast Serverless は、サーバレスアプリケーション向けのセキュリティツールで、動的· 静的スキャン、グラフによる可視化、リソースの可観測性などを提供し、お使いの環境の状況を把握するのに役立ちます。

Contrast Serverless では、以下が可能です。

- 必要に応じた関数のスキャン (648ページ)
- 検出結果の表示 (649ページ)
- インベントリ基準の変更 (654ページ)
- サーバレスのスキャン設定の変更 (582ページ)
- 関数とサービスの関係の表示 (655ページ)

関連項目

- Contrast Serverless と JIRA のインテグレーション (769ページ)

Contrast Serverless リリース情報

- IAST 解析レイヤー(IDS)リリース情報 (612ページ)

IAST 解析レイヤー(IDS)リリース情報

- IAST 解析レイヤー　バージョン 1.5.0 (612ページ)
- IAST 解析レイヤー　バージョン 1.4.0 (613ページ)
- IAST 解析レイヤー　バージョン 1.3.0 (613ページ)
- IAST 解析レイヤー　バージョン 1.2.0 (614ページ)
- IAST 解析レイヤー　バージョン 1.1.0 (614ページ)
- IAST 解析レイヤー　バージョン 1.0.0 (615ページ)

IAST 解析レイヤー　バージョン 1.5.0

リリース日： 2023 年 9 月 4 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能と改善点：

- Node.js に、正規表現による DoS(ReDoS)のサポートを新たに追加しました。
- より多くのシンクをサポートするために Python の ReDoS の計測機能を強化しました。
- Node.js と Python で `Unvalidated Input`(検証されていない入力)の脆弱性のサポートを新たに追加しました。
- トレースのサポートを強化するために Lambda トリガーの検出を追加しました。
- DynamoDB の NoSQL インジェクションにおける過検知を修正しました。
- いくつかの機能拡張と問題に対処し、全体的なパフォーマンスと安定性を向上しました。

セキュリティに関する修正：

- CVE-2023-36665 に対し Node.js レイヤーの `protobufjs` を v7.1.1 に更新
- CVE-2023-38704 対し Node.js レイヤーの `import-in-the-middle` を v1.4.1 に更新

問題の可能性：

- 依存関係の衝突
  - 含まれるサードパーティ製パッケージ (616ページ)を確認して下さい。
- Node.js
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。
  - Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

IAST 解析レイヤー　バージョン 1.4.0

リリース日： 2023 年 8 月 3 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能と改善点：

- Python に、正規表現による DoS(ReDoS)のサポートを新たに追加しました(Node.js は次のリリースで対応)。
- Python と Node.js の両方に、DynamoDB の NoSQL インジェクションと MongoDB の NoSQL インジェクションのルールを新たに追加しました。
- Lambda に IAST 解析が設定されている場合に、DAST 攻撃の検証を改善するためのサポートを追加しました。
- コマンドインジェクションルールのパフォーマンスを改善しました。
- いくつかの機能拡張と問題に対処し、全体的なパフォーマンスと安定性を向上しました。

問題の可能性：

- 依存関係の衝突：
  - 含まれるサードパーティ製パッケージ (616ページ)を確認して下さい。
- Node.js：
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。
  - Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

IAST 解析レイヤー　バージョン 1.3.0

リリース日： 2023 年 7 月 4 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能と改善点：

- レイヤーサイズの改善(10MB 削減)
- 配列への入力評価の最適化
- DynamoDB の SQLI のサポートの追加
- ネストされたキーの配列評価の最適化
- イベントタイプの識別と処理の改善

問題の可能性：

- 依存関係の衝突：
  - 含まれるサードパーティ製パッケージ (616ページ)を確認して下さい。
- Node.js：
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。
  - Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

IAST 解析レイヤー バージョン 1.2.0

リリース日： 2023 年 6 月 4 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能と改善点：

- SQS、 SNS、S3 などの「Records」を使用するサービスからのイベントのサポートを追加しました。
- 攻撃フェーズでのルール評価を最適化しました。
- トレースの初期サポートを追加しました。
- コールドスタートのパフォーマンスを最適化しました。

修正された不具合：

- NodeJS レイヤーでサポートされていない構文を修正しました。
- ルールの評価を修正しました。1 つのルールが失敗しても評価が続行されるようになりました。

問題の可能性：

- 依存関係の衝突：
  - 含まれるサードパーティ製パッケージを確認して下さい。
- Node.js：
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。
  - Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

IAST 解析レイヤー バージョン 1.1.0

リリース日： 2023 年 5 月 22 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能と改善点：

- Node.js における ローカルファイルインクルード(LFI)の過検知を取り除きました。
- 様々なパフォーマンスを最適化しました。

セキュリティに関する修正：

- Prometheus における脆弱性 CVE-2019-3826

問題の可能性：

- 依存関係の衝突：
  - 含まれるサードパーティ製パッケージを確認して下さい。
- Node.js
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。
  - Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

IAST 解析レイヤー　バージョン 1.0.0

リリース日： 2023 年 5 月 11 日

現在サポートしている言語バージョン：

- Node.js 12、Node.js 14、Node.js 16
- Python 3.8、Python 3.9

最小要件：

- メモリ：256 MB
- タイムアウト：5 秒

含まれるサードパーティ製パッケージ： こちら (616ページ)を参照下さい。

新機能：

- IAST 解析(IDS)の最初のリリース
- サポートされるセキュリティルール：
  - OWASP Serverless トップ 10

問題の可能性：

- 依存関係の衝突：
  - 含まれるサードパーティ製パッケージを確認して下さい。
- Node.js：
  - `async function (event, context, callback)`のように `async` と `callback` の両方を使用して定義されているハンドラー関数はサポートされません。

- Webpack ライブラリのターゲットが `commonjs2` で、TypeScript コンパイラのオプションモジュールが `Commonjs` と異なる場合は、サポートされません。

サードパーティ製パッケージ

以下の依存関係がレイヤーのラッパーとして含まれます。

- IAST 解析レイヤー　バージョン 1.5.0 (616ページ)
- IAST 解析レイヤー　バージョン 1.4.0 (622ページ)
- IAST 解析レイヤー　バージョン 1.0.0 - 1.3.0 (628ページ)

IAST 解析レイヤー　バージョン 1.5.0

以下の表は、IAST 解析レイヤーのバージョン 1.5.0 のものです。

Node.js

名前	バージョン
@cspotcode/source-map-support	0.8.1
@grpc/grpc-js	1.9.1
@grpc/proto-loader	0.7.9
@jridgewell/resolve-uri	3.1.1
@jridgewell/sourcemap-codec	1.4.15
@jridgewell/trace-mapping	0.3.9
@opentelemetry/api	1.4.1
@opentelemetry/api-logs	0.41.2
@opentelemetry/context-async-hooks	1.15.2
@opentelemetry/core	1.15.2
@opentelemetry/exporter-jaeger	1.15.2
@opentelemetry/exporter-metrics-otlp-http	0.41.2
@opentelemetry/exporter-metrics-otlp-proto	0.41.2
@opentelemetry/exporter-trace-otlp-grpc	0.41.2
@opentelemetry/exporter-trace-otlp-http	0.41.2
@opentelemetry/exporter-trace-otlp-proto	0.41.2
@opentelemetry/exporter-zipkin	1.15.2
@opentelemetry/instrumentation	0.41.2
@opentelemetry/instrumentation-aws-lambda	0.37.0
@opentelemetry/instrumentation-aws-sdk	0.36.0
@opentelemetry/instrumentation-fs	0.8.1
@opentelemetry/instrumentation-http	0.41.2
@opentelemetry/instrumentation-mongodb	0.37.0
@opentelemetry/instrumentation-mysql2	0.34.1
@opentelemetry/otlp-exporter-base	0.41.2
@opentelemetry/otlp-grpc-exporter-base	0.41.2
@opentelemetry/otlp-proto-exporter-base	0.41.2
@opentelemetry/otlp-transformer	0.41.2
@opentelemetry/propagation-utils	0.30.1
@opentelemetry/propagator-aws-xray	1.3.1
@opentelemetry/propagator-b3	1.15.2
@opentelemetry/propagator-jaeger	1.15.2
@opentelemetry/resource-detector-aws	1.3.1
@opentelemetry/resources	1.15.2
@opentelemetry/sdk-logs	0.41.2
@opentelemetry/sdk-metrics	1.15.2
@opentelemetry/sdk-node	0.41.2

@opentelemetry/sdk-trace-base	1.15.2
@opentelemetry/sdk-trace-node	1.15.2
@opentelemetry/semantic-conventions	1.15.2
@opentelemetry/sql-common	0.40.0
@protobufjs/aspromise	1.1.2
@protobufjs/base64	1.1.2
@protobufjs/codegen	2.0.4
@protobufjs/eventemitter	1.1.0
@protobufjs/fetch	1.1.0
@protobufjs/float	1.0.2
@protobufjs/inquire	1.1.0
@protobufjs/path	1.1.2
@protobufjs/pool	1.1.0
@protobufjs/utf8	1.1.0
@tsconfig/node10	1.0.9
@tsconfig/node12	1.0.11
@tsconfig/node14	1.0.3
@tsconfig/node16	1.0.4
@types/aws-lambda	8.10.119
@types/node	20.5.7
@types/shimmer	1.0.2
abbrev	1.1.1
accepts	1.3.8
acorn	8.10.0
acorn-import-assertions	1.9.0
acorn-walk	8.2.0
ansi-color	0.2.1
ansi-regex	5.0.1
ansi-styles	4.3.0
anymatch	3.1.3
arg	4.1.3
array-flatten	1.1.1
asynckit	0.4.0
available-typed-arrays	1.0.5
aws-sdk	2.1450.0
axios	1.5.0
balanced-match	1.0.2
base64-js	1.5.1
binary-extensions	2.2.0
body-parser	1.20.1
body-parser	1.20.2
brace-expansion	1.1.11
braces	3.0.2
buffer	4.9.2
bufrw	1.3.0
bytes	3.1.2
call-bind	1.0.2
chokidar	3.5.3
cjs-module-lexer	1.2.3
cliui	8.0.1
cn-otel-node	0.0.1
color-convert	2.0.1

color-name	1.1.4
combined-stream	1.0.8
concat-map	0.0.1
content-disposition	0.5.4
content-type	1.0.5
cookie	0.5.0
cookie-signature	1.0.6
create-require	1.1.1
debug	2.6.9
debug	3.2.7
debug	4.3.4
delayed-stream	1.0.0
depd	2.0.0
destroy	1.2.0
diff	4.0.2
ee-first	1.1.1
emoji-regex	8.0.0
encodeurl	1.0.2
error	7.0.2
escalade	3.1.1
escape-html	1.0.3
etag	1.8.1
events	1.1.1
express	4.18.2
fill-range	7.0.1
finalhandler	1.2.0
follow-redirects	1.15.2
for-each	0.3.3
form-data	4.0.0
forwarded	0.2.0
fresh	0.5.2
fsevents	2.3.3
function-bind	1.1.1
get-caller-file	2.0.5
get-intrinsic	1.2.1
glob-parent	5.1.2
gopd	1.0.1
has	1.0.3
has-flag	3.0.0
has-proto	1.0.1
has-symbols	1.0.3
has-tostringtag	1.0.0
hexer	1.5.0
http-errors	2.0.0
iconv-lite	0.4.24
ieee754	1.1.13
ignore-by-default	1.0.1
import-in-the-middle	1.4.2
inherits	2.0.4
ipaddr.js	1.9.1
is-arguments	1.1.1
is-binary-path	2.1.0

is-callable	1.2.7
is-core-module	2.13.0
is-extglob	2.1.1
is-fullwidth-code-point	3.0.0
is-generator-function	1.0.10
is-glob	4.0.3
is-number	7.0.0
is-typed-array	1.1.12
isarray	1.0.0
jaeger-client	3.19.0
jmespath	0.16.0
lodash.camelcase	4.3.0
lodash.merge	4.6.2
long	2.4.0
long	5.2.3
lru-cache	6.0.0
make-error	1.3.6
media-typer	0.3.0
merge-descriptors	1.0.1
methods	1.1.2
mime	1.6.0
mime-db	1.52.0
mime-types	2.1.35
minimatch	3.1.2
minimist	1.2.8
module-details-from-path	1.0.3
ms	2.0.0
ms	2.1.2
ms	2.1.3
negotiator	0.6.3
node-int64	0.4.0
nodemon	3.0.1
nopt	1.0.10
normalize-path	3.0.0
object-inspect	1.12.3
on-finished	2.4.1
opentracing	0.14.7
parseurl	1.3.3
path-parse	1.0.7
path-to-regexp	0.1.7
picomatch	2.3.1
process	0.10.1
protobufjs	7.2.5
proxy-addr	2.0.7
proxy-from-env	1.1.0
pstree.remy	1.1.8
punycode	1.3.2
qs	6.11.0
querystring	0.2.0
range-parser	1.2.1
raw-body	2.5.1
raw-body	2.5.2

readdirp	3.6.0
require-directory	2.1.1
require-in-the-middle	7.2.0
resolve	1.22.4
safe-buffer	5.2.1
safer-buffer	2.1.2
sax	1.2.1
semver	7.5.4
send	0.18.0
serve-static	1.15.0
setprototypeof	1.2.0
shimmer	1.2.1
side-channel	1.0.4
simple-update-notifier	2.0.0
statuses	2.0.1
string-template	0.2.1
string-width	4.2.3
strip-ansi	6.0.1
supports-color	5.5.0
supports-preserve-symlinks-flag	1.0.0
thriftrw	3.11.4
to-regex-range	5.0.1
toidentifier	1.0.1
touch	3.1.0
ts-node	10.9.1
tsc	2.0.4
type-is	1.6.18
typescript	4.9.5
undefsafe	2.0.5
unpipe	1.0.0
url	0.10.3
util	0.12.5
utils-merge	1.0.1
uuid	8.0.0
uuid	8.3.2
v8-compile-cache-lib	3.0.1
vary	1.1.2
which-typed-array	1.1.11
wrap-ansi	7.0.0
xml2js	0.5.0
xmlbuilder	11.0.1
xorshift	1.2.0
xtend	4.0.2
y18n	5.0.8
yallist	4.0.0
yargs	17.7.2
yargs-parser	21.1.1
yn	3.1.1

Python

名前	バージョン

Deprecated	1.2.14
asgiref	3.7.2
backoff	2.2.1
certifi	2023.7.22
charset-normalizer	3.2.0
dnspython	2.4.2
googleapis-common-protos	1.60.0
idna	3.4
importlib-metadata	6.8.0
opentelemetry-api	1.19.0
opentelemetry-distro	0.40b0
opentelemetry-exporter-otlp-proto-common	1.19.0
opentelemetry-exporter-otlp-proto-http	1.19.0
opentelemetry-instrumentation	0.40b0
opentelemetry-instrumentation-aiohttp-client	0.40b0
opentelemetry-instrumentation-asgi	0.40b0
opentelemetry-instrumentation-asyncpg	0.40b0
opentelemetry-instrumentation-boto	0.40b0
opentelemetry-instrumentation-botocore	0.40b0
opentelemetry-instrumentation-celery	0.40b0
opentelemetry-instrumentation-dbapi	0.40b0
opentelemetry-instrumentation-django	0.40b0
opentelemetry-instrumentation-elasticsearch	0.40b0
opentelemetry-instrumentation-falcon	0.40b0
opentelemetry-instrumentation-fastapi	0.40b0
opentelemetry-instrumentation-flask	0.40b0
opentelemetry-instrumentation-grpc	0.40b0
opentelemetry-instrumentation-jinja2	0.40b0
opentelemetry-instrumentation-mysql	0.40b0
opentelemetry-instrumentation-psycopg2	0.40b0
opentelemetry-instrumentation-pymemcache	0.40b0
opentelemetry-instrumentation-pymongo	0.40b0
opentelemetry-instrumentation-pymysql	0.40b0
opentelemetry-instrumentation-pyramid	0.40b0
opentelemetry-instrumentation-redis	0.40b0
opentelemetry-instrumentation-requests	0.40b0
opentelemetry-instrumentation-sqlalchemy	0.40b0
opentelemetry-instrumentation-sqlite3	0.40b0
opentelemetry-instrumentation-starlette	0.40b0
opentelemetry-instrumentation-tornado	0.40b0
opentelemetry-instrumentation-urllib	0.40b0
opentelemetry-instrumentation-urllib3	0.40b0
opentelemetry-instrumentation-wsgi	0.40b0
opentelemetry-propagator-aws-xray	1.0.1
opentelemetry-proto	1.19.0
opentelemetry-sdk	1.19.0
opentelemetry-semantic-conventions	0.40b0
opentelemetry-util-http	0.40b0
packaging	23.1
protobuf	4.24.2
pymongo	4.5.0
requests	2.31.0

setuptools	68.1.2
typing_extensions	4.7.1
urllib3	2.0.4
wrapt	1.15.0
zipp	3.16.2

IAST 解析レイヤー　バージョン 1.4.0

以下の表は、IAST 解析レイヤーのバージョン 1.4.0 のものです。

Node.js

名前	バージョン
@cspotcode/source-map-support	0.8.1
@grpc/grpc-js	1.9.1
@grpc/proto-loader	0.7.9
@jridgewell/resolve-uri	3.1.1
@jridgewell/sourcemap-codec	1.4.15
@jridgewell/trace-mapping	0.3.9
@opentelemetry/api	1.4.1
@opentelemetry/api-logs	0.41.2
@opentelemetry/api-metrics	0.32.0
@opentelemetry/context-async-hooks	1.15.2
@opentelemetry/core	1.14.0
@opentelemetry/core	1.15.2
@opentelemetry/core	1.9.1
@opentelemetry/exporter-jaeger	1.15.2
@opentelemetry/exporter-metrics-otlp-http	0.41.2
@opentelemetry/exporter-metrics-otlp-proto	0.41.2
@opentelemetry/exporter-trace-otlp-grpc	0.35.1
@opentelemetry/exporter-trace-otlp-grpc	0.41.2
@opentelemetry/exporter-trace-otlp-http	0.35.1
@opentelemetry/exporter-trace-otlp-http	0.41.2
@opentelemetry/exporter-trace-otlp-proto	0.41.2
@opentelemetry/exporter-zipkin	1.15.2
@opentelemetry/instrumentation	0.32.0
@opentelemetry/instrumentation	0.35.1
@opentelemetry/instrumentation	0.40.0
@opentelemetry/instrumentation	0.41.2
@opentelemetry/instrumentation-aws-lambda	0.34.1
@opentelemetry/instrumentation-aws-sdk	0.36.0
@opentelemetry/instrumentation-fs	0.7.4
@opentelemetry/instrumentation-http	0.40.0
@opentelemetry/instrumentation-mongodb	0.36.1
@opentelemetry/instrumentation-mysql2	0.32.1
@opentelemetry/otlp-exporter-base	0.35.1
@opentelemetry/otlp-exporter-base	0.41.2
@opentelemetry/otlp-grpc-exporter-base	0.35.1
@opentelemetry/otlp-grpc-exporter-base	0.41.2
@opentelemetry/otlp-proto-exporter-base	0.41.2
@opentelemetry/otlp-transformer	0.35.1
@opentelemetry/otlp-transformer	0.41.2
@opentelemetry/propagation-utils	0.30.1

@opentelemetry/propagator-aws-xray	1.3.1
@opentelemetry/propagator-b3	1.15.2
@opentelemetry/propagator-jaeger	1.15.2
@opentelemetry/resource-detector-aws	1.3.1
@opentelemetry/resources	1.15.2
@opentelemetry/resources	1.9.1
@opentelemetry/sdk-logs	0.41.2
@opentelemetry/sdk-metrics	1.15.2
@opentelemetry/sdk-metrics	1.9.1
@opentelemetry/sdk-node	0.41.2
@opentelemetry/sdk-trace-base	1.15.2
@opentelemetry/sdk-trace-base	1.9.1
@opentelemetry/sdk-trace-node	1.15.2
@opentelemetry/semantic-conventions	1.14.0
@opentelemetry/semantic-conventions	1.15.2
@opentelemetry/semantic-conventions	1.9.1
@protobufjs/aspromise	1.1.2
@protobufjs/base64	1.1.2
@protobufjs/codegen	2.0.4
@protobufjs/eventemitter	1.1.0
@protobufjs/fetch	1.1.0
@protobufjs/float	1.0.2
@protobufjs/inquire	1.1.0
@protobufjs/path	1.1.2
@protobufjs/pool	1.1.0
@protobufjs/utf8	1.1.0
@tsconfig/node10	1.0.9
@tsconfig/node12	1.0.11
@tsconfig/node14	1.0.3
@tsconfig/node16	1.0.4
@types/aws-lambda	8.10.81
@types/node	20.5.7
@types/shimmer	1.0.2
abbrev	1.1.1
accepts	1.3.8
acorn	8.10.0
acorn-import-assertions	1.9.0
acorn-walk	8.2.0
ansi-color	0.2.1
ansi-regex	5.0.1
ansi-styles	4.3.0
anymatch	3.1.3
arg	4.1.3
array-flatten	1.1.1
asynckit	0.4.0
available-typed-arrays	1.0.5
aws-sdk	2.1448.0
axios	1.5.0
balanced-match	1.0.2
base64-js	1.5.1
binary-extensions	2.2.0
body-parser	1.20.1

body-parser	1.20.2
brace-expansion	1.1.11
braces	3.0.2
buffer	4.9.2
bufrw	1.3.0
bytes	3.1.2
call-bind	1.0.2
chokidar	3.5.3
cjs-module-lexer	1.2.3
cliui	8.0.1
cn-otel-node	0.0.1
color-convert	2.0.1
color-name	1.1.4
combined-stream	1.0.8
concat-map	0.0.1
content-disposition	0.5.4
content-type	1.0.5
cookie	0.5.0
cookie-signature	1.0.6
create-require	1.1.1
debug	2.6.9
debug	3.2.7
debug	4.3.4
delayed-stream	1.0.0
depd	2.0.0
destroy	1.2.0
diff	4.0.2
ee-first	1.1.1
emoji-regex	8.0.0
encodeurl	1.0.2
error	7.0.2
escalade	3.1.1
escape-html	1.0.3
etag	1.8.1
events	1.1.1
express	4.18.2
fill-range	7.0.1
finalhandler	1.2.0
follow-redirects	1.15.2
for-each	0.3.3
form-data	4.0.0
forwarded	0.2.0
fresh	0.5.2
fsevents	2.3.3
function-bind	1.1.1
get-caller-file	2.0.5
get-intrinsic	1.2.1
glob-parent	5.1.2
gopd	1.0.1
has	1.0.3
has-flag	3.0.0
has-proto	1.0.1

has-symbols	1.0.3
has-tostringtag	1.0.0
hexer	1.5.0
http-errors	2.0.0
iconv-lite	0.4.24
ieee754	1.1.13
ignore-by-default	1.0.1
import-in-the-middle	1.3.5
import-in-the-middle	1.4.2
inherits	2.0.4
ipaddr.js	1.9.1
is-arguments	1.1.1
is-binary-path	2.1.0
is-callable	1.2.7
is-core-module	2.13.0
is-extglob	2.1.1
is-fullwidth-code-point	3.0.0
is-generator-function	1.0.10
is-glob	4.0.3
is-number	7.0.0
is-typed-array	1.1.12
isarray	1.0.0
jaeger-client	3.19.0
jmespath	0.16.0
lodash.camelcase	4.3.0
lodash.merge	4.6.2
long	2.4.0
long	5.2.3
lru-cache	6.0.0
make-error	1.3.6
media-typer	0.3.0
merge-descriptors	1.0.1
methods	1.1.2
mime	1.6.0
mime-db	1.52.0
mime-types	2.1.35
minimatch	3.1.2
minimist	1.2.8
module-details-from-path	1.0.3
ms	2.0.0
ms	2.1.2
ms	2.1.3
negotiator	0.6.3
node-int64	0.4.0
nodemon	3.0.1
nopt	1.0.10
normalize-path	3.0.0
object-inspect	1.12.3
on-finished	2.4.1
opentracing	0.14.7
parseurl	1.3.3
path-parse	1.0.7

path-to-regexp	0.1.7
picomatch	2.3.1
process	0.10.1
protobufjs	7.2.5
proxy-addr	2.0.7
proxy-from-env	1.1.0
pstree.remy	1.1.8
punycode	1.3.2
qs	6.11.0
querystring	0.2.0
range-parser	1.2.1
raw-body	2.5.1
raw-body	2.5.2
readdirp	3.6.0
require-directory	2.1.1
require-in-the-middle	5.2.0
require-in-the-middle	7.2.0
resolve	1.22.4
safe-buffer	5.2.1
safer-buffer	2.1.2
sax	1.2.1
semver	7.5.4
send	0.18.0
serve-static	1.15.0
setprototypeof	1.2.0
shimmer	1.2.1
side-channel	1.0.4
simple-update-notifier	2.0.0
statuses	2.0.1
string-template	0.2.1
string-width	4.2.3
strip-ansi	6.0.1
supports-color	5.5.0
supports-preserve-symlinks-flag	1.0.0
thriftrw	3.11.4
to-regex-range	5.0.1
toidentifier	1.0.1
touch	3.1.0
ts-node	10.9.1
tsc	2.0.4
type-is	1.6.18
typescript	4.9.5
undefsafe	2.0.5
unpipe	1.0.0
url	0.10.3
util	0.12.5
utils-merge	1.0.1
uuid	8.0.0
uuid	8.3.2
v8-compile-cache-lib	3.0.1
vary	1.1.2
which-typed-array	1.1.11

wrap-ansi	7.0.0
xml2js	0.5.0
xmlbuilder	11.0.1
xorshift	1.2.0
xtend	4.0.2
y18n	5.0.8
yallist	4.0.0
yargs	17.7.2
yargs-parser	21.1.1
yn	3.1.1

Python

名前	バージョン
Deprecated	1.2.14
asgiref	3.7.2
backoff	2.2.1
certifi	2023.7.22
charset-normalizer	3.2.0
dnspython	2.4.2
googleapis-common-protos	1.60.0
idna	3.4
importlib-metadata	6.0.0
importlib-metadata	6.8.0
opentelemetry-api	1.19.0
opentelemetry-distro	0.40b0
opentelemetry-exporter-otlp-proto-common	1.19.0
opentelemetry-exporter-otlp-proto-http	1.19.0
opentelemetry-instrumentation	0.40b0
opentelemetry-instrumentation-aiohttp-client	0.40b0
opentelemetry-instrumentation-asgi	0.40b0
opentelemetry-instrumentation-asyncpg	0.40b0
opentelemetry-instrumentation-boto	0.40b0
opentelemetry-instrumentation-botocore	0.40b0
opentelemetry-instrumentation-celery	0.40b0
opentelemetry-instrumentation-dbapi	0.40b0
opentelemetry-instrumentation-django	0.40b0
opentelemetry-instrumentation-elasticsearch	0.40b0
opentelemetry-instrumentation-falcon	0.40b0
opentelemetry-instrumentation-fastapi	0.40b0
opentelemetry-instrumentation-flask	0.40b0
opentelemetry-instrumentation-grpc	0.40b0
opentelemetry-instrumentation-jinja2	0.40b0
opentelemetry-instrumentation-mysql	0.40b0
opentelemetry-instrumentation-psycopg2	0.40b0
opentelemetry-instrumentation-pymemcache	0.40b0
opentelemetry-instrumentation-pymongo	0.40b0
opentelemetry-instrumentation-pymysql	0.40b0
opentelemetry-instrumentation-pyramid	0.40b0
opentelemetry-instrumentation-redis	0.40b0
opentelemetry-instrumentation-requests	0.40b0
opentelemetry-instrumentation-sqlalchemy	0.40b0

opentelemetry-instrumentation-sqlite3	0.40b0
opentelemetry-instrumentation-starlette	0.40b0
opentelemetry-instrumentation-tornado	0.40b0
opentelemetry-instrumentation-urllib	0.40b0
opentelemetry-instrumentation-urllib3	0.40b0
opentelemetry-instrumentation-wsgi	0.40b0
opentelemetry-propagator-aws-xray	1.0.1
opentelemetry-proto	1.19.0
opentelemetry-sdk	1.19.0
opentelemetry-semantic-conventions	0.40b0
opentelemetry-util-http	0.40b0
packaging	23.1
protobuf	4.24.2
pymongo	4.5.0
requests	2.31.0
setuptools	68.1.2
typing_extensions	4.7.1
urllib3	2.0.4
wrapt	1.15.0
zipp	3.16.2

IAST 解析レイヤー　バージョン 1.0.0 - 1.3.0

以下の表は、IAST 解析レイヤーのバージョン 1.0.0 から 1.3.0 までのものです。

Node.js

名前	バージョン
@babel/code-frame	7.12.11
@babel/helper-validator-identifier	7.19.1
@babel/highlight	7.18.6
@cspotcode/source-map-support	0.8.1
@eslint/eslintrc	0.4.3
@grpc/grpc-js	1.8.14
@grpc/proto-loader	0.7.7
@humanwhocodes/config-array	0.5.0
@humanwhocodes/object-schema	1.2.1
@isaacs/cliui	8.0.2
@jridgewell/resolve-uri	3.1.1
@jridgewell/sourcemap-codec	1.4.15
@jridgewell/trace-mapping	0.3.9
@nodelib/fs.scandir	2.1.5
@nodelib/fs.stat	2.0.5
@nodelib/fs.walk	1.2.8
@opentelemetry/api	1.3.0
@opentelemetry/api	1.4.1
@opentelemetry/api-metrics	0.32.0
@opentelemetry/context-async-hooks	1.13.0
@opentelemetry/core	1.13.0
@opentelemetry/exporter-jaeger	1.13.0
@opentelemetry/exporter-metrics-otlp-http	0.34.0
@opentelemetry/exporter-metrics-otlp-proto	0.34.0

@opentelemetry/exporter-trace-otlp-grpc	0.35.1
@opentelemetry/exporter-trace-otlp-http	0.35.1
@opentelemetry/exporter-trace-otlp-proto	0.34.0
@opentelemetry/exporter-zipkin	1.8.0
@opentelemetry/instrumentation	0.35.1
@opentelemetry/instrumentation	0.39.1
@opentelemetry/instrumentation-aws-lambda	0.34.1
@opentelemetry/instrumentation-aws-sdk	0.33.0
@opentelemetry/instrumentation-fs	0.7.2
@opentelemetry/instrumentation-mysql2	0.32.1
@opentelemetry/otlp-exporter-base	0.34.0
@opentelemetry/otlp-grpc-exporter-base	0.35.1
@opentelemetry/otlp-proto-exporter-base	0.34.0
@opentelemetry/otlp-transformer	0.34.0
@opentelemetry/propagation-utils	0.29.3
@opentelemetry/propagator-aws-xray	1.2.0
@opentelemetry/propagator-b3	1.13.0
@opentelemetry/propagator-jaeger	1.13.0
@opentelemetry/resource-detector-aws	1.2.3
@opentelemetry/resources	1.13.0
@opentelemetry/sdk-metrics	1.8.0
@opentelemetry/sdk-node	0.34.0
@opentelemetry/sdk-trace-base	1.13.0
@opentelemetry/sdk-trace-node	1.13.0
@opentelemetry/semantic-conventions	1.13.0
@pkgjs/parseargs	0.11.0
@protobufjs/aspromise	1.1.2
@protobufjs/base64	1.1.2
@protobufjs/codegen	2.0.4
@protobufjs/eventemitter	1.1.0
@protobufjs/fetch	1.1.0
@protobufjs/float	1.0.2
@protobufjs/inquire	1.1.0
@protobufjs/path	1.1.2
@protobufjs/pool	1.1.0
@protobufjs/utf8	1.1.0
@tsconfig/node10	1.0.9
@tsconfig/node12	1.0.11
@tsconfig/node14	1.0.3
@tsconfig/node16	1.0.4
@types/aws-lambda	8.10.81
@types/json-schema	7.0.11
@types/long	4.0.2
@types/minimist	1.2.2
@types/mocha	10.0.1
@types/mysql	git+ssh://git@github.com/types/mysql.git#c26b1bc2bac17010081455e3127a90fb2eafcec9
@types/mysql2	git+ssh://git@github.com/types/mysql2.git#89378b2cb3974ea8cdd1d633b8f056e54e5d2384
@types/node	18.16.9
@types/node	20.1.4
@types/normalize-package-data	2.4.1

@types/semver	7.5.0
@typescript-eslint/eslint-plugin	4.33.0
@typescript-eslint/experimental-utils	4.33.0
@typescript-eslint/parser	4.33.0
@typescript-eslint/scope-manager	4.33.0
@typescript-eslint/types	4.33.0
@typescript-eslint/typescript-estree	4.33.0
@typescript-eslint/visitor-keys	4.33.0
abbrev	1.1.1
accepts	1.3.8
acorn	7.4.1
acorn	8.8.2
acorn-jsx	5.3.2
acorn-walk	8.2.0
ajv	6.12.6
ansi-color	0.2.1
ansi-colors	4.1.3
ansi-escapes	4.3.2
ansi-regex	5.0.1
ansi-styles	4.3.0
anymatch	3.1.3
arg	4.1.3
argparse	1.0.10
array-flatten	1.1.1
array-union	2.1.0
arrify	1.0.1
astral-regex	2.0.0
asynckit	0.4.0
available-typed-arrays	1.0.5
aws-sdk	2.1378.0
axios	1.4.0
balanced-match	1.0.2
base64-js	1.5.1
binary-extensions	2.2.0
body-parser	1.20.2
brace-expansion	1.1.11
braces	3.0.2
buffer	4.9.2
bufrw	1.3.0
bytes	3.1.2
call-bind	1.0.2
callsites	3.1.0
camelcase	5.3.1
camelcase-keys	6.2.2
chalk	4.1.2
chardet	0.7.0
chokidar	3.5.3
cli-cursor	3.1.0
cli-width	3.0.0
cliui	8.0.1
color-convert	2.0.1
color-name	1.1.4

combined-stream	1.0.8
concat-map	0.0.1
content-disposition	0.5.4
content-type	1.0.5
cookie	0.5.0
cookie-signature	1.0.6
create-require	1.1.1
cross-spawn	7.0.3
debug	2.6.9
debug	4.3.4
decamelize	1.2.0
decamelize-keys	1.1.1
deep-is	0.1.4
delayed-stream	1.0.0
denque	1.5.1
depd	2.0.0
destroy	1.2.0
diff	4.0.2
dir-glob	3.0.1
doctrine	3.0.0
eastasianwidth	0.2.0
ee-first	1.1.1
emoji-regex	8.0.0
encodeurl	1.0.2
enquirer	2.3.6
error	7.0.2
error-ex	1.3.2
escalade	3.1.1
escape-html	1.0.3
escape-string-regexp	4.0.0
eslint	7.32.0
eslint-config-prettier	7.2.0
eslint-plugin-es	3.0.1
eslint-plugin-node	11.1.0
eslint-plugin-prettier	3.4.1
eslint-scope	5.1.1
eslint-utils	3.0.0
eslint-visitor-keys	2.1.0
espree	7.3.1
esprima	4.0.1
esquery	1.5.0
esrecurse	4.3.0
estraverse	4.3.0
esutils	2.0.3
etag	1.8.1
events	1.1.1
execa	5.1.1
express	4.18.2
external-editor	3.1.0
fast-deep-equal	3.1.3
fast-diff	1.2.0
fast-glob	3.2.12

fast-json-stable-stringify	2.1.0
fast-levenshtein	2.0.6
fastq	1.15.0
figures	3.2.0
file-entry-cache	6.0.1
fill-range	7.0.1
finalhandler	1.2.0
find-up	4.1.0
flat-cache	3.0.4
flatted	3.2.7
follow-redirects	1.15.2
for-each	0.3.3
foreground-child	3.1.1
form-data	4.0.0
forwarded	0.2.0
fresh	0.5.2
fs.realpath	1.0.0
fsevents	2.3.2
function-bind	1.1.1
functional-red-black-tree	1.0.1
generate-function	2.3.1
get-caller-file	2.0.5
get-intrinsic	1.2.1
get-stream	6.0.1
glob	10.2.4
glob-parent	5.1.2
globals	13.20.0
globby	11.1.0
gopd	1.0.1
gts	3.1.1
hard-rejection	2.1.0
has	1.0.3
has-flag	3.0.0
has-flag	4.0.0
has-proto	1.0.1
has-symbols	1.0.3
has-tostringtag	1.0.0
hexer	1.5.0
hosted-git-info	4.1.0
http-errors	2.0.0
human-signals	2.1.0
iconv-lite	0.4.24
ieee754	1.1.13
ignore	5.2.4
ignore-by-default	1.0.1
import-fresh	3.3.0
imurmurhash	0.1.4
indent-string	4.0.0
inflight	1.0.6
inherits	2.0.4
inquirer	7.3.3
ipaddr.js	1.9.1

is-arguments	1.1.1
is-arrayish	0.2.1
is-binary-path	2.1.0
is-callable	1.2.7
is-core-module	2.12.0
is-extglob	2.1.1
is-fullwidth-code-point	3.0.0
is-generator-function	1.0.10
is-glob	4.0.3
is-number	7.0.0
is-plain-obj	1.1.0
is-property	1.0.2
is-stream	2.0.1
is-typed-array	1.1.10
is-typedarray	1.0.0
isarray	1.0.0
isexe	2.0.0
jackspeak	2.2.0
jaeger-client	3.19.0
jmespath	0.16.0
js-tokens	4.0.0
js-yaml	3.14.1
json-parse-even-better-errors	2.3.1
json-schema-traverse	0.4.1
json-stable-stringify-without-jsonify	1.0.1
json5	2.2.3
kind-of	6.0.3
levn	0.4.1
lines-and-columns	1.2.4
locate-path	5.0.0
lodash	4.17.21
lodash.camelcase	4.3.0
lodash.merge	4.6.2
lodash.truncate	4.4.2
long	4.0.0
lru-cache	6.0.0
make-error	1.3.6
map-obj	4.3.0
media-typer	0.3.0
meow	9.0.0
merge-descriptors	1.0.1
merge-stream	2.0.0
merge2	1.4.1
methods	1.1.2
micromatch	4.0.5
mime	1.6.0
mime-db	1.52.0
mime-types	2.1.35
mimic-fn	2.1.0
min-indent	1.0.1
minimatch	3.1.2
minimist	1.2.8

minimist-options	4.1.0
minipass	6.0.0
module-details-from-path	1.0.3
ms	2.0.0
ms	2.1.2
mute-stream	0.0.8
mysql2	2.3.0
named-placeholders	1.1.3
natural-compare	1.4.0
ncp	2.0.0
negotiator	0.6.3
node-int64	0.4.0
nodemon	2.0.22
nopt	1.0.10
normalize-package-data	3.0.3
normalize-path	3.0.0
npm-run-path	4.0.1
object-inspect	1.12.3
on-finished	2.4.1
once	1.4.0
onetime	5.1.2
opentracing	0.14.7
optionator	0.9.1
os-tmpdir	1.0.2
p-limit	2.3.0
p-locate	4.1.0
p-try	2.2.0
parent-module	1.0.1
parse-json	5.2.0
parseurl	1.3.3
path-exists	4.0.0
path-is-absolute	1.0.1
path-key	3.1.1
path-parse	1.0.7
path-scurry	1.9.1
path-to-regexp	0.1.7
path-type	4.0.0
picomatch	2.3.1
prelude-ls	1.2.1
prettier	2.8.8
prettier-linter-helpers	1.0.0
process	0.10.1
progress	2.0.3
protobufjs	7.2.3
proxy-addr	2.0.7
proxy-from-env	1.1.0
pstree.remy	1.1.8
punycode	1.3.2
punycode	2.3.0
qs	6.11.0
querystring	0.2.0
queue-microtask	1.2.3

quick-lru	4.0.1
range-parser	1.2.1
raw-body	2.5.2
read-pkg	5.2.0
read-pkg-up	7.0.1
readdirp	3.6.0
redent	3.0.0
regexpp	3.2.0
require-directory	2.1.1
require-from-string	2.0.2
require-in-the-middle	5.2.0
require-in-the-middle	7.1.0
resolve	1.22.2
resolve-from	4.0.0
restore-cursor	3.1.0
reusify	1.0.4
rimraf	5.0.0
run-async	2.4.1
run-parallel	1.2.0
rxjs	6.6.7
safe-buffer	5.2.1
safer-buffer	2.1.2
sax	1.2.1
semver	7.5.1
send	0.18.0
seq-queue	0.0.5
serve-static	1.15.0
setprototypeof	1.2.0
shebang-command	2.0.0
shebang-regex	3.0.0
shimmer	1.2.1
side-channel	1.0.4
signal-exit	3.0.7
simple-update-notifier	1.1.0
slash	3.0.0
slice-ansi	4.0.0
spdx-correct	3.2.0
spdx-exceptions	2.3.0
spdx-expression-parse	3.0.1
spdx-license-ids	3.0.13
sprintf-js	1.0.3
sqlstring	2.3.3
statuses	2.0.1
string-template	0.2.1
string-width	4.2.3
string-width-cjs	npm:string-width@4.2.3
strip-ansi	6.0.1
strip-ansi-cjs	npm:strip-ansi@6.0.1
strip-final-newline	2.0.0
strip-indent	3.0.0
strip-json-comments	3.1.1
supports-color	5.5.0

supports-color	7.2.0
supports-preserve-symlinks-flag	1.0.0
table	6.8.1
text-table	0.2.0
thriftrw	3.12.0
through	2.3.8
tmp	0.0.33
to-regex-range	5.0.1
toidentifier	1.0.1
touch	3.1.0
trim-newlines	3.0.1
ts-node	10.9.1
tsc	2.0.4
tslib	1.14.1
tsutils	3.21.0
type-check	0.4.0
type-fest	0.20.2
type-is	1.6.18
typedarray-to-buffer	3.1.5
typescript	4.9.5
typescript	5.0.4
undefsafe	2.0.5
unpipe	1.0.0
uri-js	4.4.1
url	0.10.3
util	0.12.5
utils-merge	1.0.1
uuid	8.0.0
v8-compile-cache	2.3.0
v8-compile-cache-lib	3.0.1
validate-npm-package-license	3.0.4
vary	1.1.2
which	2.0.2
which-typed-array	1.1.9
word-wrap	1.2.3
wrap-ansi	7.0.0
wrap-ansi	8.1.0
wrap-ansi-cjs	npm:wrap-ansi@7.0.0
wrappy	1.0.2
write-file-atomic	3.0.3
xml2js	0.5.0
xmlbuilder	11.0.1
xorshift	1.2.0
xtend	4.0.2
y18n	5.0.8
yallist	4.0.0
yargs	17.7.2
yargs-parser	20.2.9
yargs-parser	21.1.1
yn	3.1.1

Python

名前	バージョン
deprecated	1.2.13
asgiref	3.6.0
backoff	2.2.1
certifi	2023.5.7
charset-normalizer	3.1.0
googleapis-common-protos	1.59.0
idna	3.4
importlib-metadata	6.0.1
opentelemetry-api	1.17.0
opentelemetry-distro	0.38b.0
opentelemetry-distro	0.38b0
opentelemetry-exporter-otlp-proto-http	1.17.0
opentelemetry-instrumentation	0.38b0
opentelemetry-instrumentation-aiohttp-client	0.38b0
opentelemetry-instrumentation-asgi	0.38b0
opentelemetry-instrumentation-asyncpg	0.38b0
opentelemetry-instrumentation-boto	0.38b0
opentelemetry-instrumentation-botocore	0.38b0
opentelemetry-instrumentation-celery	0.38b0
opentelemetry-instrumentation-dbapi	0.38b0
opentelemetry-instrumentation-django	0.38b0
opentelemetry-instrumentation-elasticsearch	0.38b0
opentelemetry-instrumentation-falcon	0.38b0
opentelemetry-instrumentation-fastapi	0.38b0
opentelemetry-instrumentation-flask	0.38b0
opentelemetry-instrumentation-grpc	0.38b0
opentelemetry-instrumentation-jinja2	0.38b0
opentelemetry-instrumentation-mysql	0.38b0
opentelemetry-instrumentation-psycopg2	0.38b0
opentelemetry-instrumentation-pymemcache	0.38b0
opentelemetry-instrumentation-pymongo	0.38b0
opentelemetry-instrumentation-pymysql	0.38b0
opentelemetry-instrumentation-pyramid	0.38b0
opentelemetry-instrumentation-redis	0.38b0
opentelemetry-instrumentation-requests	0.38b0
opentelemetry-instrumentation-sqlalchemy	0.38b0
opentelemetry-instrumentation-sqlite3	0.38b0
opentelemetry-instrumentation-starlette	0.38b0
opentelemetry-instrumentation-tornado	0.38b0
opentelemetry-instrumentation-wsgi	0.38b0
opentelemetry-propagator-aws-xray	1.0.1
opentelemetry-proto	1.17.0
opentelemetry-sdk	1.17.0
opentelemetry-semantic-conventions	0.38b0
opentelemetry-util-http	0.38b0
protobuf	4.23.0
requests	2.30.0
setuptools	67.7.2
typing_extensions	4.5.0

urllib3	2.0.2
wrapt	1.15.0
zipp	3.15.0

Contrast Serverless のサポート対象の言語

Contrast Serverless では、以下のプログラミング言語をサポートしています。

言語	ランタイムバージョン
Java	8.x、11.x
.NET	.NET 5.x、6.x .NET Core 3.1
Node.js	12.x、14.x、16.x、18.x
Python	3.6, 3.7, 3.8, 3.9

Contrast Serverless のサポート対象のプラットフォーム

Contrast Serverless では、以下のプラットフォームをサポートしています。

- AWS Lambda
- Microsoft Azure

マルチリージョンのサポート

Contrast Serverless はマルチリージョンをサポートし、1 つの AWS アカウント内で異なるリージョンにアプリケーションをデプロイするお客様を支援するようになりました。

これにより、同じ AWS アカウント内の複数のリージョンにエージェントをオンボードすることができます。また、アカウントとリージョンの組み合わせごとに、アカウントとリージョンを個別に表示・管理ができます。

注記
サポート対象の全てのリージョンを 1 つの AWS アカウントに追加できます。

サポート対象のリージョン

Contrast Serverless では、以下のリージョンをサポートします。

サーバレス環境名	サイト	サポート対象のリージョン

AppUS	`https://cs001.contrastsecurity.com`
	`https://app.contrastsecurity.com`
	`https://eval.contrastsecurity.com`
	`https://security-research.contrastsecurity.com`
	`https://ce.contrastsecurity.com`
		`us-east-1`
		`us-east-2`
		`eu-west-1`
		`eu-west-2`
		`eu-central-1`
		`eu-north-1`
		`ap-northeast-1`
		`ap-northeast-3`
		`ap-southeast-1`
		`us-west-1`
		`us-west-2`
		`ca-central-1`

AppTWO	`https://apptwo.contrastsecurity.com`	`us-east-1`
		`eu-central-1`
		`eu-west-1`

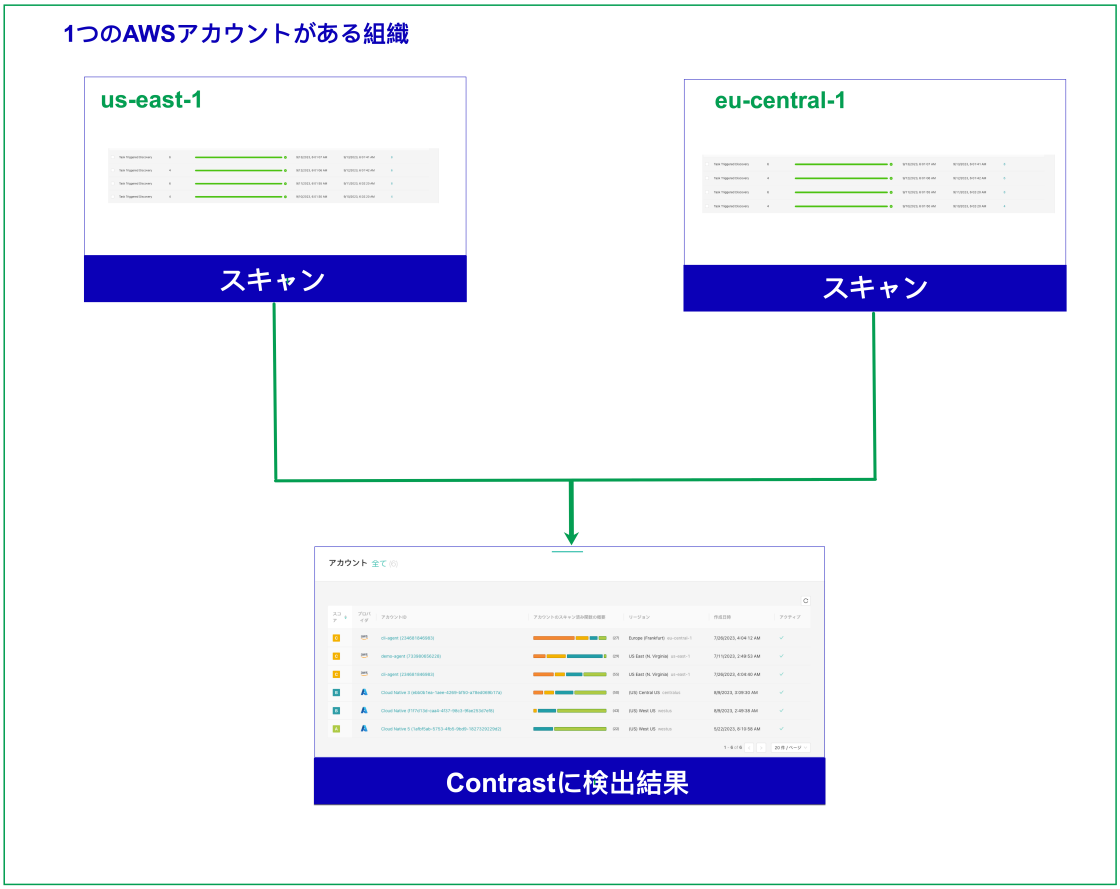
サーバレス環境名	サイト	サポート対象のリージョン
AppJAPAN	`https://app.contrastsecurity.jp`	`ap-northeast-1` `ap-northeast-3`
AppUK	`https://cs002.contrastsecurity.com`	`eu-west-2`

AppEU	`https://eval003.contrastsecurity.com`
	`https://cs003.contrastsecurity.com`
		`eu-west-1`
		`eu-west-2`
		`eu-central-1`
		`us-east-1`

ステージング (dev1eu)	`https://teamserver-staging.contsec.com`	`eu-central-1`
		`us-east-1`
		`eu-west-1`

使い方

- 1 つの組織に対して 1 つの AWS アカウントで、最初のリージョンをオンボードします。そして、次のリージョンをオンボードします。
  例 : ある開発担当は、*us-east-1* でアカウントをオンボードし、別の開発担当が使用しているのと同じアプリケーション(ただし、モジュール/コードパッケージは異なる)がある *eu-central-1* でもオンボードします。
- 各リージョンを設定したら、通常通りスキャンを実行し、Contrast で検出結果やグラフを確認できます。
  - Contrast Web インターフェイスでは、リージョンの場所に基づいてオンボードされたアカウントがそれぞれ表示されます。
  - アカウントの詳細(リージョンの情報、アカウント内の関数の数、深刻度など)は、通常通り、Contrast Web インターフェイスで表示されます。
  - また、チーム/リージョンで使用されている関数(タグ付けに基づいて)の管理や、異なるリージョンに関係なくアプリケーションの全ての関数をグラフで表示する機能も引き続き使用できます。



インベントリ

Contrast から AWS アカウントに接続すると、対象の環境内のすべての Lambda 関数と、さまざまなリソース(S3、API Gateway、DynamoDB など)との関係が自動的に検出されます。

特定の基準を指定 (654ページ)しない限り、デフォルトではインベントリのすべての関数がスキャンされます。

インベントリの基準

Contrast ではスキャンの対象とする関数を決定するのに、次のような基準を指定できます。

- **Tag：** このオプションを使用すると、指定したタグ、およびタグの値に関連付けられた関数を含めたり除外したりできます。
- **Name：** このオプションを使用すると、関数の名前の一致(または前方一致や後方一致)で対象に含めたり除外したりできます。

スキャンの種類と監視について

Contrast Serverless では、次の種類のスキャンをサポートします。

静的スキャン

このスキャンでは、関連する静的コードや設定評価をほぼリアルタイムで自動的にスキャンし、以下のカテゴリにおいて新たな脆弱性を検出します。

- 最小権限：デプロイ前のサーバレスワークロード内の IAM の脆弱性(過剰に権限が設定されている関数)を検出し、推奨するアクセス許可を修正案として表示します。
- Contrast SCA：Contrast SCA エンジンを使用して、オープンソースライブラリの SCA を行います。

このスキャンによる、コードへの永久的な影響はありません。

動的スキャン

このスキャンでは、検査対象の環境で発生する特定の更新に基づいて動的な検査を行います。

これは、検査対象の環境で生じた特定の更新に合わせて、ほぼリアルタイムで自動的に実行され、動的な検査が行われます。動的スキャンは、OWASP Top 10 の判定を基準としています。例：

- SQL インジェクション
- コードインジェクション
- ローカルファイルインクルード(LFI)

動的スキャンでは、Contrast は悪意のあるデータを送信して関数の実行を試行し、脆弱性を検出します。この動作はコードに影響を与えませんが、スキャン対象の関数は呼び出されます。

IAST 解析

注記

IAST 解析を実行する場合は、*テストカバレッジ*を使用する必要があります。

スキャンの設定を指定する際に、このオプションを選択することを推奨します。

IAST 解析により、AWS アカウントで悪用可能な全ての AWS Lambda 関数が明らかになります。最新の AWS Lambda サービスに対するサポートにより、AWS Step Functions(複数の Lambda 関数を柔軟なワークフローに調整するサービス)のセキュリティの問題を明らかにすることができます。

以下のような OWASP トップ 10 の脆弱性を検出できます。

- コンテンツインジェクション、OS コマンドインジェクション、 SQL インジェクション(限定的)、コードインジェクション
- クロスサイトスクリプティング(XSS)
- ローカルファイルインクルード(LFI)

さらに、*未使用の関数*(シャドウ関数)を含め、サーバレスアカウントの全ての資産が明らかになります。これにより、AWS アカウントの観測性が向上し、セキュリティカバレッジが広がります。未使用の関数は、通常のメンテナンスがされておらず、脆弱性につながる古い依存関係が含まれている可能性があります。

継続的な監視

Contrast からアカウントに接続すると、Contrast Serverless がこのアカウントを監視します。関数のコードや設定が変更されると、Contrast は自動的に新しいスキャンを開始します。

AWS で Contrast Serverless を使い始める

Contrast Serverless の使用を開始するには、Contrast Web インタフェースを開き、AWS アカウントに接続して、新しいスタックを作成します。

開始する前に

- AWS アカウントの情報を準備します。
- Contrast Serverless のスタックをデプロイ/更新/削除するために最小限の権限 (642ページ)が必要です。

手順

1. Contrast Web インタフェースのページ上部の**新規登録**を選択します。

2. **サーバレス**のカードを選択します。

3. **クラウドプロバイダ**のセクションで **AWS** を選択します。
4. 必要に応じて、スキャンの設定をします。
   - **インベントリ：** インベントリにはスキャンの対象とする関数を指定します。
     デフォルトでは、AWS アカウントのすべての関数が対象になります。
   - **初回スキャン：** この設定には、関数をスキャンするアクションを指定します。

静的解析	動的解析
対象： • **最小権限：** 使用されていない権限を検出します。Java、.NET Core 6、.NET Core 7、Node.js、Python が対象です。 • **CVE：** 脆弱な OSS ライブラリを検出します。Java、.NET Core 6、.NET Core 7、Node.js、Python が対象です。 • **SAST：** カスタムコードの脆弱性を検出します。 Java が対象です。 • **マルウェア:** 悪意のあるファイルを検出します。Python が対象です。	対象： • アプリケーションのストレステストを行い、潜在的な脆弱性を検出します。 • **IAST 解析**オプションを使用すると、アカウント環境全体および全てのサービスにおいて、関数の脆弱性を見つけることができます。詳細は、スキャンの種類と監視について (640ページ)を参照してください。この解析機能をアカウントに完全に設定するには、*IAST 解析の手順* セクションにある手順に従ってください。Node.js と Python が対象です。

   - **Deployment：** AWS で新しいスタックにデプロイするか、パイプラインで使用する CFT をダウンロードします。

   上記の設定は、「設定」タブでいつでも設定の変更 (655ページ)ができます。
5. **Create new stack** を選択します。
6. 表示される AWS の画面で、アカウント情報を入力し**スタックの作成**を選択します。 または、AWS CloudFormation のテンプレートをダウンロードして、開発パイプラインで使用することもできます。
   この操作により、アカウントの AWS CloudFormation スタックコンソールに接続し、最初のスキャンが開始します。
7. AWS のコンソールでスタックのデプロイを承認します。
   スタックのデプロイには、完了するまでに約 2 分かかります。
8. Contrast Web インタフェースに戻り、アカウント接続とスキャン開始のメッセージが表示されることを確認します。
9. 関数の詳細やスキャンの結果を見るには、「アカウントに接続」のメッセージ内にある**関数**を選択して表示するか、ナビゲーションバーから**サーバレス**タブを選択します。

次の手順

- オンデマンドで関数をスキャン (648ページ)
- 結果の表示 (649ページ)
- インベントリ基準の変更 (654ページ)
- スキャン設定の変更 (655ページ)

Contrast Serverless を実行するための AWS ポリシーと権限

ここでは、Contrast Serverless を実行するための AWS アカウントのポリシーと権限のサンプルを紹介します。

アクセス管理の例

以下は、アカウントの権限とポリシーのサンプルです。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CustomResources",
            "Effect": "Allow",
            "Action": [
```

```
                "sns:Publish"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SNS2",
            "Effect": "Allow",
            "Action": [
                "sns:CreateTopic",
                "sns:GetTopicAttributes",
                "sns:DeleteTopic"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreatePolicy",
                "iam:CreateRole",
                "iam:CreateServiceLinkedRole",
                "iam:DeletePolicy",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetPolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListPolicyVersions",
                "iam:ListRoleTags",
                "iam:PassRole",
                "iam:PutRolePolicy",
                "iam:TagRole",
                "iam:UntagRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3",
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "s3:DeleteBucketPolicy",
                "s3:GetBucketPolicy",
                "s3:PutBucketPolicy",
                "s3:PutBucketPublicAccessBlock",
                "s3:PutBucketTagging",
                "s3:PutEncryptionConfiguration",
                "s3:PutLifecycleConfiguration"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "Lambda",
            "Effect": "Allow",
            "Action": [
                "lambda:GetFunction",
                "lambda:CreateFunction",
                "lambda:DeleteFunctionEventInvokeConfig",
                "lambda:DeleteFunction",
                "lambda:TagResource",
                "lambda:PutFunctionEventInvokeConfig"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3LambdaCode",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EventsRule",
            "Effect": "Allow",
            "Action": [
                "events:DeleteRule",
                "events:DescribeRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CloudTrail",
            "Effect": "Allow",
            "Action": [
                "cloudtrail:AddTags",
                "cloudtrail:CreateTrail",
                "cloudtrail:DeleteTrail",
                "cloudtrail:StartLogging",
                "cloudtrail:PutEventSelectors"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS の `iam create-policy` を実行：

```
aws iam create-policy --policy-name Contrast-create-stack --policy-
document '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CustomResources",
      "Effect": "Allow",
```

```
      "Action": ["sns:Publish"],
      "Resource": "*"
    },
    {
      "Sid": "SNS2",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:GetTopicAttributes",
        "sns:DeleteTopic"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IAM",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:CreatePolicy",
        "iam:CreateRole",
        "iam:CreateServiceLinkedRole",
        "iam:DeletePolicy",
        "iam:DeleteRole",
        "iam:DeleteRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetPolicy",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:ListPolicyVersions",
        "iam:ListRoleTags",
        "iam:PassRole",
        "iam:PutRolePolicy",
        "iam:TagRole",
        "iam:UntagRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3",
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucket",
        "s3:DeleteBucketPolicy",
        "s3:GetBucketPolicy",
        "s3:PutBucketPolicy",
        "s3:PutBucketPublicAccessBlock",
        "s3:PutBucketTagging",
        "s3:PutEncryptionConfiguration",
        "s3:PutLifecycleConfiguration"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Lambda",
```

```
      "Effect": "Allow",
      "Action": [
        "lambda:GetFunction",
        "lambda:CreateFunction",
        "lambda:DeleteFunctionEventInvokeConfig",
        "lambda:DeleteFunction",
        "lambda:TagResource",
        "lambda:PutFunctionEventInvokeConfig"
      ],
      "Resource": "*"
    },
    {
      "Sid": "S3LambdaCode",
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": "*"
    },
    {
      "Sid": "EventsRule",
      "Effect": "Allow",
      "Action": [
        "events:DeleteRule",
        "events:DescribeRule",
        "events:PutRule",
        "events:PutTargets",
        "events:RemoveTargets"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudTrail",
      "Effect": "Allow",
      "Action": [
        "cloudtrail:AddTags",
        "cloudtrail:CreateTrail",
        "cloudtrail:DeleteTrail",
        "cloudtrail:StartLogging",
        "cloudtrail:PutEventSelectors"
      ],
      "Resource": "*"
    }
  ]
}
```

レスポンスを取得：

ユーザポリシーをアタッチ：

```
aws iam attach-user-policy --policy-arn
arn:aws:iam::402181209224:policy/Contrast-serverless-create-stack --user-
name
<USER-NAME></USER-NAME>
```

レスポンスを取得：

これで、デプロイメントを実行できるようになりました。

Azure で Contrast Serverless を使い始める

Azure で Contrast Serverless の使用を開始するには、Contrast Web インタフェースを開き、Azure アカウントに接続して、新しいスタックを作成します。

開始する前に

- 有効なサブスクリプションを持つ Azure アカウントがあること。
- Active Directory テナントにアプリ登録を作成できる権限があること。
- アカウントに*所有者*ロールがあること。これによりアプリ登録にロールを割り当てることができます。

手順

1. Contrast Web インタフェースのページ上部の**新規登録**を選択します。

2. **サーバレス**のカードを選択します。

3. **クラウドプロバイダ**のセクションで **Azure** を選択します。
4. 必要に応じて、スキャンの設定をします。
   - **インベントリ**： Azure では利用できません。
   - **初回スキャン**： この設定には、関数をスキャンするアクションを指定します。

静的解析	動的解析
内容： • **最小権限**- 使用されていない権限を検出します。Java、.NET Core 6、.NET Core 7、Node.js、Python が対象です。 • **CVE** - 脆弱な OSS ライブラリを検出します。Java、.NET Core 6、.NET Core 7、Node.js、Python が対象です。 • **SAST** - カスタムコードの脆弱性を検出します。Java が対象です。 • **マルウェア** - 悪意のあるファイルを検出します。Python が対象です。	Azure では利用できません。

   上記の設定は、「設定」タブでいつでも設定の変更 (655ページ)ができます。
5. **Deployment** セクションの手順を続けます。
6. Contrast Web インタフェースに戻り、アカウント接続とスキャン開始のメッセージが表示されることを確認します。

次の手順

- オンデマンドで関数をスキャン (648ページ)
- 結果の表示 (649ページ)

オンデマンドで関数をスキャン

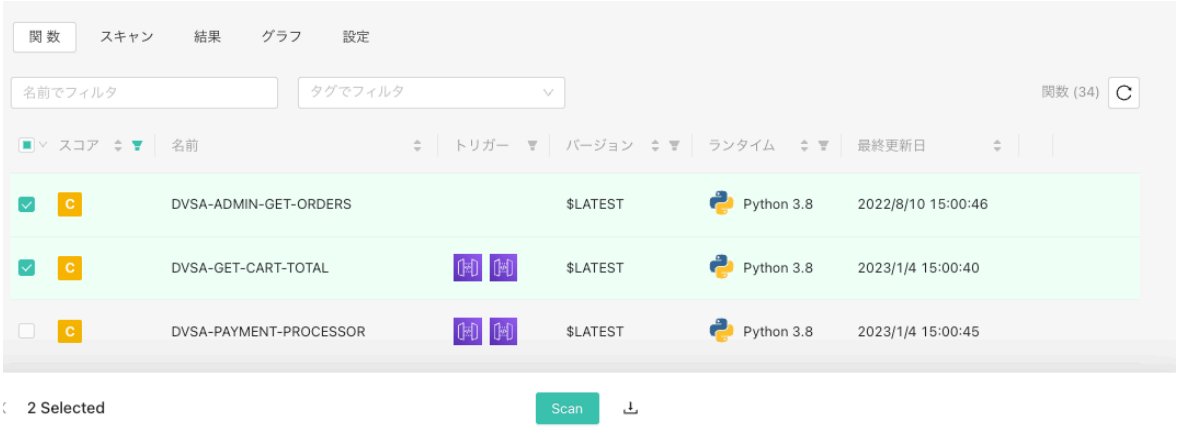
アカウントの全ての関数のスキャンは自動的に行われますが、必要に応じて特定の関数をスキャンすることもできます。

開始する前に

- スキャンを行いたい関数を特定してください。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. 一覧からアカウントを選択します。
3. スキャンする関数(1 つまたは複数)の横にあるチェックボックスを選択します。



4. **Scan** を選択します。
5. 「スキャンを確認」の画面で、選択した関数に対するスキャンの種類を確認して、**OK** を選択します。
   スキャン開始のメッセージが表示されます。

6. スキャンの結果を確認するにはスキャン開始のメッセージにある**スキャンを表示**を選択します。
   または、**スキャン**タブで **Ad Hoc Scan** の行を選択すると、結果を確認できます。
   スキャンした関数で同じ脆弱性が複数検出された場合、新たな脆弱性として報告されるのではなく、報告済みの既存の脆弱性が新しいデータ(タイムスタンプなど)で更新されます。

関数タブの情報

関数の一覧には、次の情報があります。

- **スコア**：関数のコンテキストリスクスコア (659ページ)
- **名前**：関数名
- **トリガー**：イベントを発生させたサービス
- **バージョン**：関数のバージョン
- **ランタイム**：ランタイム言語
- **最終更新日**：関数が最後に更新された日時
- **最終スキャン**：スキャンが最後に実行された日時
- **問題**：スキャン中に検出された、注意が必要な項目と注意を必要としない項目。ここでの検出結果は、Contrast Serverless と AWS Inspector から生成されたソースを参照しています。Contrast Serverless と AWS Inspector からの結果をサポートしているお客様のみに表示されます。

結果の表示

結果を表示して、権限、依存関係、攻撃、CVE などの脆弱性に関する情報を確認することができます。

開始する前に

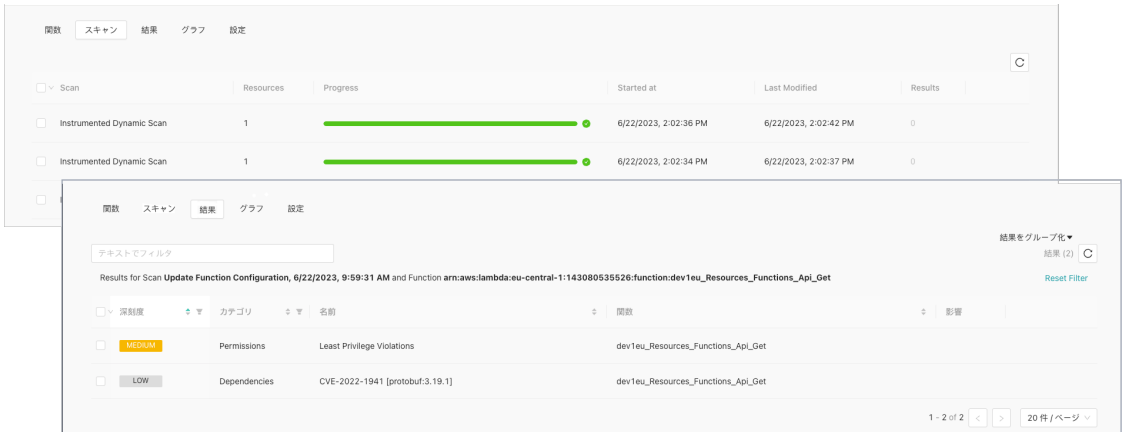
- 少なくとも 1 つのスキャンが完了していること。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. すべてのスキャンの結果を表示するには、**結果**タブを選択します。スキャン結果の意味の詳細については、スキャンのステータス情報 (652ページ)を参照してください。
3. 結果タブでフィルタをかけるには、列のヘッダの横にある**フィルタ**アイコン( )を選択します。
   a. **深刻度**は、アプリケーションの脆弱性に基づいています。深刻度のレベルについては、アプリケーションのスコアガイド (958ページ)を参照してください。
   b. **カテゴリ**は、関数の種類ごとの脆弱性に基づいています。
   c. **関数**は、アカウントで検知された関数に基づいています。
   画面右側の**結果をグループ化**でオプションを選択すると、結果を**カテゴリ(Category)**や**関数(Function)**でグループ化することもできます。
   d. **ソース**は、結果が提供されるプラットフォームに基づいています。Contrast からか、AWS Inspector からのいずれかです。アイコンをクリックすると、結果の詳細情報が表示されます。AWS Inspector の結果は、アカウントで AWS Inspector をご利用の場合のみに表示されることに注意してください。

   フィルタを解除するには列のヘッダの横にある緑色の**フィルタ**アイコン( )を選択し、**リセット**を選択します。
4. 結果タブで関数を検索するには、検索フィールドに検索ワードを入力します。

5. 1 つのスキャンの結果を表示するには、スキャンタブにアクセスして、スキャンの行を選択します。スキャンの行を選択すると、スキャンの詳細ページが表示されます。
6. 関数の検出結果の詳細を表示するには：
   a. 結果タブからは、一覧で該当の行を選択します。
   b. スキャンの詳細ページからは、関数の結果列にある数字をクリックします。



結果の詳細ページは、以下の例のようになります。

結果の詳細

ここに表示される結果の詳細は、脆弱性のカテゴリによって異なります。

全ての脆弱性で、以下の情報が表示されます。

- **説明**：脆弱性についての概要。
- **何が起こったか**：スキャンで脆弱性が検出された時の状況。
- **対応策**：脆弱性を修正するための手順。

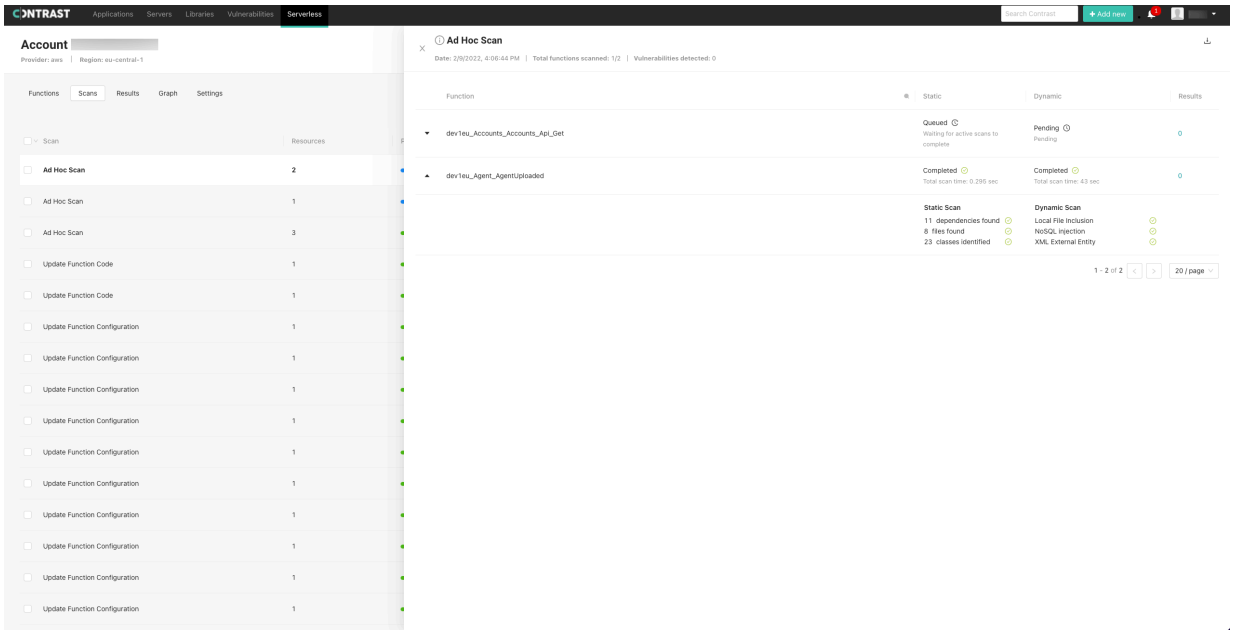
以下の情報も表示される場合があります。

- **違反しているポリシー**：脆弱性が違反しているポリシー。
- **影響**：脆弱性の影響を受ける領域。例、SES、S3、Logs、DynamoDB など。
- **深刻度と評価基準**：脆弱性に対して算出された深刻度のスコア、および脆弱性が影響を与える特性を評価するための基準。

スキャンのステータス情報

サーバレスの Scans タブでは、システムがスキャンしている内容と、関数が検査されたスキャンのタイプについて確認できます。

Scans ページで、 スキャン名をクリックすると、スキャンのステータスの詳細タブが開きます。



以下は、静的スキャンと動的スキャンのステータスに関する情報です。

ステータス	説明
Scanning…..	スキャン中
Pending	静的スキャンの結果を保留中
Queued	X 個のアクティブなスキャンを完了待ち X は、アクティブなスキャンの数になります。スキャンはキューに入っており、まもなく開始されます。
Completed	スキャン完了
Unsupported	サポート対象外の Lambda ランタイム 関数のランタイム言語がサポート対象外です。
	サポート対象外の Lambda トリガー 関数にサポート対象外のトリガー設定があるか、識別できないトリガー設定がある。
Excluded	設定 (655ページ)でスキャンが無効になっている スキャンするには、インベントリの設定を変更するか、関数でアドホックスキャン(Ad Hoc Scan)を実行します。
Canceled	より新しいスキャンが開始 この関数の新しいバージョンが既にキューに入っています。
	Lambda のステータスが非アクティブ

ステータス	説明
	Lambda が 5 レイヤーの制限に達した Contrast では、上限値である 5 つのレイヤーがある関数はスキャンできません。
	Lambda のスキャンが既に進行中
	Lambda の最終ステータスの更新に失敗
Failed	エージェントを確認できない
	環境変数を復号化/暗号化できない
	エージェントのエラー Contrast の Cloud Agent 関数が起動できないか、静的解析中にスキャンが失敗しています。
	エージェントが変更された
	Lambda ハンドラーの設定ミス
	解析エラー

- 矢印アイコンをクリックして詳細を展開すると、スキャンで検出された依存関係、ファイル、クラスの情報が表示されます。
- 「Results」列の下の数字をクリックすると、Results (649ページ)タブが開きます。

サーバレスのスキャン結果のダウンロード

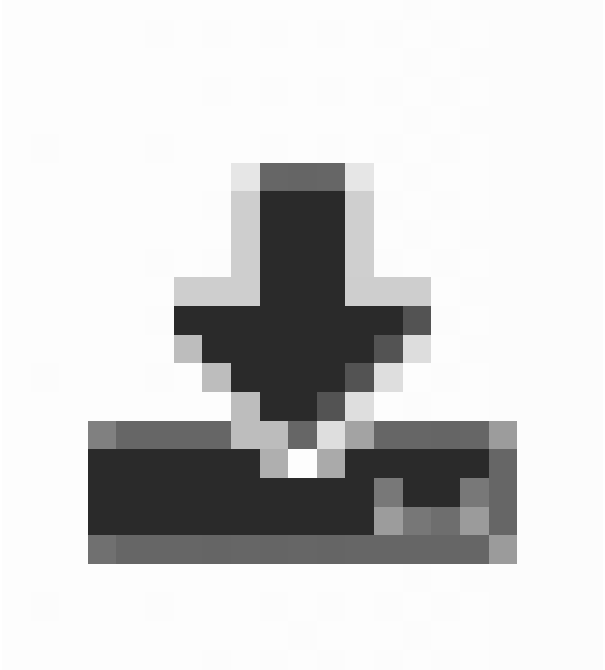
スキャンが完了したら、結果を CSV ファイルでダウンロードできます。

開始する前に

- 結果をダウンロードするスキャンを決めます。

手順

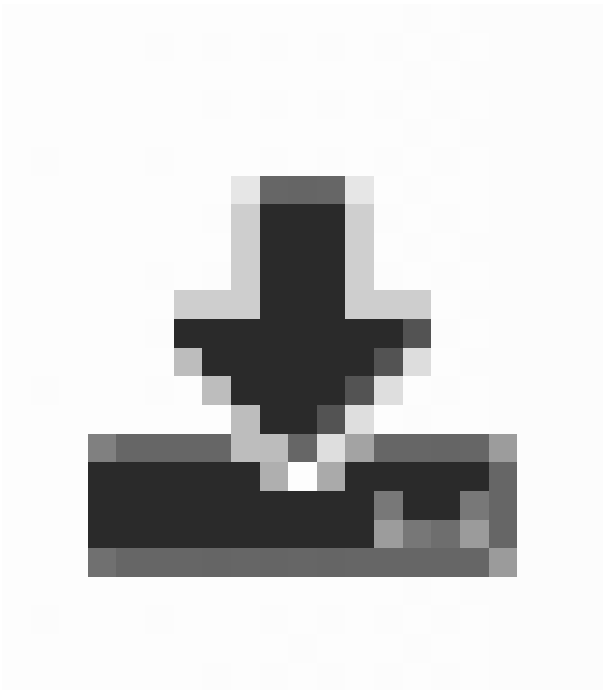
1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. **Scans** タブまたは **Results** タブを選択します。
3. **Scans** タブからダウンロードするには、スキャンの行の最後に表示されるダウンロードアイコンを選択します。
4. **Results** タブからダウンロードするには:
   a. スキャンの行の最後に表示されるダウンロードアイコン



( )を選択します。

または

b. スキャンの行をクリックし、詳細ページの右上にあるダウンロードアイコン



( )を選択します。

結果が CSV ファイルでダウンロードされます。

インベントリ基準の変更

インベントリ基準を登録することにより、Contrast Serverless でスキャンされる関数を指定できます。デフォルトでは、AWS アカウント内で検知されたすべての関数がスキャンされます。

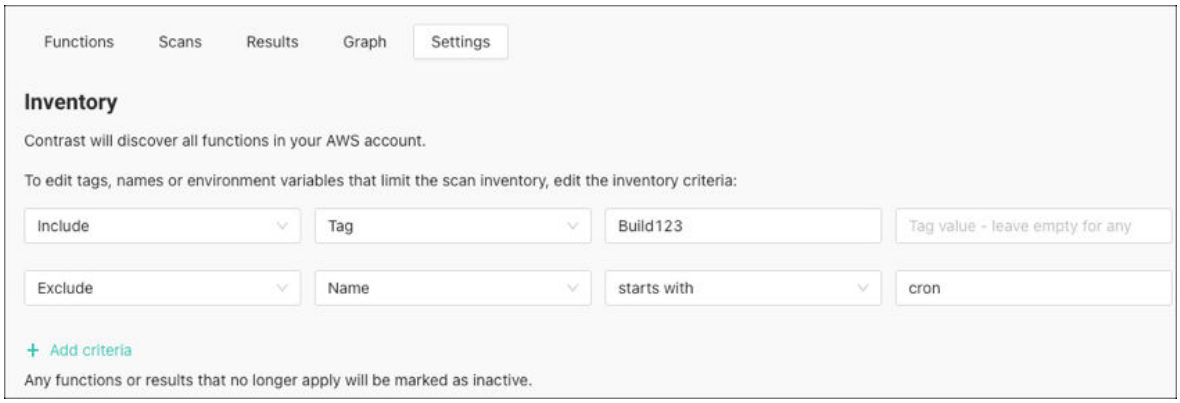
関数を除外するように指定すると、その関数はインベントリやスキャンの対象から外されます。

開始する前に

- スキャンの対象とする関数、または対象外とする関数を決めます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. **Settings** タブを選択します。
3. 「Inventory」でインベントリ基準を指定します。
   - 関数に関連付けられたタグ名を登録することによって、関数をスキャン対象に含むか対象外とするかを指定します。必要に応じて、タグの値も指定します。
   - 関数の名前によって、スキャン対象に含むか、対象外とするかを指定します。関数名を指定するためのオプションは、**Name is**(名前一致)、**Name starts with**(前方一致)、**Name ends with**(後方一致)より選択します。



4. **Save and Rescan** を選択します。
   新しいインベントリ基準で自動的に再スキャンが行われます。

サーバレスのスキャン設定の変更

このスキャン設定は、Contrast Serverless ですべての関数に対して実行されるスキャンの種類に影響します。

選択した関数を手動でスキャンするために、これらの設定を変更 (648ページ)できます。

開始する前に

- 静的スキャン、動的スキャンもしくは両方を使うか決めておくこと。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. **設定**タブを選択します。
3. **スキャン**のセクションで、使用するスキャンの種類を選択します。
   - **静的解析**： このスキャンでは、該当する静的コードと設定評価を調べて、脆弱性を検出します。
     静的スキャン中に、Contrast によって Lambda 関数がアカウントに追加されます。スキャンが完了すると、この関数は終了します。
   - **動的解析**： AWS アカウントのみが対象です。このスキャンでは、検査対象の環境で発生する特定の更新に基づく動的な検査を行います。
     動的スキャンでは、Contrast は悪意のあるデータを送信して関数の実行を試行し、脆弱性を検出します。
     IAST 解析に関する詳細は、スキャンの種類と監視について (640ページ)を参照してください。

   > **重要**
   > Contrast Serveless のスキャンによって、関数のコードが変更されることはありません。

4. **保存**を選択します。

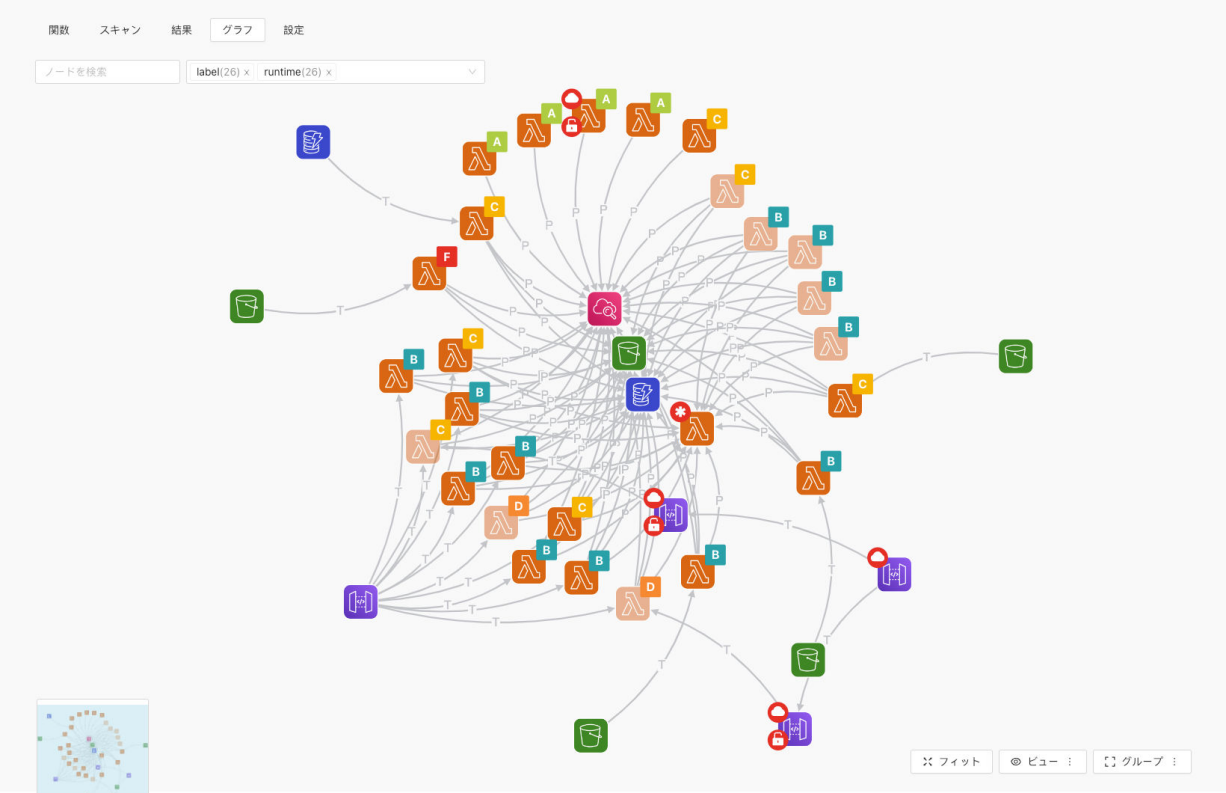
関数とサービスの関係の表示

「グラフ」タブでは、関数とサービスの関係を表す図が表示されます。図の各要素について、以下のような情報も参照できます。

- タグ
- 脆弱性の検出結果
- 権限

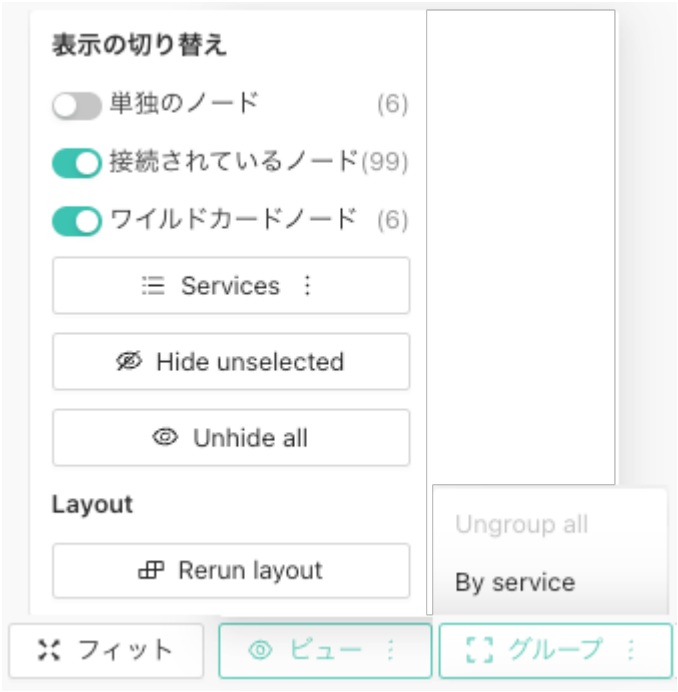
- セキュリティ状況を表すスコア：セキュリティ状況のスコアは、関数のトリガー設定に基づいています。インターネットからアクセス可能なトリガーや、公開されたバケットや認証されていない API などの設定ミスは、低いスコアになります。
- IAST 解析による未使用の関数(シャドウ関数)

警告
このグラフで得られる結果には上限があります。詳細については、グラフの制限(658ページ)を参照してください。



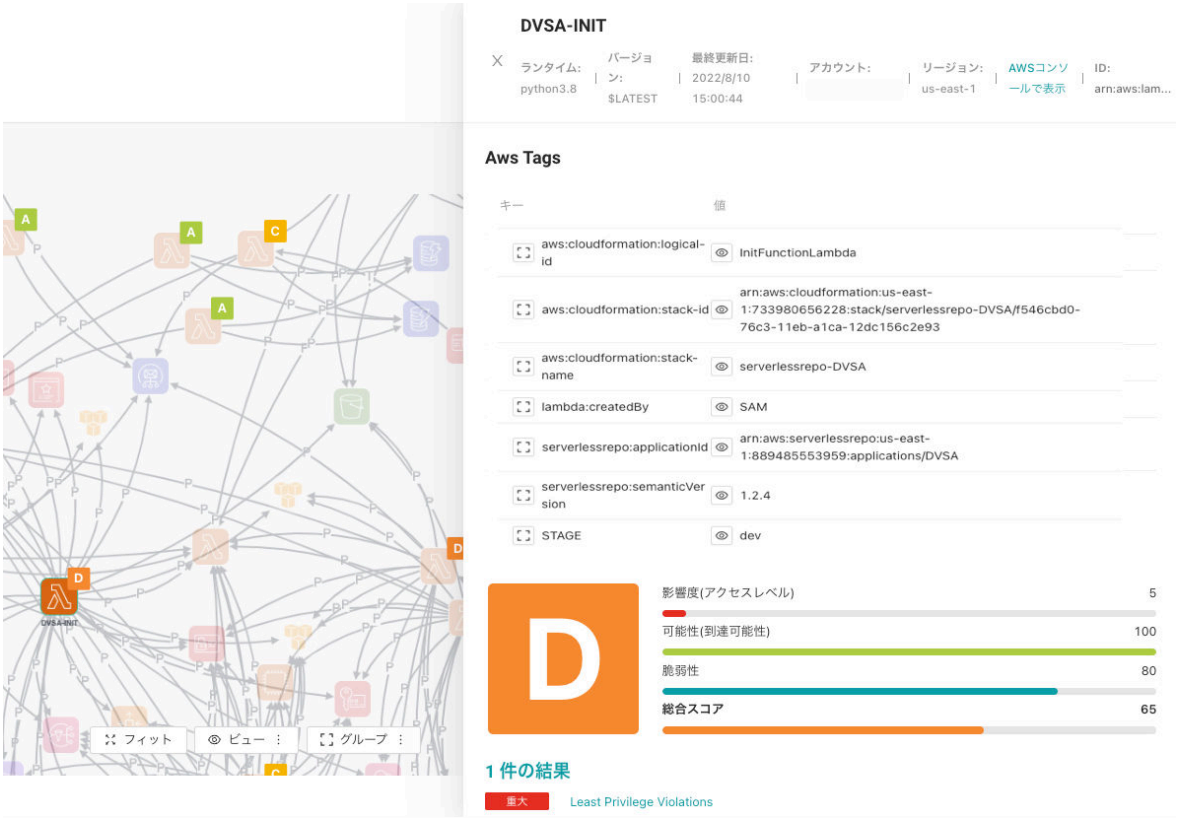
手順

1. Contrast Web インタフェースのナビゲーションバーで**サーバレス**を選択します。
2. **グラフ**タブを選択します。
3. 表示を調整するには、下部にあるオプションを使用します。



- **フィット**： 画面に合わせてグラフのサイズを変更できます。
- **ビュー**： ノードやサービスで表示にフィルタをかけることができます。ノードを表示· 非表示に切り替えることもできます。
- **グループ**： サービスをグループにまとめることができます。

4. 要素をクリックすると、各要素の詳細情報を参照できます。
選択した要素の詳細ウィンドウが表示されます。タグを選択すると、グラフの表示がフィルタされます。



グラフの制限

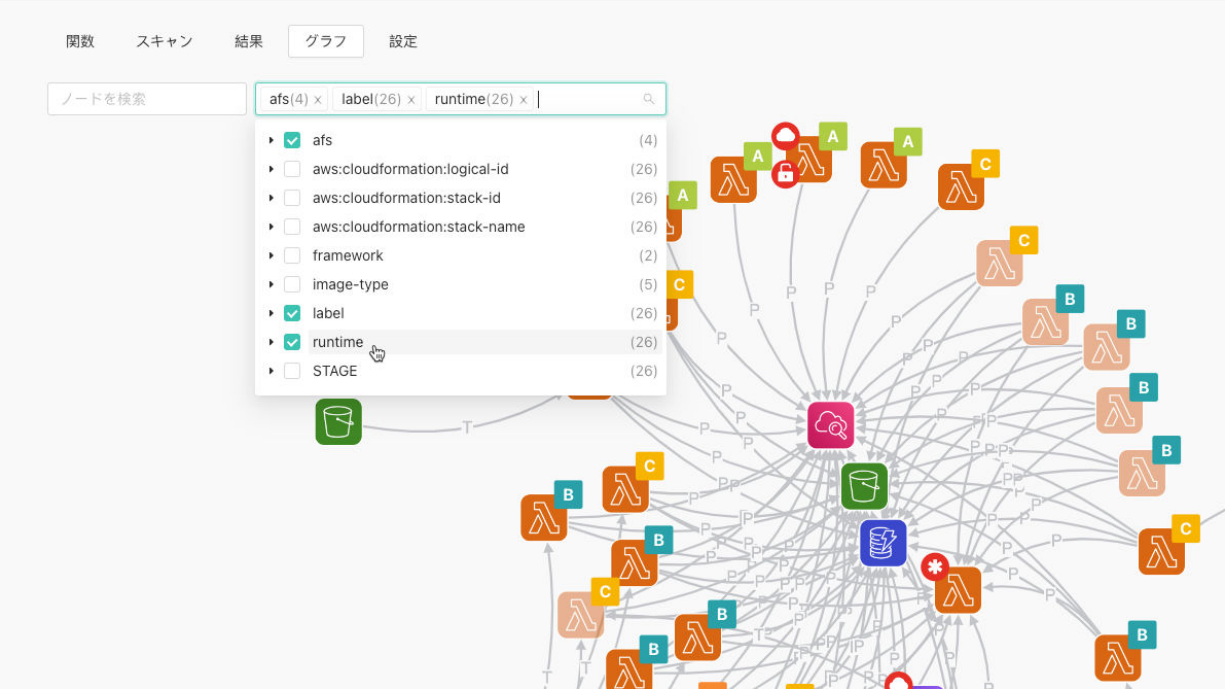
パフォーマンス上の考慮から、現在、要素数が特定の数(8000)を超えるとグラフのレンダリングを無効にしています。

要素数とは？

これは、製品の尺度ではなくパフォーマンスに関するものであるため、要素数は AWS/Azure リソースの数ではなく、レンダリング可能な要素の数になります。つまり、すべての**ノード**と**エッジ**(接続)を意味します。したがって、実際の要素数は各環境に固有のグラフ構造のみに依存するため、任意の一つの関数とその関連コンポーネントをレンダリングした場合に生じる具体的な要素数は予測することができません。

要素が多すぎてグラフがレンダリングされない場合は？

アカウントが大きすぎてもグラフをレンダリングできるように、リソースタグを使用したフィルタオプション(**タグフィルタ**)を追加しました。



フィルタを使用すると、返されるリソースの数が減り、選択したタグとその関連リソースを含む関数のみが表示されます。これによって要素の制限を超えないようになりますが、選択範囲が広すぎると同じ問題が発生する可能性があります。タグの選択は、**関数**タブと同じ方法で機能します。

重要

タグフィルタが、現在のところ、上限に対してカウントされるレンダリング要素の数を実際に制限できる唯一の方法です。特定の要素を検索したり非表示にしても、効果はありません。

タグの追加・更新

タグが付けられていない関数がある場合、タグフィルタを使用してそれらを表示することはできません。この機能を適切に利用するためには、システムの一部を絞り込めるように、異なる値をいくつか使用して、すべての関数にタグを付ける必要があります。考えられるタグとしては、`application`(アプ

リケーション)、service(サービス)、runtimeLanguage(ランタイム言語)、team(チーム)などがあります。また、その他必要に応じた独自のグループ化も利用できます。

AWS コンソールで直接追加するか、IaC ソリューションを使用するか、最適な方法を使用してください。

グラフに新しいタグが表示されるのはいつか？

TagResource のようなイベントの変更に基づいて、**関数**タブはほぼ即時に更新されますが、現在のところ、グラフの部分的な更新には対応しておらず、スケジュールされたフル検出のみによって実行されるため、即座にグラフには適用されません。通常、検出は午前 6 時(GMT)にスケジュールされています。つまり、次の検出が行われるまで、タグが表示されるのを待つ必要があります。

クラウドネイティブ開発者は、Resources_Utils_TriggerGraphDiscovery_V1 Lambda 関数を使用して、グラフ検出を早めにトリガーすることもできます。

あるいは、インベントリ設定 (654ページ)を変更すると、アカウント全体の**リソース**と**グラフ**の両方の検出がトリガーされます。インベントリ設定が空の場合、ダミーの **Exclude** の条件を追加すると変更とみなされますが、それでも全てが含まれることになります。

要約すると、次のいずれかの場合に新しいタグが表示されます。

- グラフ検出は、午前 6 時(GMT)にスケジュールに従って、毎日実行されます(一部の地域では時間設定が異なる場合があります)。
- Resources_Utils_TriggerGraphDiscovery_V1 Lambda 関数は、特定の組織 ID· アカウント ID でトリガーされます。
- インベントリ設定を変更します。

アカウントのインベントリ設定

オンボード中またはオンボード後に**設定**にて、**インベントリ設定**を変更して、システムで表示されるものやスキャンされる対象の範囲を制限することができます。ここでは、**関数**のタグと名前の両方を使うことができます。

ただし、これを変更すると、**関数**に表示される内容や実際にスキャンされたり保護される内容など、システム全体の範囲が変更されることに注意してください。要件によっては、グラフのサイズを制限する合理的な方法になる場合もあります。

コンテキストのリスクスコア

Contrast ではコンテキストに基づいてリスクをスコアで表しますが、これは、ユーザが実際のリスクポイントがどこにあり、何を優先すべきかを理解できるようにすることを目的としています。関数のコンテキストスコアは、各関数の全体的な評価を把握するのにに役立ちます。

関数には、関数の総合評価を表す **A** から **F** までのレターグレードと、35 から 100 までの数値スコアがあります。

- **A**： 90 - 100
- **B**： 80 - 89
- **C**： 70 - 79
- **D**： 60 - 69
- **F**： 35 - 59

総合のコンテキストのリスクスコアの計算：**平均 = (脆弱性スコア + 影響度スコア + 可能性スコア) / 3**

脆弱性スコア

関数のスキャン(静的および動的)中に特定された脆弱性。

- 脆弱性の種類：
  - カスタムコードの脆弱性(静的および動的)
  - 依存関係(CVE)
  - 最小権限違反
- カスタムコードの脆弱性スコアの算出は 100 点から始まり、関数で検出された脆弱性の数に、深刻度に応じたペナルティの重み付けを掛けたペナルティ点数が差し引かれます。
  - 重大： 脆弱性の数 x 20
  - 高： 脆弱性の数 x 10
  - 中： 脆弱性の数 x 5
  - 低： 脆弱性の数 x 1
    - 例えば、関数の脆弱性が、「重大」0 件、「高」1 件、「中」0 件、「低」2 件だった場合、スコアは次のようになります：100 - (20 X 0) - (10 X 1) - (5 X 0) - (1 X 2) = 88

影響度(アクセスレベル)スコア

関数に与えられた権限(IAM ロール)。関数の権限が多いほど、リスクは高くなります。

影響度スコアを計算するために、各サービスに対して 5 つのアクセスレベル分類(List、Read、Write、Tagging、Permissions Management)を検査してスコアを付けます。そして、100 点から開始して、各サービスのアクセスレベルに応じてペナルティ点数を減算していきます。

例えば、以下のような IAM ポリシーがあるとします。

```
{
    "Effect": "Allow" ,
    "Action": [
        "s3:GetObject",
        "sqs:*"
    ],
    "Resource": "*"
}
```

各サービスのアクセスレベルのスコアが、以下のように算出されたとします。

```
{
  "s3": {
    "Read": 6,
    "Write": 3,
    "List": 3,
    "Tagging": 1,
    "Permissions management": 12
  },
  "sqs": {
    "Read": 3,
    "Write": 3,
    "List": 1,
    "Tagging": 1,
    "Permissions management": 6
  }
}
```

総合スコアは、次のように算出されることになります。

- s3: [6], sqs: [3,3,1,1,6] --> 100 - (6+3+3+1+1+6) = 80

可能性(到達可能性)スコア

攻撃者が関数に到達する可能性は、関数のトリガーの設定に基づきます。

各サービスには、攻撃者がその関数にアクセスできるかどうかだけでなく、トリガーの設定(例えば、認証あり/認証なし)に基づいた、異なるスコアがあります。

例：

例えば、関数に EventBus がトリガーとして設定されている場合、潜在的な攻撃者がこの Lambda 関数にアクセスできる可能性は、 API Gateway がトリガーとして設定されている Lambda にアクセスするよりも低くなります。さらに、API Gateway が認証なし(つまり Open)で設定されている場合には、その関数には誰でもどこからでもアクセスできることになります。

そのため、関数の可能性スコアは以下のようになります。

- EventBus をトリガーとして設定している場合：90
- API Gateway(認証あり)をトリガーとして設定している場合：75
- API Gateway(認証なし)をトリガーとして設定している場合：5 (最低スコア)
- トリガーがない場合：100 (最高スコア)

Contrast Serverless のアップグレード

Contrast Serverless は、ほとんどの場合、自動更新に設定されています。手動でアップグレードする必要がある場合は、このページに記載されている手順に従って下さい。

開始する前に

- 必要最低限のポリシーを持つロール/ユーザを作成すること。設定方法は、こちらの例 (642ページ)を参照して下さい。

手順

1. 前回のデプロイメントまでに Contrast Serverless のスタックに手動で変更を加えていない場合は、現在の Contrast Serverless スタックをアンインストール (666ページ)するだけでよいです。
   以前の Contrast Serverless スタックに手動で変更を加えた場合は、 スタックの変更セットを作成 (663ページ)してから、次の手順に進むことをお勧めします。
2. ツールバーで**新規登録**をクリックします。
3. **Download CFT** を選択して、Contrast から新しいテンプレートをダウンロードします。

4. JSON または YAML のいずれかを選択します。

5. スタックを更新します。

6. 表示された画面で、**送信**をクリックします。

スタックの変更セット

スタックの変更セットを作成するには、次の手順を実行して下さい。

開始する前に

- AWS CloudFormation コンソールで、Contrast Serverless のスタックを確認して下さい。

手順

1. Contrast Web インターフェイスにログインし、**Download CFT** を選択して、Contrast から新しいテンプレートをダウンロードします。

2. JSON または YAML のいずれかを選択します。
3. AWS アカウントにログインするか、AWS CLI/API を使用します。
4. **Contrast Serverless のスタック**を選択します。
5. **スタックアクション> 既存スタックの変更セットを作成**を選択します。

6. **既存テンプレートを置き換える** > **テンプレートファイルのアップロード**を選択して、手順 1 でダウンロードしたテンプレートファイルをアップロードします。

アカウントのブロック

アカウントのすべての Contrast Serverless アクティビティをブロックできます。

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. **Settings** タブを選択します。
3. ページの下部にスクロールし **Block Account** をクリックします。

これにより、自動的なスキャンやユーザが要求したスキャン、機能の更新など、すべてのアクティビティがブロックされます。

> **注記**
> アカウントのブロックを解除するには、Contrast サポートに連絡してください。

Contrast Serverless のオフボード

Azure から Contrast Serverless のアカウントをオフボーディングするには、スクリプトを使用してデプロイを削除する必要があります。

開始する前に

- 削除する対象を特定します

手順

1. Contrast Web インターフェイスのナビゲーションバーで**サーバレス**を選択します。
2. オフボードさせるアカウントを選択します。
3. **設定**タブを選択します。
4. **アカウントのオフボード**セクションまでスクロールして、スクリプトをコピーします。
5. 認証済みの `az` コマンドを使用して、シェルで実行します。 例えば、`Bash Azure CloudShell` などです。

Contrast Serverless のアンインストール

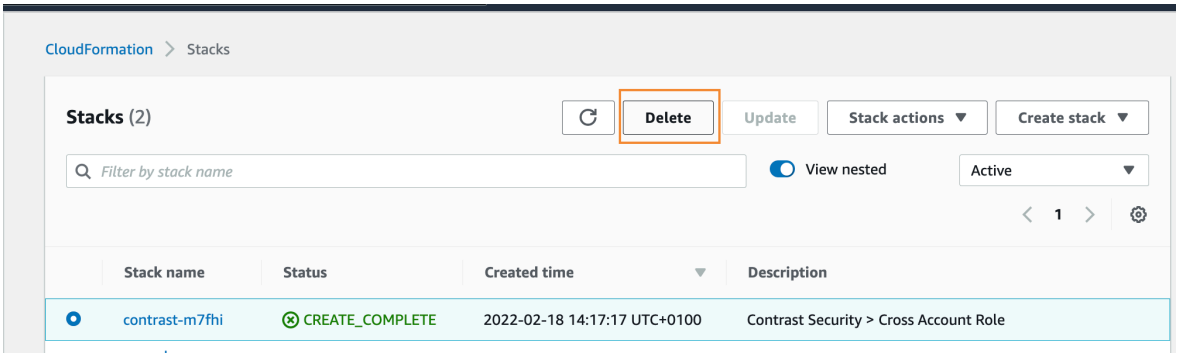
Contrast Serverless をアンインストールするには、AWS コンソールでスタックを削除する必要があります。

開始する前に

- 削除する対象を特定します

手順

1. AWS アカウントにログインするか、AWS CLI/API を使用します。
2. スタックを削除するための手順を続行します。AWS でのスタックの削除を参照してください。



Contrast CLI

Contrast CLI は、SCA、SAST、サーバレスの機能をコマンドラインで実現します。Contrast CLI はローカルで使用するか、CI/CD パイプラインの自動化で使用できます。

開始する前に

次の点に注意して下さい。

- CodeSec 利用者は、こちらから開始します。
- Contrast の企業ユースの場合は、Contrast CLI をインストール (667ページ)して使用を開始します。
- 従来の Contrast CLI は、2022 年 10 月をもって非推奨となります。

Contrast CLI について

Contrast Security の Contrast CLI により、開発者の端末で直接セキュリティテストを行うことが可能になります。迅速なスキャン時間、業界トップの検出精度、すぐに対策可能な検出結果、シームレスなインテグレーションなどにより、ソースコードやサーバレスのセキュリティがシンプルかつ効率的になります。

Contrast CLI は、以下を提供します。

- 最速で高精度の SAST スキャナ
- ソースコードやサーバレス環境のスキャン結果 - すぐに対策できる検出結果
- SCA 機能 - 脆弱性が導入された箇所も含めてオープンソースライブラリ間の依存関係を表示

簡単な手順でスキャンを開始できます。Contrast アカウントを既に持っている場合は、Contrast CLI をインストールする (667ページ)ことで開始できます。

Contrast CLI のサポート対象言語とパッケージマネージャ

Contrast CLI では、`audit` コマンドを使用する場合に次の言語をサポートします。

パッケージマネージャ	言語	必要なファイル	備考
RubyGems	**Ruby**	*gemfile* と *gemfile.lock*	
Composer	**PHP**	*composer.json* と *composer.lock*	
PyPI	**Python**	*pipfile* と *pipfile.lock*	
NuGet	**.NET Core** **.NET Framework**	MSBuild 15.0 以上と *packages.lock.json*	*packages.lock.json* ファイルがない場合、各*.csproj* ファイル内で `RestorePackagesWithLockFile` を「true」に設定して dotnet build を実行することで生成できます。 .NET では、`--legacy` オプションの使用のみがサポートされます。
Maven Central	**Java**	*pom.xml*	Maven ビルドプラットフォーム(依存関係プラグインを含む) が必要
		build.gradle	*./gradlew dependencies* (または、Windows の場合 gradlew dependencies) が必要
npm これには、クライアントサイドおよびサーバサイドの JavaScript パッケージが含まれます。	**JavaScript**	**JavaScript** ロックファイルは、バージョン 2 とバージョン 3 の両方がサポートされています。	
Gopm	**Go**	*go.mod*	

Contrast CLI のインストール

以下の手順で、Contrast CLI をインストールします。

Homebrew を使用する場合

以下のコマンドを実行し、Homebrew 経由で Contrast の tap からインストールします。

```
brew tap contrastsecurity/tap
brew install contrast
```

NPM/YARN を使用する場合

注記

現在は、Node.js 18 以降のみがサポートされます。

npm または yarn `@contrast/contrast` を使用してインストールします。

以下のコマンドを使用します。

```
npm install --location=global @contrast/contrast@2
```

バイナリを使用する場合

- アーティファクトにアクセスします。
- 最新のパッケージをダウンロードします。

- OS によっては、ファイルに*実行*権限を設定する必要があります。

インストールが完了したら、続けてアカウントの認証 (668ページ)を行います。

アカウント情報の認証

脆弱性をスキャンする前に認証を行い、認証情報を保存します。

認証情報をローカルに保存するには、以下の `auth` コマンドを実行します。

```
contrast auth
--api-key <your API key>
--authorization <your authorization header>
--host <your host domain>
--organization-id <your organization ID>
```

Contrast Web インターフェイスで、**ユーザメニュー** > **ユーザの設定** > **プロファイル**にアクセスし、以下の情報を取得してください。

- API キー
- 組織 ID
- 認証ヘッダー

`--host` の行には、Contrast の URL も必要です。

認証されたら、解析を行います (668ページ)。

セキュリティ解析

Contrast CLI を使用して、セキュリティ解析を行うことができます。

SAST スキャンの実行

1. ターミナルで、次のコマンドを入力します : `contrast scan -f <file name>`
2. 検出結果で、リンクをクリックしてスキャン結果 (569ページ)を確認します。

脆弱なライブラリの検査

1. ターミナルで、次のコマンドを入力します : `contrast audit`
2. 'audit'コマンドで`--track` フラグを使用した場合、検出結果でリンクをクリックしてライブラリを表示 (600ページ)します。

AWS Lambda 関数の脆弱性の検査

1. ターミナルで、次のコマンドを入力します : `contrast lambda--function-name []`
2. 検出結果で、推奨事項を確認し、提供された情報を基にポリシーを更新します。

Contrast Assess による脆弱性の検出

1. Contrast エージェントをインストールまたは更新します。
   - Java エージェントで Assess CLI を使用する (669ページ)
   - .NET エージェントで Assess CLI を使用する (670ページ)
   - Node.js エージェントで Assess CLI を使用する (671ページ)
   - Python エージェントで Assess CLI を使用する (671ページ)
   - Ruby エージェントで Assess CLI を使用する (672ページ)
   - Go エージェントで Assess CLI を使用する (673ページ)
2. ターミナルで、次のコマンドを入力します : `contrast assess`
   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。この設定ファイルのデフォルトの場所は次のとおりです。

- **MacOS および Linux：**
  `/etc/contrast/contrast_security.yaml`
- **Windows：**
  `%ProgramData%\Contrast\contrast_security.yaml`

`--config-path` オプションを使用して、別の場所を指定することもできます。

> **注記**
> 設定ファイルがあるディレクトリへの書き込み権限がない場合は、sudo または同様の方法を使用してフォルダを作成してください。例：
>
> ```
> sudo mkdir /etc/contrast
> ```
>
> そして、全てのユーザに読み取りおよび書き込み権限を付与します。例：
>
> ```
> sudo chmod 777 /etc/contrast
> ```

3. IDE または 2 つ目のターミナルウィンドウでアプリケーションを実行します。
4. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
5. Assess CLI コマンドを入力したターミナルウィンドウに検出結果が表示されます。

Java エージェントで Assess CLI を使用する

Contrast の Java エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

開始する前に

- Java エージェントのサポート対象テクノロジ (75ページ)を確認して、お使いのアプリケーションで Assess CLI を利用できるか確認してください。

手順

1. 最新の Java エージェントをインストールするために、エージェントを Maven Central からダウンロードします。

> **重要**
> エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。

2. ターミナルウィンドウを開き、Assess CLI コマンドを入力します。

```
contrast assess
```

このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
以下のような出力が表示されます。

```
Configuration file found at "user_path" ("user_path")
 Waiting for the session to be created. ()
```

3. IDE または 2 つ目のターミナルウィンドウで、次のコマンドでアプリケーションを実行します。

```
java -javaagent:<YourContrastJarPath> -jar <AppName>.jar
```

別の方法：

- **IntelliJ：** 実行構成を変更して、VM 引数として以下を含めます。

  ```
  -javaagent:<YourContrastJarPath>
  ```

  <YourContrastJarPath>を Java エージェントの contrast.jar ファイルへのパスに置き換えます。
  この実行構成によって、Java アプリケーションで自動的に Contrast エージェントを使用して、実行されます。
- **VS Code：**起動の構成(launch.json)の vmArgs 設定を変更して、VM 引数として以下を含めます。

  ```
  -javaagent:<YourContrastJarPath>
  ```

  <YourContrastJarPath>を Java エージェントの contrast.jar ファイルへのパスに置き換えます。
  上記の引数を vmArgs の項目に設定します。

4. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
5. Assess CLI コマンドを入力したターミナルウィンドウに検出結果が表示されます。

.NET エージェントで Assess CLI を使用する

Contrast の.NET エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

開始する前に

- .NET Core エージェントのサポート対象テクノロジ (225ページ)または.NET Framework エージェントのサポート対象テクノロジ (167ページ)を確認して、お使いのアプリケーションで Assess CLI が使用できることを確認してください。

手順

1. 使用する Contrast エージェントを手動でインストールまたは更新します。
   - .NET Core (227ページ)
   - .NET Framework (169ページ)

   重要
   エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。
2. ターミナルウィンドウを開き、Assess CLI コマンドを入力します。

   ```
   contrast assess
   ```

   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
   以下のような出力が表示されます。

   ```
   Configuration file found at "user_path" ("user_path")
    Waiting for the session to be created. ()
   ```
3. IDE または 2 つ目のターミナルウィンドウを使用して、アプリケーションを実行します。
4. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
5. Assess CLI コマンドを入力したターミナルウィンドウで、検出結果を確認します。

Node.js エージェントで Assess CLI を使用する

Contrast の Node.js エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

開始する前に

- Node.js エージェントのサポート対象テクノロジ (285ページ)を確認して、お使いのアプリケーションで Assess CLI を利用できるか確認してください。
- Contrast Assess は、サーバサイドのアプリケーションのみを対象としています。クライアントサイドのコードの脆弱性は検出されません。
- Node.js エージェントは、JavaScript アプリケーションのみに組み込み可能です。サーバサイドのコードに TypeScript を使用している場合は、JavaScript にトランスパイル (340ページ)してください。

手順

1. 以下のコマンドを使用して、アプリケーションのルートディレクトリから最新バージョンのエージェントをインストールします。

   ```
   npm install @contrast/agent
   ```

   yarn を使用する場合は、以下のコマンドを使用してください。

   ```
   yarn add @contrast/agent
   ```

   > **重要**
   > エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。

2. ターミナルウィンドウを開き、Assess CLI コマンドを入力します。

   ```
   contrast assess
   ```

   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
   以下のような出力が表示されます。

   ```
   Configuration file found at "user_path" ("user_path")
    Waiting for the session to be created. ()
   ```

3. IDE または 2 つ目のターミナルウィンドウで、以下のようなコマンドを使用してアプリケーションを実行します。

   ```
   node -r @contrast/agent <server.js>
   ```

   `<server.js>`を Node.js アプリケーションのサーバ起動コマンドに置き換えます。アプリケーションの仕様に合わせて、コマンドを調整してください。
   このコマンドで Node.js の Contrast エージェントが必須になり、Node.js エンジンによって読み取られるアプリケーションのソースコードに Contrast エージェントが組み込まれます。
4. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
5. Assess CLI コマンドを入力したターミナルウィンドウに検出結果が表示されます。

Python エージェントで Assess CLI を使用する

Contrast の Python エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

開始する前に

- Python エージェントのサポート対象テクノロジ (360ページ)を確認して、お使いのアプリケーションで Assess CLI を利用できるか確認してください。

手順

1. pip を使用してエージェントをインストールします。

```
pip install contrast-agent
```

ヒント
requirements.txt ファイルがある場合は、このファイルに contrast-agent を追加し、pip install -r requirements.txt でインストールできます。

重要
エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。

2. エージェントを実行するシステムに autoconf がインストールされていることを確認します。
3. ターミナルウィンドウを開き、Assess CLI コマンドを入力します。

```
contrast assess
```

このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
以下のような出力が表示されます。

```
Configuration file found at "user_path" ("user_path")
 Waiting for the session to be created. ()
```

4. IDE または 2 つ目のターミナルウィンドウを使用して、アプリケーションを実行します。
5. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
6. Assess CLI コマンドを入力したターミナルウィンドウで、検出結果を確認します。

Ruby エージェントで Assess CLI を使用する

Contrast の Ruby エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

開始する前に

- Ruby エージェントのサポート対象テクノロジ (416ページ)を確認して、お使いのアプリケーションで Assess CLI を利用できるか確認してください。

手順

1. 以下のエントリをアプリケーションの gemfile に追加します。

```
gem 'contrast-agent'
```

2. Contrast エージェントをインストールまたはアップデートします。
   - 以下のコマンドでエージェントをインストール：

```
bundle install
```

- 以下のコマンドでエージェントをアップデート：

```
bundle update contrast-agent
```

> **重要**
> エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。

3. ミドルウェア(Grape、Rails、Sinatra)を設定します。
   - **Grape**：`Grape::API` を拡張するアプリケーションクラスに直接ミドルウェアを追加するか、クラスが利用できない場合は `config.ru` ファイルにミドルウェアを追加します。

     ```
     require 'contrast-agent'
     use Contrast::Agent::Middleware, true
     ```
   - **Rails**： コードの変更は不要です。
   - **Sinatra**：`Sinatra::Base` を拡張するアプリケーションクラスに直接ミドルウェアを追加するか、クラスが利用できない場合は `config.ru` ファイルにミドルウェアを追加します。

     ```
     require 'contrast-agent'
     use Contrast::Agent::Middleware, true
     ```
4. エージェントを実行するシステムに `autoconf` がインストールされていることを確認します。
5. ターミナルウィンドウを開き、Assess CLI コマンドを入力します。

   ```
   contrast assess
   ```

   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
   以下のような出力が表示されます。

   ```
   Configuration file found at "user_path" ("user_path")
    Waiting for the session to be created. ()
   ```
6. IDE または 2 つ目のターミナルウィンドウを使用して、アプリケーションを実行します。
7. アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
8. Assess CLI コマンドを入力したターミナルウィンドウで、検出結果を確認します。

Go エージェントで Assess CLI を使用する

Contrast の Go エージェントを使用しており、API またはエンドツーエンドのテストを実行中に CLI を使用して脆弱性を検出したい場合は、この手順を使用してください。

Go エージェントを使用するアプリケーションの実行は、他の多くの Contrast エージェントとは異なります。Go エージェントは、コンパイル時にアプリケーションのソースコードに組み込まれます。

開始する前に

- Assess CLI をテストするために、Contrast Go Test Bench アプリケーションを使用できます。このサンプルアプリケーションの使用方法の詳細については、Contrast CodeSec の Web サイトを参照してください。
- Go エージェントのサポート対象テクノロジ (476ページ)を確認して、お使いのアプリケーションで Assess CLI を利用できるか確認してください。

手順

1. ターミナルウィンドウを開き、お使いの環境に Contrast Go エージェント (477ページ)(バージョン 1.19 以上)をインストールします。

   重要
   エージェントの設定ファイル(YAML)は作成しないでください。設定ファイルは、Assess CLI によって自動的に作成されます。

2. 以下のコマンドを使用して、コンパイラがインストールされていることを確認します。

   ```
   go version
   go version go1.19.1 darwin/arm6
   ```

3. アプリケーションをインストールし、コンパイルし、実行します。
   アプリケーションが Contrast を実装せずに実行されていることを確認するために、ブラウザを開いてアプリケーションにアクセスします。`CTRL-C` を入力して、アプリケーションを停止します。
   例えば、Contrast Go Test Bench アプリケーションを使用している場合は、`localhost:8080` にアクセスして確認します。
4. Assess CLI コマンドを入力します。

   ```
   contrast assess
   ```

   このコマンドによって、Assess CLI と Contrast エージェントの両方で共有するエージェント設定ファイルが作成されます。Contrast CLI コマンド (674ページ)にて、設定ファイルのパスを含め、このコマンドの各オプションについて説明しています。
   以下のような出力が表示されます。

   ```
   Configuration file found at "user_path" ("user_path")
    Waiting for the session to be created. ()
   ```

5. IDE または 2 つ目のターミナルウィンドウで、アプリケーションをコンパイルして実行すると、アプリケーションに Contrast Go エージェントが組み込まれます。
   例：Contrast Go Test Bench アプリケーションを使用している場合、コマンドは以下のようになります。

   ```
   go-test-bench on  main [!?] via  v1.19.1 took 1h52m1s
    contrast-go run ./cmd/gin/app.go
   ```

6. 3 つ目のターミナルウィンドを開き、アプリケーションを対話的に実行するか、API やエンドツーエンドのテストを使用して、アプリケーションを疎通します。
   例：Contrast Go Test Bench アプリケーションを使用している場合、コマンドは以下のようになります。

   ```
   go-test-bench on  main [!?] via  v1.19.1 took 4s
    go run ./cmd/exercise
   ```

7. 最初に開いたターミナルウィンドウで、結果を確認します。

Contrast CLI コマンド

以下は、CodeSec 利用者およびエンタープライズユーザ(企業ユース)が利用できる Contrast CLI コマンドの一覧です。

使用法：`contrast [] []`

認証/接続

auth

CodeSec 利用者で、Contrast アカウントをお持ちでない場合は、GitHub または Google アカウントを使用して認証します。ログイン用の新しいブラウザ画面が開きます。

- **使用法：**`contrast auth`

Contrast のアカウントを既にお持ちの場合は、以下の `auth` コマンドを実行すると、認証情報がローカルに保存されます。

- 使用法：

```
contrast auth
--api-key <your API key>
--authorization <your authorization header>
--host <your host domain>
--organization-id <your organization ID>
```

そして、コマンドで解析を実行 (668ページ)できます。

config
保存されている認証情報を表示します。

- **使用法：** `contrast config`
  例：

```
contrastuser@userc-C02GD0LUMD6TTY ~ % contrast config
{
version: '1.0.24',
host: 'https://ce.contrastsecurity.com',
apiKey: 'wwEHMnYEIAujE03fFGH',
organizationId: '0fde1b36-6986-4a14-b16d-6258aa913e5bceerfj',
authorization: 'Z2l1bGlhbmEubWFyaWFuaUBjb250cmFzdHNlY3VyaXR5LmNvbTpDUktMUTE3T1czMDU2NjlLOPDS',
numOfRuns: 0
}
```

- **オプション：**
  - `-c、--clear`
    保存されている認証情報を削除します。

version
Contrast CLI のバージョンを表示します。

- **使用法：**`contrast version`
  例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast version
1.0.24
```

メイン機能

audit
作業ディレクトリ内にある依存関係の設定ファイルを検索して依存関係のセキュリティ検査を実行し、結果を返します。

- **使用法：**`contrast audit []`
  例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast audit
Searching for package manager files from /Users/contrastuser/Documents/

Contrast SCA audit started...
```

```
Contrast audit complete

Found 4 vulnerable libraries with 4 CVEs

CONTRAST-001 - [CRITICAL]  minimist-1.2.5 introduces 1 vulnerability
       Issue:  1 Critical
               [C]CVE-2021-44906
       Advice: Update to version 1.2.6

CONTRAST-002 - [CRITICAL]  json-schema-0.2.3 introduces 1 vulnerability
       Issue:  1 Critical
               [C]CVE-2021-3918
       Advice: Update to version 0.4.0

CONTRAST-003 - [HIGH]  glob-parent-5.1.1 introduces 1 vulnerability
       Issue:  1 High
               [H]CVE-2020-28469
       Advice: Update to version 5.1.2

CONTRAST-003 - [HIGH]  ansi-regex-0.2.1 introduces 1 vulnerability
       Issue:  1 High
               [H]CVE-2021-3807
       Advice: Update to version 6.0.1
```

- **オプション：**
  - `--fail`
    検出された CVE の深刻度に基づいて、ビルドを失敗させます。**--severity** オプションと一緒に使用します。例えば、**contrast audit --fail --severity high** です。深刻度を指定しない場合、すべての深刻度で失敗が返ります。失敗が検出された場合、CLI コマンドはコード 2 で終了します。
  - `--file`
    依存関係が宣言されているディレクトリまたはファイルを指定します(デフォルトでは、Contrast CLI によって現在のディレクトリ内でプロジェクトファイルが検索されます)。ディレクトリ内で複数のプロジェクトファイルが見つかった場合、監査するファイルを確認するプロンプトが表示されます。
    エイリアス：`-f`
  - `--help`
    `audit` コマンドの全てのオプションの使用方法を表示します。
  - `--ignore-dev`
    検査結果から開発･試験用ライブラリの依存関係を除外します。デフォルトでは、全ての依存関係が結果に含まれます。
    エイリアス：`-i`
  - `--legacy`
    Contrast Web インターフェイスにアプリケーションを作成します(旧 CLI のワークフロー)。コードの依存関係ツリー (609ページ)を表示し、メタデータを利用します。これは、Contrast CLI V2.0 以降でのみ利用可能であることにご注意ください。
  - `--name`
    カスタムのプロジェクト名を設定します。名前が既に使用されている場合は、そのプロジェクトの検出結果を置き換えます。特殊文字は使用しないで下さい。
  - `--save`
    SBOM(ソフトウェア部品表)を SPDX や CycloneDX 形式で生成して保存します。有効なオプション：**--save cyclonedx**、**--save spdx** (CycloneDX 形式がデフォルト)
    エイリアス：`-s`
  - `--severity`

ビルドを失敗させる CVE の最低レベルの深刻度を指定します。**--fail** オプションと一緒に使用します。例えば、**contrast audit --fail --severity high** です。深刻度は、*critical*、*high*、*medium*、*low*、または *note* です。

- `--track`
  デフォルトでは、結果は保持· 保存されないため、コンソールでローカルチェックを行うことになります。`--track` フラグを追加すると、Contrast Web インターフェイスのライブラリ (600ページ)ページにある**静的**タブに、プロジェクトの SCA 結果を表示できます。これは、Contrast CLI V2.0 でのみ利用可能であることにご注意ください。

- **詳細オプション：**
  - `--api-key`
    エンタープライズユーザには必須です。Contrast で提供されるエージェントの API キーを指定します。キーの検索方法については、エージェントキー (61ページ)を参照してください。
  - `--application-id`
    Contrast に登録されているアプリケーションの ID を指定します。
  - `--application-name`
    Contrast に登録されているアプリケーションの名前です。
  - `--app-groups`
    アプリケーションのオンボーディング時に、1 つ以上の既存のグループにアプリケーションを割り当てます。グループリストは、カンマで区切る必要があります。
  - `--authorization`
    エンタープライズユーザには必須です。Contrast で提供される認証ヘッダーです。
  - `--code`
    Contrast でアプリケーションに使用するアプリケーションコードです。
  - `--host`
    エンタープライズユーザには必須です。ホスト名を指定します。(例) https://app.contrastsecurity.com/
  - `--maven-settings-path`
    maven の *settings.xml* ファイルのパスを表示します。
  - `--metadata`
    アプリケーションに関連付けるユーザ定義のメタデータを指定するための、キーと値のペアのセット(RFC 2253 に準拠)を定義します。
  - `--organization-id`
    エンタープライズユーザには必須です。Contrast での組織 ID を指定します。ID の検索方法については、エージェントキー (61ページ)を参照してください。
  - `--tags`
    アプリケーションにタグを適用します。タグは、カンマ区切りのリストとして書式設定する必要があります。(例) label1,label2,label3
- **プロキシの設定：**
  - `--cacert`
    CaCert(認証局(CA)による証明書)ファイルのパスを表示します。
  - `--cert`
    Cert(証明書)ファイルのパスを表示します。
  - `--cert-self-signed`
    ローカルインストールした Contrast オンプレミス(EOP)をご利用のお客様の場合、SSL 証明書をバイパスして自己署名証明書を認識します。
  - `--key`
    証明書の鍵のパスを表示します。
  - `--proxy`
    プロキシサーバ経由の接続を許可します。認証が必要な場合は、ユーザ名とパスワード、プロトコル、ホスト、ポートを例のように指定します。(例) "http://username:password@<host>:<port>"

パイプラインでビルドを失敗させるよう `audit` を使用するには、Contrast SCA Action を参照してください。

assess

Contrast エージェントを使用しているサーバで、実行時に検出された脆弱性を報告します。

- **使用法：** `contrast assess []`
  例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast assess
 Configuration file found at "user_path"
 Session created.
CONTRAST-001 - [HIGH] Path Traversal from "RawQuery" QueryString \
Parameter on
"/pathTraversal/os.Open/:source/:mode" pagePath Traversal \
from "RawQuery" QueryString Parameter on "/pathTraversal/
os.Open/:source/:mode" page
       App: CLIAssessApplication
    Source: GET
            /pathTraversal/os.Open/:source/:mode?
input=..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2Fetc%2Fpa
sswd
  Location: /opt/homebrew/Cellar/go/1.19.1/libexec/src/os/file.go, line \
316, in os.Open()
  Dataflow: "../../../../../../../../../../../../etc/passwd"
     Issue: Because there is untrusted data being used as part of the \
file path, it may be possible
            for an attacker to read sensitive data or write, update, or \
delete arbitrary files on the
            container's file system. The ability to write arbitrary \
files to the file system is also
            called Unrestricted or Arbitrary File Uploads.


CONTRAST-002 - [HIGH] Path Traversal from "RawQuery" QueryString \
Parameter on
"/pathTraversal/os.ReadFile/:source/:mode" pagePath Traversal \
from "RawQuery" QueryString Parameter on "/pathTraversal/
os.ReadFile/:source/:mode" page
       App: CLIAssessApplication
    Source: GET
            /pathTraversal/os.ReadFile/:source/:mode?
input=..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2Fetc%2Fpa
sswd
  Location: /opt/homebrew/Cellar/go/1.19.1/libexec/src/os/file.go, line \
672, in os.ReadFile()
  Dataflow: "../../../../../../../../../../../../etc/passwd"
     Issue: Because there is untrusted data being used as part of the \
file path, it may be possible
            for an attacker to read sensitive data or write, update, or \
delete arbitrary files on the
            container's file system. The ability to write arbitrary \
files to the file system is also
            called Unrestricted or Arbitrary File Uploads.


CONTRAST-003 - [HIGH] Path Traversal from "input[0]" Parameter on "/
pathTraversal/os.Open/:source/:mode"
pagePath Traversal from "input[0]" Parameter on "/pathTraversal/
```

```
os.Open/:source/:mode" page
       App: CLIAssessApplication
    Source: POST /pathTraversal/os.Open/:source/:mode
            \
input=..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2Fetc%2Fpa
sswd
  Location: /opt/homebrew/Cellar/go/1.19.1/libexec/src/os/file.go, line \
316, in os.Open()
  Dataflow: "../../../../../../../../../../../../etc/passwd"
     Issue: Because there is untrusted data being used as part of the \
file path, it may be possible
            for an attacker to read sensitive data or write, update, or \
delete arbitrary files on the
            container's file system. The ability to write arbitrary \
files to the file system is also
            called Unrestricted or Arbitrary File Uploads.
```

- **オプション：**
  - `--config-path <path>`
    Assess CLI と Contrast エージェントで共有する `contrast_security.yaml` ファイルのパスまたはディレクトリを指定します。
    指定しない場合、デフォルトのパスは以下の通りです。
    - **MacOS および Linux：** `/etc/contrast`
    - **Windows：** `%ProgramData%\Contrast\`

    エイリアス：`-c`
  - `--file <filename>`
    出力された脆弱性結果ファイルのパスまたはディレクトリを指定して、Contrast がそのファイルを読み取ってターミナルに結果を表示できるようにします。ファイル名は `contrast-assess-{Date}.jsonl` で、{Date}はエポックミリ秒で表されています。例：`contrast-assess-1691520302714.jsonl`
    エイリアス：`-f`
  - `--help`
    `assess` コマンドの全てのオプションの使用方法を表示します。
  - `--no-watch [true|false]`
    Contrast エージェントを使用して Assess を実行する際に `true` を設定すると、CLI はアクセス可能な脆弱性に対して Contrast バックエンドを監視(またはポーリング)しません。CLI は、特定の `buildNumber`(ビルド番号)に対して、一度だけ脆弱性を取得します。 デフォルトの設定は、`false` です。
    エイリアス：`-n`
  - `--output-path <path>`
    脆弱性結果ファイルを出力するパスまたはディレクトリを指定します。出力ファイルは、JSONL 形式です。ファイル名は `contrast-assess-{Date}.jsonl` となり、{Date}はエポックミリ秒で表されます。例：`contrast-assess-1691520302714.jsonl`
    エイリアス： `-o`
  - `--report-notes [true|false]`
    `true` に設定すると、access コマンドでは深刻度が「注意」の脆弱性を表示します。 デフォルトの設定は `false` で、深刻度の高い脆弱性が表示されます。
    エイリアス：`-r`

scan

SAST スキャンを実行します。

- **使用法：**`contrast scan []`
  例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast scan
Searching for files to scan from from /Users/contrast/Documents/
Searched 3 directory levels & found...
- spring-petclinic-1.5.1.jar
- webgoat-server-8.2.2.jar
- webgoat.jar

Java Scan requires a .war or.jar file. Javascript Scan requires a .js \
or .zip file.
To start a Scan enter "contrast scan -f <path-to-file>"
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast scan -f webgoat.jar
Found existing project...
Uploading...
Uploaded file successfully.
Contrast Scan started.

Here are your top priorities to fix

CRITICAL    sql-injection (2)
            1. org/owasp/webgoat/plugin/challenge6/Assigment6.java @43
            2. org/owasp/webgoat/plugin/challenge5/challenge6/
Assigment5.java @38
```

- **オプション：**
  - `--fail`
    検出された脆弱性の深刻度に基づいて、ビルドを失敗させます。**--severity** オプションと一緒に使用します。例えば、**contrast scan --fail --severity high** です。深刻度を指定しない場合、すべての深刻度で失敗が返ります。失敗が検出された場合、CLI コマンドはコード 2 で終了します。
  - `--file`
    スキャンするファイルのパスを指定します。ファイルが指定されていない場合は、Contrast はサポート対象ファイル(.jar、 .war、.js、.zip ファイル)を作業ディレクトリ内で検索します。
    エイリアス： `-f`
  - `--help`
    `scan` コマンドの全てのオプションの使用方法を表示します。
  - `--host`
    エンタープライズユーザには必須です。ホスト名を指定します。(例) https://app.contrastsecurity.com/
  - `--language`
    有効な値：JAVA、JAVASCRIPT、DOTNET
    エイリアス： `-l`
  - `--name`
    Contrast のプロジェクト名を指定します。指定されていない場合、Contrast は *contrast.settings* を使用してプロジェクトを識別するか、プロジェクトを作成します。
    エイリアス：`-n`
  - `-r`
    Contrast のリソースグループ名を指定します。このオプションは、SaaS 版をご利用のお客様で口ールベースのアクセス制御が有効になっている場合に必要です。
  - `--save`
    検出結果を SARIF(Static Analysis Results Interchange Format)ファイルでダウンロードします。ファイルは、デフォルトの名前の *results.sarif* で現在の作業ディレクトリにダウンロードされます。このファイルは、任意のテキストエディタで表示できます。
    エイリアス：`-s`
  - `--severity`

ビルドを失敗させる脆弱性の最低レベルの深刻度を指定します。**--fail** オプションと一緒に使用します。例えば、**contrast scan --fail --severity high** です。深刻度は、*critical*、*high*、*medium*、*low*、または *note* です。
  - `--timeout`
    スキャンが完了するまでの待機時間(wait)を秒数で指定します。デフォルトの値は、300 秒です。
    エイリアス：`-t`
- **詳細オプション：**
  - `--api-key`
    エンタープライズユーザには必須です。Contrast で提供されるエージェントの API キーを指定します。キーの検索方法については、エージェントキー (61ページ)を参照してください。
  - `--authorization`
    エンタープライズユーザには必須です。Contrast で提供される認証ヘッダーです。
  - `-ff`
    ファイア・アンド・フォーゲット(Fire-&forget)。実行させておくだけで、結果を監視・待つ必要がありません。
  - `--host`
    エンタープライズユーザには必須です。ホスト名を指定します。(例) https://app.contrastsecurity.com/
  - `--label`
    スキャンにラベルを付けます。デフォルトは、*CLI ツールで開始した日付[現在の日付]*です。
  - `--organization-id`
    エンタープライズユーザには必須です。Contrast での組織 ID を指定します。ID の検索方法については、エージェントキー (61ページ)を参照してください。
  - `--project-id`
    スキャンプロジェクトに関連付けられた ID です。ID は、Contrast Web インターフェイスでスキャンプロジェクトを選択した際の URL にある最後の文字列です。
- **プロキシの設定：**
  - `--cacert`
    CaCert(認証局(CA)による証明書)ファイルのパスを表示します。
  - `--cert`
    Cert(証明書)ファイルのパスを表示します。
  - `--cert-self-signed`
    ローカルインストールした Contrast オンプレミス(EOP)をご利用のお客様の場合、SSL 証明書をバイパスして自己署名証明書を認識します。
  - `--key`
    証明書の鍵のパスを表示します。
  - `--proxy`
    プロキシサーバ経由の接続を許可します。認証が必要な場合は、ユーザ名とパスワード、プロトコル、ホスト、ポートを例のように指定します。(例) "http://username:password@<host>:<port>"

lambda

AWS Lambda 関数をスキャンします。スキャンする AWS Lambda の関数名を指定してください。

- **使用法：**`contrast lambda --function-name <> []`
- **エイリアス：**`-f`
- **オプション：**
  - `--endpoint-url`
    AWS エンドポイントを上書きします。AWS CLI と同様です。
    エイリアス：`-e`
  - `--help`
    `lambda` コマンドの全てのオプションの使用方法を表示します。
  - `--region`

AWS リージョンを上書きします。デフォルトは、AWS_DEFAULT_REGION です。AWS CLI と同様です。
エイリアス：`-r`
- `--profile`
AWS の設定プロファイルを上書きします。AWS CLI と同様です。
エイリアス：`-p`
- `--json`
人が判読できるデフォルトのフォーマットではなく、レスポンスを JSON 形式で返します。
エイリアス：`-j`
- `--verbose`
ターミナルに詳細情報を返します。
エイリアス：`-v`
- `--list-functions`
スキャン可能な Lambda 関数を一覧表示します。
- `--help`
使用方法を表示します。
エイリアス：`-h`

- **プロキシの設定：**
  - `--cacert`
  CaCert(認証局(CA)による証明書)ファイルのパスを表示します。
  - `--cert`
  Cert(証明書)ファイルのパスを表示します。
  - `--cert-self-signed`
  ローカルインストールした Contrast オンプレミス(EOP)をご利用のお客様の場合、SSL 証明書をバイパスして自己署名証明書を認識します。
  - `--key`
  証明書の鍵のパスを表示します。
  - `--proxy`
  プロキシサーバ経由の接続を許可します。認証が必要な場合は、ユーザ名とパスワード、プロトコル、ホスト、ポートを例のように指定します。(例) "http://username:password@<host>:<port>"

ヘルプと学習コンテンツ

help

使用方法を表示します。CLI コマンドの詳細なヘルプを一覧表示 するには、**-h** か **--help** フラグをコマンドに追加します。

- **使用法：**`contrast help`
例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast help
Contrast CLI @ v1.0.24
Contrast Scan CLI
Pre-requisites  Java, Javascript and .NET supported                        \

To scan a Java project you will need a .jar or .war file for analysis      \

To scan a Javascript project you will need a single .js or a .zip of       \

multiple .js files                                                         \

To scan a .NET c# webforms project you will need a .exe or a .zip file \
for
analysis                                                                   \
```

```
                                                                          \
                                                                          \


The file argument is optional. If no file is given, Contrast will search \
for
a .jar, .war, .exe or .zip file in the working directory.                 \
                                                                          \


Submitted files are encrypted during upload and deleted in 24 hours.      \

Scan Options
-l, --language string   (optional): Valid values are JAVA, JAVASCRIPT \
and DOTNET
--label string          (optional): adds a label to the scan - defaults \
to 'Started by CLI tool at
current date'                                                             \

-n, --name string       (optional): Contrast project name. If not \
specified, Contrast uses
contrast.settings to identify the project or creates a project.           \

-f, --file string       (optional): Path of the file you want to scan. \
If no file is specified,
Contrast searches for a .jar, .war, .exe or .zip file in the working      \

directory.                                                                \

-t, --timeout number    (optional): Time in seconds to wait for scan to \
complete. Default value is
300 seconds.                                                              \

--fail                  (optional):  Use with contrast scan or contrast \
audit. Detects failures based
on the severity level specified with the --severity command. For \
example,
"contrast scan --fail --severity high". Returns all failures if no \
severity level is specified.                                              \

--severity type         (optional): Use with "contrast scan --fail --
severity high" or "contrast
audit --fail --severity high". Set the severity level to detect           \

vulnerabilities or dependencies. Severity levels are critical, high, \
medium,
low or note.                                                              \

-s, --save string       (optional): Saves the Scan Results SARIF to file.
Advanced
-o, --organization-id string   (required for Contrast Enterprise): The \
ID of your organization as provided
                                by Contrast UI
--api-key string                (required for Contrast Enterprise): An \
agent API key as provided by Contrast
```

```
UI                                                                          \

--authorization string         (required for Contrast Enterprise): An \
authorization header as provided by
Contrast UI                                                                 \

--host string                  (required for Contrast Enterprise): host \
name e.g.
https://app.contrastsecurity.com                                            \

--proxy string                 (optional): Allows for connection via a \
proxy server. If authentication is
required please provide the username and password with the protocol, \
host and
port. For instance: "https://username:password@<host>:<port>".              \

--key string                   (optional): Path to the Certificate Key      \

--cacert string                (optional): Path to the CaCert file          \

--cert string                  (optional): Path to the Cert file            \

--cert-self-signed             (optional):For EOP users with a local \
Teamserver install, this will bypass
the SSL certificate and recognise a self signed certificate.                \

-p, --project-id string        (optional): The ID associated with a scan \
project. Replace <ProjectID> with
the ID for the scan project. To find the ID, select a scan project in       \

Contrast and locate the last number in the URL.                             \

-l, --language string          (optional): Valid values are JAVA, \
JAVASCRIPT and DOTNET
--ff                           (optional): Fire and forget. Do not wait \
for the result of the scan.
--label string                 (optional):adds a label to the scan - \
defaults to 'Started by CLI tool at
current date'                                                               \


Need More Help? NEW users
Check out: https://support.contrastsecurity.com                             \

Learn more at: https://www.contrastsecurity.com/developer                   \

Join the discussion: https://www.contrastsecurity.com/developer/community

Existing Contrast Licensed user?
Read our docs: https://docs.contrastsecurity.com/en/run-contrast-
cli.html
Want to UP your game? type 'contrast learn'                                 \

 Advance your security knowledge and become an All-star coder   with        \
```

```
Contrast Secure Code Learning Hub.
```

- **エイリアス：**`-h`

learn

Contrast の Secure Code ラーニングハブを表示します。

- **使用法：**`contrast learn`
  例：

```
contrastuser@usercsa-C02GD0LUMD6TTY ~ % contrast learn
Opening Contrast’s Secure Code Learning Hub...
If the page does not open you can open it directly via https://
www.contrastsecurity.com/developer/learn
```

旧 Contrast CLI

> **重要**
> 従来の Contrast CLI は、2022 年 10 月をもって非推奨となります。新しい Contrast CLI (666ページ)をご利用頂きますようお願いします。

Contrast コマンドラインインターフェイス(Contrast CLI)を使用すれば、ソフトウェア開発ライフサイクル(SDLC)の初期段階でライブラリを解析できます。

Contrast CLI は Node.js で実行しますが、あらゆるアプリケーションで使用でき、コマンドラインでコンポジション解析機能が提供されます。サポート対象のプラットフォームや言語については、Contrast CLI のサポート対象言語 (685ページ)のページを参照してください。

このコンポジション解析によって、以下が可能です。

- 脆弱なライブラリを特定する
- CVE の深刻度に基づいてビルドを失敗させる
- 依存関係ツリー (609ページ)を表示し、ライブラリ間の依存関係や脆弱性がある箇所を把握する
- 依存関係かく乱の危険性がある Node.js ライブラリを特定する
- SBOM を作成する

Contrast CLI は、Contrast エージェントによる既存のランタイムの解析を補完し、コンパイル前の解析(通常、ランタイムには不可)を実現するものです。

Contrast CLI をインストール (686ページ)して、アプリケーションを登録 (687ページ)すれば、開発フェーズでコマンドラインオプション (689ページ)を使用してライブラリの解析を開始できます。

旧 Contrast CLI のサポート対象言語

Contrast CLI でサポートされる言語と要件は以下の通りです。

言語	要件	備考
Java	Maven	Maven プロジェクトは、`pom.xml` ファイルで定義され、Apache Maven Dependency プラグインが必要です。CLI がプロジェクトで動作するかをテストするには、`mvn dependency:tree` を実行して依存関係ツリーを表示します。
	Gradle (バージョン 4.8 以上)	*build.gradle* ファイルが必須となり、**gradle dependencies** または **./gradlew dependencies** にも対応している必要があります。

言語	要件	備考
.NET Framework、.NET Core	MSBuild 15.0 以上と *packages.lock.json* ファイル	*packages.lock.json* ファイルがない場合、各 *csproj* 内で `RestorePackagesWithLockFile` を **true** に設定して、.NET build を実行することで生成できます。
Node.js	*package-lock.json* または *yarn.lock* ファイルのいずれかが必要です。	脆弱性の報告は、React や Angular などのフロントエンドテクノロジに対してサポートされます。
PHP	*compose.lock* ファイルと *composer.json* ファイルが存在している必要があります。	
Python	*pipfile* および *pipfile.lock* ファイルが必要です。	
Ruby	*gemfile* および *gemfile.lock* ファイルが必要です。	
Go	*go.mod* ファイルが必要です。	

注記
現時点では、単一言語のアプリケーションのみがサポートされます。

旧 Contrast CLI のインストール

Contrast CLI (685ページ)をインストールするには：

1. Node.js をインストールします。Contrast CLI は Node.js パッケージとして実行されるため、これは必須です。現在、バージョン 10、12、14 に対応しています。
2. 検査対象のアプリケーションに Contrast エージェントを組み込んで起動します。これによって、Contrast CLI で参照するためのアプリケーション ID を取得できます。

   注記
   まだエージェントが組み込まれていないアプリケーションも、Contrast CLI で登録 (687ページ)することができます。ただし、アプリケーションのライブラリスコア (610ページ)を評価して、ライブラリ一覧を作成するには、エージェントを組み込んでアプリケーションを起動する必要があります。
3. プロキシ経由で Contrast と通信を行うには、エージェントの設定で `cli_proxy` プロパティを使用してください。
   認証が必要な場合は、ユーザ名とパスワード、プロトコル、ホスト、ポートを以下の例のように指定します。例：

   ```
   http://username:password@<host>:<port>
   ```
4. Contrast CLI で解析を行うには、対象のアプリケーションのソースコードが、ローカルで利用できる必要があります。アプリケーションの言語 (685ページ)の要件に従ってください。
5. 以下のコマンドを実行して、Contrast CLI をインストールします。

   ```
   npm install -g @contrast/contrast-cli
   ```

   または、以下のコマンドで yarn を使用して Contrast CLI をインストールすることもできます。

   ```
   yarn global add @contrast/contrast-cli
   ```

> **注記**
> Contrast CLI のインストールは、グローバルインストールで行う必要があります。

6. Contrast CLI のインストールが完了したら、アプリケーションを登録 (687ページ)して、コードの解析を始めることができます。

旧 Contrast CLI でアプリケーションを登録

Contrast CLI をインストール (686ページ)したら、結果を Contrast Web インターフェイスで確認するために、アプリケーションを最初に登録する必要があります。

> **ヒント**
> Contrast CLI を CI パイプラインの一部として呼び出し、ビルドプロセスの一部として自動化することもできます。

1. アプリケーション ID を確認します。アプリケーション ID は、ブラウザの Contrast URL の最後の URI セグメントです。

2. 認証に関するキー (522ページ)を確認します。以下が必要です。
   - API キー
   - 組織 ID
   - 認証ヘッダー
   - Contrast URL のサーバホスト名

   > **注記**
   > 入力する必要があるのは、サーバホスト名だけです。例えば、Contrast URL が *https://app.contrastsecurity.com/file/path/*の場合は、以下のように入力します。
   >
   > ```
   > --host app.contrastsecurity.com
   > ```

3. 以下のオプションのいずれかを使用して CLI を実行し、解析を開始します。
   - <APIKey>、<AuthorizationKey>、<OrganizationID>、<Host>および<ApplicationID>を、自分の API キー、認証ヘッダ、組織 ID、ホスト名およびアプリケーション ID に置き換えて、以下のコマンドを実行します。

     ```
     contrast-cli \
     --api_key <APIKey> \
     --authorization <AuthorizationKey> \
     --organization_id <OrganizationId> \
     --host <Host> \
     --application_id <ApplicationId>
     ```

   - 上記と同様に<>の箇所を置き換えて、認証情報を YAML ファイル内に指定します。

     ```
     cli:
         api_key: <APIKey>
         authorization: <AuthorizationKey>
     ```

```
    organization_id: <OrganizationId>
    host: <Host>
    application_id: <ApplicationId>
```

<path/to/yaml>を自分の YAML パスに置き換えて、以下のコマンドを実行して開始します。

```
contrast-cli --yaml_path <path/to/yaml>
```

注記

TLS(Transport Layer Security)などの通信プロトコルを経由する必要がある場合は、YAML ファイルに以下のパラメータを追加します。

```
key: pathToKey
cert: pathToCert
cacert: pathToCaCert
```

4. 「SUCCESS」のメッセージが表示されたら、依存関係ツリーを表示 (609ページ)できます。

ヒント

まだ Contrast で検査を行なっていないアプリケーションでも、`--catalogue_application` と`--application_name` オプションを使用して、新規にアプリケーションを Contrast に登録することができます。ただし、Contrast Web インターフェイスでライブラリのスコア (610ページ)とライブラリー覧が表示されるようにアプリケーションにエージェントを組み込んで (46ページ)起動することお勧めします。

例：

```
contrast-cli \
--catalogue_application \
--api_key <YourApiKey> \
--authorization <YourAuthorizationKey> \
--organization_id <YourOrganizationID> \
--host <YourHost> \
--application_name <YourApplicationName> \
--language <YourApplicationLanguage>
```

<APIKey>、<AuthorizationKey>、<OrganizationID>、<Host>、<ApplicationName>を、自分の API キー、認証ヘッダ、組織 ID、ホスト名、およびアプリケーション名に置き換えます。また<ApplicationLanguage>は、解析するアプリケーションの言語に置き換えます。指定できる言語の値は、JAVA、DOTNET、NODE、PHP、PYTHON、RUBY、GO です。

この登録(catalogue)操作が成功すると、コンソールにアプリケーション ID が出力されます。

注記

CLI コマンドの一連のオプション (689ページ)を指定することで、アプリケーションの登録と SBOM レポートの作成を同時に行うこともできます。

旧 Contrast CLI のコマンド

CLI で`-h`または`--help`オプションを指定すると、コマンドラインのヘルプを参照できます。ヘルプガイドには、Contrast の設定、アプリケーション、脆弱性などに関する情報を理解するために使用できる以下のコマンドオプションがあります。

下記の例では、`<string>`や`<level>`をご利用の環境に合わせた値に置き換えてください。

一般的なコマンド

接続や設定に関するコマンドオプションです。

オプション	説明
`--api_key <string>`	Contrast で提供されるエージェントの API キー (61ページ)(必須)
`--application_id <string>`	Contrast に登録されているアプリケーションの ID(必須)
`--application_name <string>`	Contrast に登録されているアプリケーションの名前(任意)
`--authorization <string>`	Contrast で提供されるユーザの認証ヘッダ(必須)
`-h, --help`	ヘルプを表示します。
`--host <string>`	ホスト名。オプションでポートを`<host>:<port>`と指定することもできます。 URL のプロトコル要素 (*https://*)は含めません。デフォルトは、*app.contrastsecurity.com* です。(任意)
`--language <string>`	アプリケーションの言語。有効な値は、JAVA、DOTNET、NODE、PHP、PYTHON、RUBY、GO。`project_path` に複数のプロジェクトの設定ファイルがある場合、言語の指定が必須。(catalogue 操作には必須)
`--organization_id <string>`	Contrast での組織 ID (61ページ)(必須)
`--project_path <string>`	検査したいプロジェクトやアプリケーションのディレクトリのルート。デフォルトは現在のディレクトリです。(任意、但し Windows で実行する場合は必須)
`--proxy <string>`	Proxy サーバ経由の接続を許可します。認証が必要な場合は、ユーザ名とパスワード、プロトコル、ホスト、ポートを例のように指定します(任意)。例、*http://username:password@<host>:<port>*
`--silent`	JSON 出力をサイレントにします。(任意)
`--sub_project <string>`	Gradle アプリケーション内のサブプロジェクトを指定します。(任意)
`-v, --version`	現在使用している CLI のバージョンを表示します。
`--yaml_path <string>`	YAML ファイルからパラメータを読み込む場合に指定する YAML ファイルへのパス(任意) `yaml_path` を使用すると、ターミナルからの以下の接続パラメータは無視されます。 • `yamlOnly:` • `key:pathToKey` • `cert:pathToCert` • `cacert:pathToCaCert`

注記

特殊文字の問題を回避するために、これらのコマンドのパラメータを引用符で囲む必要がある場合があります。例：

```
--application_name = "My_app_name_$+=(/\"
```

SCA

Contrast SCA の検査に関するコマンドオプションです。

オプション	説明
アプリケーションのカタログ作成	
`--app_groups <string>`	`catalogue` コマンドを使用時に、アプリケーションを 1 つ以上の既存のグループに割り当てます。グループリストは、カンマで区切る必要があります。(任意)
`--catalogue_application`	アプリケーションを登録(catalogue)します(必須)。アプリケーション名が存在しない場合は、アプリケーションを作成して依存関係ツリーが送信され、存在する場合は既存のアプリケーションに依存関係ツリーが追加されます。
`--code <string>`	Contrast でこのアプリケーションに使用するアプリケーションコード(任意)
`--metadata <string>`	アプリケーションに関連付けるユーザ定義のメタデータを指定するための、キーと値のペアのセット(RFC 2253 に準拠)を定義します。(任意)
`--tags <string>`	アプリケーションにタグを適用します。タグは、カンマ区切りのリストとして書式設定する必要があります(任意)。例：label1,label2,label3
スナップショット - Java のみ	
`--maven_settings_path <PathToFile>`	Maven の `settings.xml` ファイルの別の場所を指定できます。`<PathToFile>`の箇所をファイルのフルパスに置き換えます。 Contrast CLI でアプリケーションを登録 (687ページ)するときに、一連のキーにこのパスを追加してください。(任意)
アプリケーションの登録	

`--cli_api_key <string>`
`--cli_authorization <string>`
`--cli_organization_id <string>`
`--cli_host <string>`
`--language <string>`
`--application_name <string>`
`--sbom`
	アプリケーションの登録と SBOM レポートの取得を同時に行うには、この一連のコマンドオプション(値は前述と後述の表に記載)を使用します。
	注：パラメータの"cli_"という接頭辞は、将来のリリースで廃止される予定です。

レポート	
`--cve_severity <level>`	`--report` と組み合わせると、選択した深刻度 (707ページ)以上の脆弱性があるライブラリが報告されます。例えば、`cve_severity medium` を指定すると、**中**(Medium)と中より高い深刻度の脆弱性のみが報告されます。
`--cve_threshold <number>`	ビルドが失敗するまでに許容する CVE の数をしきい値として設定します。しきい値を超える CVE があると、ビルドは失敗になります。
`--fail`	脆弱性が検出された場合、ビルドを失敗にします。`cve_severity` と組み合わせて使用すると、定義した深刻度の脆弱性検出時にビルドを失敗にすることができます。
`--report`	コンパイル時のアプリケーションの脆弱性情報を表示します。
`--ignore_dev`	`--report` コマンドと組み合わせると、開発・試験用ライブラリの依存関係を脆弱性レポートから除外します。デフォルトでは、全ての依存関係がレポートに含まれます。
SBOM	
`--sbom`	SBOM(ソフトウェア部品表) (721ページ)を CycloneDX JSON フォーマットで生成してダウンロードします。

ヒント

`--report` コマンドを使用すると、全ての脆弱なライブラリの情報がターミナル に返されます。検出された全ての CVE は、以下のように出力されます。

```
org.webjars/jquery-ui/1.11.4 is vulnerable

CVE-2016-7103 MEDIUM Cross-site scripting (XSS) vulnerability \
in jQuery UI before 1.12.0 might allow remote attackers to \
inject arbitrary web script or HTML via the closeText \
parameter of the dialog function.
```

`--cve_severity` パラメータを使用して、レポートする CVE の最小しきい値を設定することで、出力される脆弱性情報を制限できます。

`--fail` パラメータを、自動化された CI/CD パイプラインの一部に使用することで、深刻度のしきい値を超えたライブラリでアプリケーションがデプロイされないようにできます。例えば、次のように YAML ファイルを使用して CLI を実行できます。

```
contrast-cli --yaml_path path/to/yaml --report --cve_severity \
high --fail
```

スキャン

Contrast Scan に関連するコマンドオプションです。ビルドとスキャンのインテグレーション (583ページ)もご覧ください。

Contrast Scan は、.NET プロジェクトの EXE および ZIP ファイルをサポートします。アプリケーションの言語は DOTNET に設定し、ZIP は DLL を含む./bin フォルダの ZIP である必要があります。

オプション	説明
`--project_id <ProjectID>`	スキャンプロジェクトに関連付けられた ID。<ProjectID>をスキャンプロジェクトの ID に置き換えます。ID は、Contrast Web インターフェイスでスキャンプロジェクトを選択した際の URL にある最後の文字列です。 **推奨：**最初のスキャンでは、このオプションの代わりに**--project_name** オプションを使用してください。スキャンによってプロジェクトが作成されます。
`--project_name`	スキャンプロジェクトの名前を指定します。プロジェクト名にスペースが含まれる場合は、二重引用符(")で囲みます。 新しい名前を指定すると、プロジェクトが作成されます。既存の名前を指定すると、アップロードされたファイルはそのプロジェクトに追加されます。
`--save_scan_results`	このオプションが指定された場合は、SARIF ファイルが *results.json* として現在のディレクトリに保存されます。(任意)
`--scan<FileToBeScanned>`	指定された WAR または JAR ファイルの静的スキャンを開始します。`<FileToBeScanned>`を、スキャン用にアップロードする WAR または JAR ファイルのパスに置き換えます。
`--scan_results_file_name`	JSON ファイル形式である必要があります。このオプションが指定された場合は、SARIF ファイルを保存するためのデフォルトファイル名が上書きされます。(任意)
`--scan_timeout`	スキャンがタイムアウトするまでの時間(秒単位で)を指定します。`scan_timeout` が設定されていない場合、デフォルトのタイムアウトは 20 秒です。

オプション	説明
`--wait_for_scan`	スキャンの結果を待機(wait)します。

脆弱性

アプリケーションにエージェントを組み込む (46ページ)と、アプリケーションで検出された全ての脆弱性が Contrast に表示され、最も一般的で重大なリスクやその他の多くの脆弱性に対処することができます。

Contrast エージェントにより、アプリケーションにあるコードの欠陥が検出されて報告されます。これらの脆弱性は深刻度を付けて表示・分類されるので、必要に応じて、対処する脆弱性の優先順位を決め、脆弱性のステータスを更新することができます。

- 脆弱性の表示 (692ページ)
- アプリケーションの脆弱性を表示 (693ページ)
- 脆弱性の解析 (700ページ)
- 脆弱性の追跡 (699ページ)
- 脆弱性の修正 (701ページ)

組織レベルで脆弱性を表示

開始する前に

- Contrast で脆弱な箇所を検出して結果を表示できるよう、アプリケーションを疎通(閲覧や操作)します。
- より詳細な脆弱性情報を参照したい場合は、セッションメタデータ (535ページ)を報告するよう Contrast エージェントを設定します。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択します。
2. 脆弱性一覧の一番上で、**ライセンスありのみを表示**を選択すると、ライセンスのあるアプリケーションのみが表示されます。
3. 列でフィルタをかけるには、列のヘッダの横にある**フィルタ**アイコン( )を選択します。フィルタには以下のオプションがありますが、選択したアプリケーションに適用可能な場合に利用できます。
   - **深刻度**：利用可能なフィルタは、重大、高、中、低、注意です。
   - **脆弱性** ：利用可能なフィルタは次の通りです。
     - **脆弱性のタグ：** 作成したカスタムタグに関連付けられた脆弱性。
     - **種類**：脆弱性の種類。
     - **サーバ**：選択したサーバに関連するアプリケーションの脆弱性。
     - **環境：** 選択した環境(開発、QA、本番)にあるアプリケーションの脆弱性。
     - **シンク**：共通のシンクに起因する脆弱性。
       シンクは、複数のデータフローの脆弱性間で共有されているカスタムコードです。
       シンクでフィルタをかけることで、複数の脆弱性の原因となっているコード行を特定することができます。
     - **URL**：特定の URL に関連する脆弱性。
     - **コンプライアンスポリシー**：コンプライアンスポリシーに関連する脆弱性。
   - **アプリケーション**：利用可能なフィルタは次の通りです。
     - **アプリケーション名：** アプリケーションに関連付けられた名前。
     - **カスタムタグ：** アプリケーションに割り当てたタグ。
     - **言語：** アプリケーションで使用されている言語。
     - **テクノロジ：** アプリケーションで使用されているテクノロジ。例えば、JSON や jQuery など。
     - **アプリケーションの重要度：** アプリケーションの設定で指定したアプリケーションの重要性。

- **アプリケーションメタデータ**：アプリケーションに関連付けられたアプリケーションメタデータ。
- **最後の検出**：利用可能なフィルタは、最初の検出、最後の検出、時間範囲です。 特定の日付と時刻を指定する場合は、**カスタム**を選択します。
- **ステータス**： 利用可能なフィルタは、ステータスおよび脆弱性を追跡中かどうかです。

フィルタを解除するには、列のヘッダの横にある**クリア**を選択します。

4. 脆弱性の詳細を表示するには、脆弱性の名前を選択します。以下のカテゴリの情報が表示されます。
   - HTTP 情報
   - 脆弱性を修正する方法
   - ビルド番号、脆弱性を報告しているサーバ、脆弱性のカテゴリ、セキュリティ基準など、脆弱性の識別やタイミング、場所に関する詳細な情報。

関連項目

アプリケーションの脆弱性を表示 (693ページ)

アプリケーションの脆弱性の表示

アプリケーションページのアプリケーション一覧から、特定のアプリケーションの脆弱性を表示できます。

開始する前に

- Contrast で脆弱な箇所を検出して結果を表示できるよう、アプリケーションを疎通(閲覧や操作)します。
- より詳細な脆弱性情報を参照したい場合は、セッションメタデータ (535ページ)を報告するように Contrast エージェントを設定します。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   アプリケーションの一覧には、各アプリケーションでオープン中の脆弱性の数が表示されます。特定の種類(重大、高など)の脆弱性に関する情報を表示するには、「オープン中の脆弱性」の列にある棒グラフで該当する箇所を選択します。

   オープン中の脆弱性とは、ステータスが報告済、疑わしい、確認済のものです。
2. あるいは、アプリケーションの一覧でアプリケーション名を選択して、**脆弱性**タブを選択します。
   選択したアプリケーションの脆弱性の一覧が表示されます。
3. 脆弱性タブで脆弱性にフィルタをかけるには、一覧の最上部にある小さい三角形を選択します。

   利用できるフィルタ：
   - オープン中
   - 信頼性の高い問題
   - ポリシー違反
   - レビュー待ち

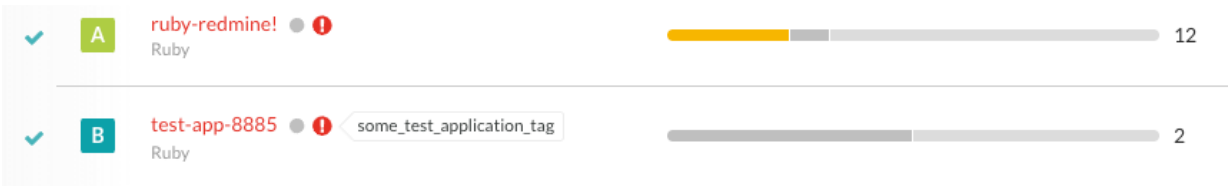
4. 特定の脆弱性を検索するには、虫眼鏡アイコン( )を選択します。
5. 一覧の上部にある**傾向線**のマーク( )を選択すると、脆弱性のタイムラインが表示されます。
   グラフの上のボタンを使用すると、**深刻度**(Severity)または**検出**(Discovery)別にデータの表示が切り替わります。グラフの傾向線にカーソルを合わせると、その時点でのデータの内訳(脆弱性の数、タイムスタンプまたはステータス)が表示されます。
   脆弱性一覧のフィルタを適用すると、グラフのデータも更新されます。**最後の検出**の列のフィルタを使用すると、タイムラインに表示される期間が更新されます。
6. 列でフィルタをかけるには、列のヘッダの横にあるフィルタアイコン▼を選択します。フィルタには以下がありますが、選択したアプリケーションに適用可能な場合に利用できます。
   - **深刻度**：利用可能なフィルタは、重大、高、中、低、注意です。
   - **脆弱性**：選択したアプリケーションで適用可能な場合、利用可能なフィルタは以下の通りです。
     - **脆弱性のタグ**：脆弱性に割り当てたカスタムタグ。
     - **バグ管理システム**：バグ管理システムとの連携を利用して、脆弱性を追跡しているかどうか。
     - **種類**：脆弱性の種類。
     - **モジュール**:脆弱性に関連するアプリケーションモジュール(マージされたアプリケーションのモジュールも含む)。
     - **サーバ**：アプリケーションをホストしているサーバ。
     - **環境**：開発環境、QA 環境、本番環境。
     - **シンク**：共通のシンクに起因する脆弱性。
       シンクは、複数のデータフローの脆弱性間で共有されているカスタムコードです。
       シンクでフィルタをかけることで、複数の脆弱性の原因となっているコード行を特定することができます。
     - **URL**：特定の URL に関連する脆弱性。
     - **コンプライアンスポリシー**：選択したコンプライアンスポリシーに関連する脆弱性。
     - **ルート**：選択したルートに関連する脆弱性。
   - **アプリケーション**：アプリケーションに含まれるモジュール。
     マージされていないアプリケーションを参照している場合は、選択したアプリケーションの脆弱性がこのフィルタによって表示されます。
     マージされたアプリケーションの脆弱性を参照している場合は、マージされたアプリケーションのモジュールの脆弱性がこのフィルタによって表示されます。
   - **最後の検出**：利用可能なフィルタは、最初の検出、最後の検出、時間範囲です。 特定の日付と時刻を指定する場合は、**カスタム**を選択します。
   - **ステータス**：利用可能なフィルタは、ステータスおよび脆弱性を追跡中であるかどうかです。
   - **セッション**： この列は、エージェントの設定ファイルにセッションメタデータが設定されているが、セッションメタデータフィルタを選択していない場合に表示されます。「セッション」列フィルタを使用して、結果を絞り込むことができます。
     一覧の上にある**〜で表示**(〜はプロパティ名)メニューを使用して、エージェントの設定ファイルで指定したセッションメタデータ値で、データを絞り込みます。このフィルタは、「セッション」列に表示される値を更新します。
     「〜で表示」メニューは、エージェントの設定ファイルにセッションメタデータが設定されているが、セッションメタデータフィルタを選択していない場合に表示されます。

マージしたアプリケーションでオープン中の脆弱性

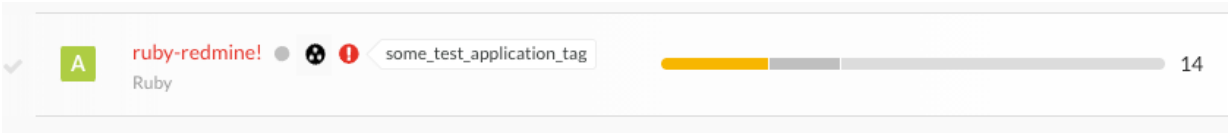
マージしたアプリケーションの場合、アプリケーション一覧の「オープン中の脆弱性」列には、メインアプリケーションにある全てのアプリケーションモジュールの脆弱性の数が表示されます。アプリケーション一覧には、メインアプリケーションは表示されますが、メインアプリケーションに含まれるモジュールは表示されません。

例：

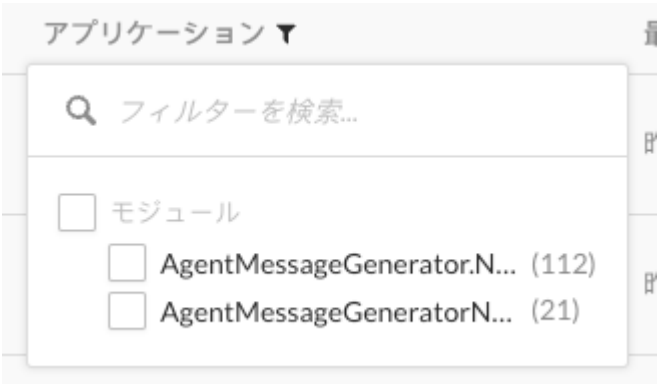
アプリケーションをマージする前の「オープン中の脆弱性」列が、以下のようであるとします。



アプリケーションをマージしたら、「オープン中の脆弱性」列にある棒グラフには、メインアプリケーションとマージされた全てのモジュールの脆弱性が表示されます。



マージされたアプリケーションの「脆弱性」タブを表示して、「アプリケーション」列のフィルタを使用すると、Contrast で脆弱性が検出されたモジュール名が表示されます。



関連項目

組織レベルで脆弱性を表示 (692ページ)

脆弱性の発生率の表示

オープン中およびクローズされた脆弱性の比率を表すグラフは、組織内のすべてのアプリケーションの脆弱性の傾向を示します。

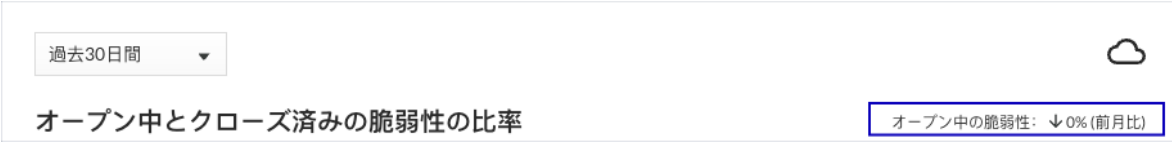
手順

1. **新しいダッシュボードを表示**を選択して、新しいダッシュボードを表示します。

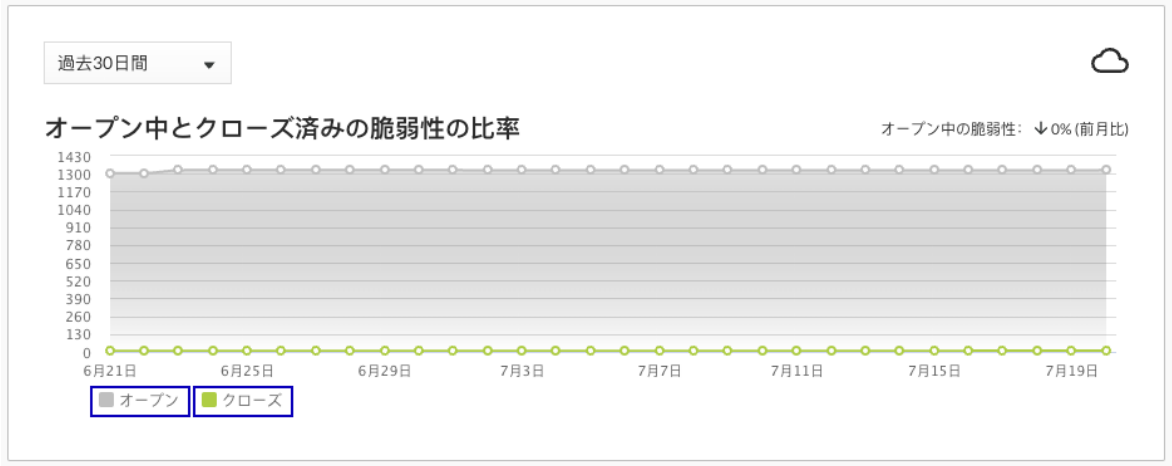
2. 新しいダッシュボードにある、**オープン中とクローズ済みの脆弱性の比率**のグラフを使用します。
3. グラフの期間を選択：
   - **過去 7 日間**： 現在の日付から 7 日前までの脆弱性の発生率が表示されます。
     グラフには、各日のデータポイントが表示されます。
   - **過去 30 日間**：現在の日付から 30 日前までの脆弱性の発生率が表示されます。
     グラフには、各日のデータポイントが表示されます。
   - **過去 12 か月間**：現在の日付から 12 か月前までの脆弱性の発生率が表示されます。
     グラフには、各月のデータポイントが表示されます。
   - **カスタマイズ**：選択した期間の脆弱性の発生率が表示されます。
     グラフには、選択した期間に応じて、各日、各週、または各月のデータポイントが表示されます。

選択した期間の一部でデータが存在しない場合、グラフにはその期間のデータは表示されません。例えば、過去 12 か月の期間を選択しても、Contrast の使用開始が 9 か月前の場合、グラフにはその期間の最初の 3 か月のデータは表示されません。

4. 選択した期間内での変化率を参照するには、グラフの上部にある値を確認してください。



5. オープン中の脆弱性のみ、またはクローズされた脆弱性のみを表示するようグラフにフィルタをかけるには、グラフの下部にあるキーを選択します。
   キーを再度選択すると、選択は解除されます。
   - オープン中の脆弱性のみを表示するには、**クローズ**キーを選択します。
   - クローズされた脆弱性のみを表示するには、**オープン**キーを選択します。



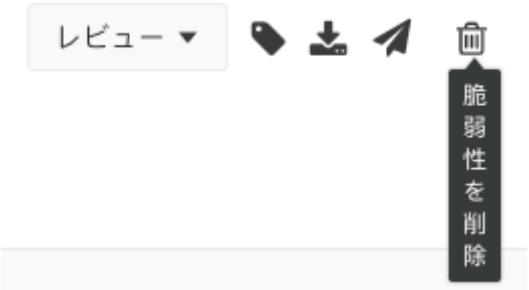
脆弱性の削除

アプリケーションにエージェントを組み込むと、脆弱性が自動的に検出され、脆弱性を Contrast で確認 (692ページ)できるようになります。これらの脆弱性は、セキュリティの問題に関する具体的な対応方針や問題の切り分け方に応じて、リスクを評価して、誤検知を排除し、修正対象の優先順位を付けることができます。

参照する必要がなくなった脆弱性は、削除することができます。

脆弱性を削除するには：

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択します。
2. 削除する脆弱性の行にカーソルを合わせて、行の右端にある**削除**アイコンを選択します。このアイコンは、脆弱性の詳細ページの右上にもあります。
   複数の脆弱性を一括で削除するには、左側の列にあるチェックマークを使用し、削除する脆弱性を選択してから、ページの下部に表示される一括アクションバーから**削除**アイコンを選択します。



3. 表示される画面で、**削除**を選択して処理を確定します。確定すると、脆弱性は削除され、Contrast で再検出されない限り一覧に表示されなくなります。

シンクで脆弱性をグループ化

脆弱性をグループ化する場合に、同じシンクを共有する脆弱性をまとめることができます。グループ内にある脆弱性は複数のアプリケーションに影響を与えます。

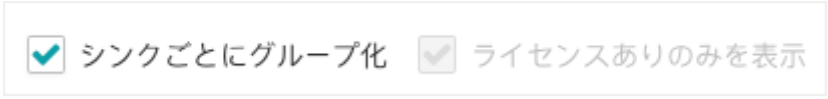
脆弱性をグループ化することにより、脆弱性一覧に表示される件数が少なくなります。グループ内の個々の脆弱性のデータは、引き続き参照できます。

開始する前に

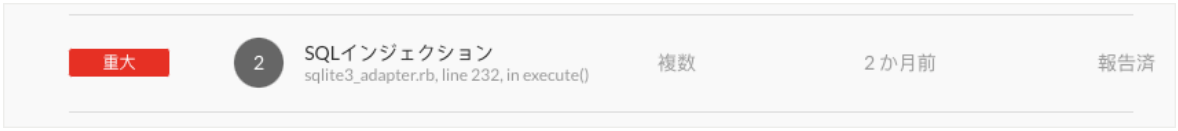
シンクごとのグループ化は、Contrast Assess ライセンスがあるアプリケーションの脆弱性にのみ適用できます。「シンクごとにグループ化」を選択すると、**ライセンスありのみを表示**が自動的に選択されます。

手順

1. Contrast Web インタフェースのナビゲーションバーで**脆弱性**を選択します。
2. 一覧の上の**シンクごとにグループ化**を選択します。



一覧に使用するフィルタによっては、グループを見つけるために下にスクロールする必要がある場合があります 。
グループは、次の例のような表示になります：



数字は、グループ内の脆弱性の数を示します。
グループ内の脆弱性の深刻度が異なる、対象のアプリケーションが複数ある、またはステータスが異なるものがある場合は、「深刻度」、「アプリケーション」、「ステータス」列の値が**複数**に変わります。

3. 一覧をさらに絞り込むには、**脆弱性**フィルタを 1 つ以上選択します。
4. グループ内の個々の脆弱性を参照するには、そのグループをクリックします。
   そのグループの脆弱性のみのが一覧に表示されます。
5. グループ化を解除するには、**シンクごとにグループ化**のチェックを外します。

脆弱性のマージ

同じアプリケーションから同じ種類の脆弱性が検出された場合に、これらをマージして検出結果をまとめることができます。これを行うには：

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択します。
2. 左側の列にあるチェックマークを使用して、マージする 2 つ以上の脆弱性を選択します。
3. ページの下部に表示される一括アクションバーで、**マージ**アイコンを選択します。

4. 表示される画面で、このマージのまとめ先となる脆弱性を選択します。

脆弱性へのタグの追加

脆弱性にタグを付けることで、脆弱性を整理することができ、検索機能が向上します。タグを付けるには、以下の手順を実行します。

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択して脆弱性を一覧表示し、タグを付ける脆弱性の行にカーソルを合わせます。
2. 行の右端にある**タグ**アイコンを選択します。このアイコンは、脆弱性の詳細ページの右上からもアクセスできます。

複数の脆弱性にタグを付けるには、脆弱性一覧の左側の列にあるチェックマークを使用して、タグを付ける脆弱性を選択します。ページの下部に表示される一括アクションメニューで、**タグ**アイコンを選択します。

3. 表示される画面で入力を始めると、既に作成済みのタグの一覧が表示されます。既存のタグを使用する場合は、ドロップダウンから 1 つ以上のタグを選択します。もしくは、新規にタグを作成する場合は、フィールドに新しいタグを入力します。タグを外すには、タグの横にある **X** をクリックします。変更を保存するには、**保存**を選択します。
4. タグで脆弱性を絞り込むするには、一覧の**脆弱性**列の横にあるフィルターを選択して、絞り込むタグのチェックボックスを選択します。

5. タグは、脆弱性の詳細ページでも脆弱性名の横に表示されます。タグの横にある **X** を選択すると、タグが外れます。

脆弱性の追跡

バグ管理システムを Contrast とインテグレーションしている場合、複数の方法で脆弱性を追跡することができます。

- 脆弱性の情報を組織の他のメンバーに送信する。
- 攻撃を防ぐためにタイムリーなパッチ適用を計画し、管理する。
- 脆弱性の情報をバグ管理システムに直接送信することで、ワークフローを効率化する。
- 深刻度が高または重大の脆弱性が新たにアプリケーションで検出された場合に、通知を受信する。

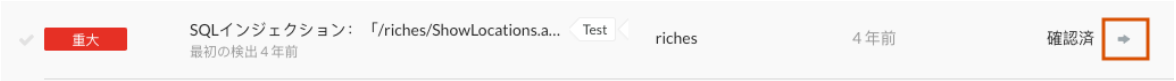
開始する前に

少なくとも 1 つのバグ管理システム (726ページ)と Contrast を連携してください。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択します。
2. 1 つの脆弱性を追跡するには、追跡したい脆弱性の行の最後にカーソルを合わせます。
3. 行の右端にある**送信**アイコンを選択します
   この送信アイコンは、脆弱性の詳細ページの右上からもアクセスできます。

4. 複数の脆弱性を追跡する場合：
   a. 追跡したい各脆弱性の横にあるチェックマークを選択します。
   b. ページの下部に表示される一括アクションメニューで、 **送信**アイコン( )を選択します。
   c. **バグ管理システムへ送信**を選択します。
      脆弱性の追跡情報をメールで送信するよう選択することもできます。
5. 「脆弱性を送信」の画面で、ドロップダウンから使用するバグ管理システムを選択し(複数のシステムと連携している場合)、関連する情報を追加して、**送信**を選択します。
   脆弱性のステータスが自動的に**報告済**に更新され、脆弱性の行のステータスの横に矢印アイコンが表示されます。矢印アイコンの上にカーソルを合わせると、バグ管理システム名や対応するチケット番号などの詳細情報が表示されます。



6. バグ管理システムで追跡中の脆弱性を確認するには、**脆弱性**列のフィルタを使用して、**追跡中**のチェックボックスを選択します。

ヒント
カスタムレポートなどのカスタマイズ処理用に、脆弱性データを CSV や XML ファイルにエクスポート (701ページ)できます。また、API を使用して Contrast Web インターフェイスの外部でデータを収集することもできます。

脆弱性イベントの解析

Contrast では、脆弱性イベントというものを使用して、アプリケーションの操作中に観察された内容を情報として提供します。これらのイベントには、脆弱性がコード内のどこで検出されたかや、コードがどのように使用されたかの情報があります。脆弱性イベントには、いくつかの種類があります。

- **ソースイベント**：ソースイベントは、脆弱性の開始点で発生します。ソースイベントのファイルと行番号を使用して、呼出しが行われた箇所を確認できます。そして、ソースのスタックトレースを使用して、注目すべきメソッドがプログラムでどのように呼び出されているかを理解できます。また、以下のようなメソッドに関する全てのデータも参照できます。
  - **オブジェクト**： このコールを呼び出しているオブジェクトのインスタンス(静的な呼出しでない場合)。
  - **リターン**： このコールから返される値(void の場合は null)。
  - **パラメータ**： このコールに渡されている値。
- **伝播イベント**：各脆弱性には、1 つまたは複数の伝播イベントが含まれる場合があります。伝播イベントには、ソースイベントと同じ情報が含まれていますが、データが伝播された方法を示すタイプも含まれます。例えば、P2R の伝播イベントでは、1 つ以上のパラメータからデータを受け取り("P2R"の"P")、そのデータをメソッドの戻り値に送ります("P2R"の"R")。
- **タグイベント**： **validated** や **html-encoded** などのタグを脆弱性に追加するイベントです。これらのタグは、誤検知を排除し、安全で信頼できる結果を得るのに役立ちます。また、他の種類のイベント

と同じコンテキスト情報もあります。タグイベントは脆弱性内で発生する場合がありますが、検出された脆弱性とは無関係です。
- **トリガーイベント**：トリガーは、脆弱性の最後のイベントです。トリガーとなった呼び出しによって、Contrast の JVM プラグインのルールエンジンの解析が実行されて脆弱性が認識され、トレースが生成されます。

重要

Contrast では、アプリケーションの実際の動作のみを検出します。脆弱性の問題が正当ではないと思われる場合、管理者は適切なポリシーを設定 (783ページ)して、この問題が再び発生しないようにする必要があります。最もよく報告される誤検知は、アプリケーションに Contrast で認識されていないカスタム制御がある場合です。

オンプレミス版(EOP)をご利用のお客様は、Contrast のポリシーで、適切なタグリストにカスタムメソッドの呼出しを登録できます。例えば、HTML エンコードを行うカスタムメソッドで、文字列を受け取り HTML エンコードされた文字列を返す場合、データに **html-encoded** のタグを追加します。

誤検知に対処するために、セキュリティ制御 (783ページ)やアプリケーションの例外 (804ページ)を使用することができます。

脆弱性の修正

脆弱性が検出されたら、自社のセキュリティ要件に応じてリスクを評価する必要があります。検出された脆弱性のリスクを評価して、その脆弱性を修正すると判断した場合は：

1. 脆弱性の名前を選択して詳細ページを開き、脆弱性の詳細情報を確認します。**修正方法**タブを選択して、この問題を解決するための推奨策を確認します。
2. 適切と思われる方法で、脆弱性を修正します。
3. 修正した脆弱性を確認します。確認する方法は、3 つあります。
   - **リクエストを再生**： 問題を修正して適切なステータスに変更したら、HTTP リクエストを再生します。**HTTP 情報**タブから HTTP リクエストを再生して、問題が修正されているか確認します。修正されていない場合、その問題は報告済のステータスで再度表示されます。
   - **ビルド番号を確認**： アプリケーションごとに、ビルドのバージョン番号を割り当てることができます。セッションメタデータ (535ページ)を使用して、脆弱性のビルド番号を識別できます。`-javaagent` コマンドに以下のプロパティを追加します：

     ```
     -Dcontrast.override.appversion
     ```

     アプリケーションの起動時にビルド番号を指定すれば、ビルド番号をフィルターとして使用できます。備考タブを参照するか、ドロップダウンメニューでビルド番号を選択することで、そのビルドバージョンにまだ問題があるかを確認できます。
   - **単体テストの時間で確認**: 単体テストを実行した時間でフィルターをかけることもできます。脆弱性の一覧で日付範囲を指定して表示を絞りこみます。

脆弱性の検出結果のエクスポート

脆弱性の情報をエクスポートするには：

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択し、脆弱性の一覧の左側の列にあるチェックマークを使用して、エクスポートしたい脆弱性を選択します(複数選択可)。
2. ページの下部に表示される一括アクションメニューで**エクスポート**アイコンを選択して、エクスポートする形式(CSV または XML)を選択します。

Contrast で、データのエクスポートが開始します。

3. エクスポートが完了すると、通知が表示されます。通知パネル(3)をチェックして、エクスポート完了のメッセージを確認してください。 通知には、エクスポートされたデータをダウンロードするためのリンクがあります。
脆弱性ごとに、以下の情報がエクスポートされます。
   - Vulnerability Name (脆弱性名)
   - Vulnerability ID (脆弱性 ID)
   - Category (カテゴリー)
   - Rule Name (ルール名)
   - Severity (深刻度)
   - Status (ステータス)
   - Number of Events (イベント数)
   - First Seen (最初の検出)
   - Last Seen (最後の検出)
   - Application Name (アプリケーション名)
   - Application ID (アプリケーション ID)
   - Application Code (アプリケーションコード)
   - CWE ID (CWE 識別子)
   - Request Method (リクエストメソッド)
   - Request Port (リクエストポート)
   - Request Protocol (リクエストプロトコル)
   - Request Version (リクエストバージョン)
   - Request URI (リクエスト URI)
   - Request Qs (リクエストクエリ)
   - Request Body (リクエストボディ)
   - Instance ID(インスタンス ID)

ヒント

アプリケーションに関して、より詳細なカスタムのソフトウェアコンポジション解析(SCA)レポートを作成する場合、アプリケーション API を使用して Contrast の脆弱性データにアクセスできます。

また、手動で脆弱性の詳細情報を調べることもできます。

例えば、次の cURL リクエストは、脆弱性の一覧を取得し、各脆弱性が検出されたアプリケーションの一覧も表示します。カスタムレポートで使用するために、jq ツールでデータを CSV 形式にしています。

```
curl \
    -H "Authorization: $(echo -n $username:$servicekey | \
base64)" \
    -H "API-Key: $apikey" \
    https://app.contrastsecurity.com/Contrast/api/ng/$orgid/
orgtraces/filter?expand=request | \
    jq -r '.traces[] | {uuid: .uuid, \
protocol: .request.protocol} | [.uuid, .protocol] | @csv'
```

CVE に関連付けられている CWE の検索

CVE(共通脆弱性識別子)に関連付けられている CWE(共通脆弱性タイプ一覧)を見つけたい場合は、NIST(米国国立標準技術研究所)が管理している NVD(脆弱性情報データベース)を利用できます。

多くの CVE が CWE に関連付けられていますが、特定の CWE に分類されないものや、複数の CWE と関連しているものもあります。

手順

1. NVD の Web サイトに移動します。
2. CVE を検索します。
   a. NVD のメニューから **Search**(検索)を選択します。
   b. **Vulnerabilities - CVE**(脆弱性 - CVE)を選択します。
   c. **Keyword Search**(キーワード検索)で、CVE 識別子(例えば、CVE-2020-12345)を入力します。
   d. **Search**(検索)を選択します。
3. 検索結果が表示されたら、CVE のリンクを選択します。
4. CWE の情報を表示するには、「Weakness Enumeration(脆弱性タイプ一覧)」の下に表示されている一覧からいずれかの CWE のリンクを選択します。
   CWE のリンクを選択すると、 CWE の Web サイトで CWE の詳細が表示されます。

脆弱性のステータス

脆弱性のステータスは脆弱性一覧に表示され、以下の表に示すいずれかのステータスになります。脆弱性のステータスは、更新できます。

ステータス	このステータスをいつ設定するか
報告済	Contrast で脆弱性が検出された時のデフォルトのステータスです。アプリケーションでこの脆弱性が悪用される可能性あります。
確認済	ソースコードをレビューする、脆弱性を悪用してみることなどで、この脆弱性が真の判定であると確認した場合のステータスです。
疑わしい	脆弱性は、Contrast から提供された情報に基づくと真の判定のように見えますが、その有効性を判断するにはさらに調査が必要な場合のステータスです。
問題無し	脆弱性は、ソースコードを変更することなく対処した場合のステータスです。このステータスを設定するには、以下のいずれの理由を選択する必要があります。このステータスを設定した脆弱性は、再検出されても**報告済**のステータスに戻ることはありません。 • **外部制御により防御された攻撃：** WAF などの別のコンポーネントが環境にあり、この脆弱性の悪用は防御されます。 • **誤検知：**この脆弱性は誤って報告されたものです。Contrast でこれが脆弱性として判定された理由を確認するには、サポートにお問い合わせください。 • **内部のセキュリティ制御を通過：** アプリケーション内にカスタムの修正コードがあり、この脆弱性の悪用は防御されます。 • **信頼できるパワーユーザのみがアクセスできる URL：** この脆弱性は、テストなどの特定の環境にのみ存在し、本番環境には存在しない可能性があります。 • **上記以外：** この脆弱性の対応に関してソースコードの変更が必要ではない理由が他にある場合、このオプションを選択します。**上記以外**というラベルをカスタム定義に置き換え (705ページ)て、脆弱性が**問題無し**である理由として指定できます。
修復済	脆弱性は、アプリケーション内のソースコードや設定ファイルなどを変更することで修正された場合のステータスです。
修正完了	この脆弱性は、ソースコードの変更もしくは**問題無し**のステータスで指定された理由によって、修正された場合のステータスです。このステータスを設定した脆弱性は、再検出されても**報告済**のステータスに戻ることはありません(このオプションは、管理者のみ使用可能です)。
修復済 - 自動検証	このステータスは、自動でのみ設定されます(ユーザが手動で設定することはできません)。脆弱性のポリシー (786ページ)で設定した期間内に脆弱性が報告されなければ、自動的に**修復済 - 自動検証**のステータスに変わります。

報告済、**確認済**、**疑わしい**のステータスが設定されている脆弱性は、オープン中の脆弱性となります。**問題無し**、**修復済**、**修正完了**、または**修復済 - 自動検証**のステータスの脆弱性は、クローズされたものとなります。脆弱性の一覧で**オープン中**のフィルターを選択すれば、オープン中のステータスの脆弱性

のみが表示されます。**全て**を選択すると、オープン中とクローズされた両方のステータスの脆弱性が表示されます。

エージェントから報告された脆弱性が、それまでに Contrast で検出されたことがない場合、その脆弱性のエントリが Contrast で新規に作成されます。この脆弱性が既に存在する場合、Contrast は、既存のエントリ、問題数、および最後に検出されてからの日数を更新します。再検出時に全ての脆弱性は、**修復済**または**修復済 - 自動検証**.に設定されていたものを除いて、以前と同じステータスで再オープンされます。修復済と修復済 - 自動検証の場合は、**報告済**として再オープンされます。

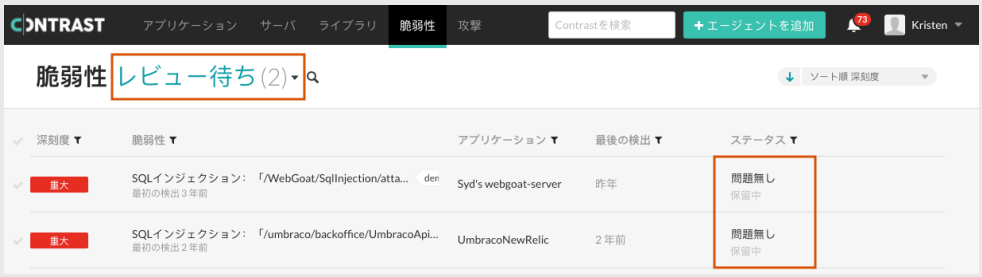
脆弱性のステータスの変更

1 つまたは複数の脆弱性のステータスを変更するには：

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択します。
2. 1 つの脆弱性ステータスを変更するには、ステータスを変更する脆弱性の行で**ステータス**列をクリックし、ステータスを選択します。脆弱性の概要ページの右上にあるステータスからも変更できます。

複数の脆弱性のステータスを一括で更新するには、左側の列にあるチェックマークを使用して、更新する脆弱性を選択します。ページの下部に表示される一括アクションメニューに、現在のステータスが表示されます。クリックすると、ステータスの一覧が表示されます。

3. 新しいステータスを選択します。ステータスの定義については、ステータスの一覧 (703ページ)を参照ください。

> **注記**
> 脆弱性には、クローズする前に組織の管理者の承認を必要 (868ページ)とするよう設定することもできます。
>
> 管理者の承認が必要な脆弱性をクローズする場合は、ステータス変更の理由を入力する必要があり、組織の管理者か RulesAdmin 権限のあるユーザにレビューされるまで**保留中**のステータスになります。**ステータス**列の**保留中**にカーソルを合わせると、保留とされた日付を確認できます。
>
> 
>
>
> 保留中の脆弱性のステータスは変更できる場合があります。新しいステータスへの変更に承認の必要がない場合は、脆弱性のステータスには**保留中**が表示されなくなります。
>
> 管理者がステータスの変更を承認または拒否すると、通知が送信されます。拒否の場合、脆弱性のステータスは以前の状態に戻りますが、管理者はその判定の理由を入力する必要があります。入力された理由は、脆弱性の**アクティビティ**タブに表示されます。

4. (**問題無し**の場合)表示される画面で、ステータスを変更する理由を選択し、説明を入力します。**問題無し**の理由には、カスタムラベルを定義 (705ページ)することができます。

問題無しの脆弱性の理由をカスタム定義

セキュリティ担当が、特定の脆弱性についてコードの変更による修正は不要と判断し、脆弱性のステータスを**問題無し**に更新する場合があります。これにより、チームは脆弱性の修正に集中することができ、Contrast でこれらの脆弱性が再検出されるのを防ぎます。

脆弱性のステータスに**問題無し**を指定する場合、理由を選択する必要があります。Contrast では、標準の理由 (703ページ)に加えて**上記以外**(標準の理由以外)というオプションがあります。

上記以外というラベルを組織にとってわかりやすい独自の値に変更できます。変更をするには、以下の手順を実行します。

1. 組織の**ポリシーの管理**の画面を開きます。
2. **脆弱性の管理**を選択します。
3. **上記以外の理由のカスタムラベルを設定**を選択します。
4. 独自に設定したい理由を入力します。25 文字以内で入力してください。
5. **保存**を選択します。

これにより、脆弱性のステータスを**問題無し**にすると、理由の一覧には**上記以外**ではなくカスタム定義の理由が表示されるようになります。

> **注記**
> **上記以外**をカスタムラベルに変更する、またはカスタムラベルを**上記以外**に戻すと、そのラベルが付いている全ての脆弱性に更新が適用されます。

保留中の脆弱性ステータス変更のレビュー

組織の管理者が、特定の脆弱性について承認を必要とする (868ページ)ように設定している場合、そのステータス (703ページ)は承認されるまで変更されません。 これは、手動による脆弱性ステータスの変更、双方向のバグ管理システムとの連携、および自動修復ポリシーに適用されます。

脆弱性のクローズを承認または拒否するには、組織ロールに RulesAdmin 権限があり、対象アプリケーションに対して RulesAdmin がある必要があります。

これを行うには：

1. Contrast Web インターフェイスの通知内のリンクを選択するか、ナビゲーションバーの**脆弱性**を選択してから、一覧の上部にあるフィルターを選択して全ての保留中のレビューを表示します。

2. 左側の列のチェックマークを使用して、1 つ以上の脆弱性を選択します。ページの下部に表示される一括アクションメニューで、**レビュー**を選択します。次に**承認**または**拒否**を選択します。脆弱性の概要ページの右上にある**レビュー**を選択することもできます。
3. ステータスの変更を拒否する場合は、理由を入力する必要があります。拒否された脆弱性は、以前のステータスに戻ります。承認された脆弱性は新しいステータスに変わり、**保留中**の表示がなくなります。どちらの場合も、レビューの結果は脆弱性の**アクティビティ**タブに表示されます。

脆弱性の深刻度の変更

Contrast では、アプリケーションの脆弱性は 5 つの深刻度レベルに分類されます。この分類は、アプリケーションの脆弱性の可能性と影響度を基準にしており、最も深刻度の高いものから低いものまでがあります。

- 重大
- 高
- 中
- 低
- 注意

脆弱性の深刻度は、以下の手順で変更できます。

1. Contrast Web インターフェイスのナビゲーションバーで**脆弱性**を選択し、脆弱性の一覧を表示します。
2. **深刻度**の列で色付きのバッジをクリックし、リストから新たに設定する深刻度を選択します(複数の脆弱性の深刻度を一括で変更することはできません)。

攻撃

攻撃とは、アプリケーションやサーバを標的とした攻撃イベントのグループです。Contrast で攻撃として含める攻撃イベントには次のように複数ありますが、これらに限定されるものではありません。

- SQL インジェクション
- 信頼できないデータのデシリアライゼーション
- コマンドインジェクション
- その他、多数の一般的に広く知られている脆弱性

同一の IP アドレスから 30 分以内に複数の攻撃イベントを検出した場合、Contrast ではこれらのイベントを攻撃としてグループ化します。コードを修正した後に、同じ IP アドレスから新たなイベントが検出されると、新たな攻撃として表示されます。

ダッシュボードに表示される攻撃の日付は、ローカルのタイムゾーンではなく、Contrast タスクが実行された Contrast サーバ上の時間に基づきます。

イベントデータの保持

Contrast では、攻撃イベントのデータは 30 日間保持されてから、削除されます。攻撃データをより長期間保持するには、以下を行います。

- Syslog に出力 (596ページ)
- Generic Webhook (750ページ)を設定
  Webhook は、指定されたイベントが発生した場合にのみ、POST リクエストでデータを受信します。Webhook はイベントを確認すると、データを収集して指定された URL に送信します。
- 攻撃の行の最後にある矢印を選択し、ドロップダウンメニューから**攻撃をエクスポート(CSV か XML 形式)**を選択 します。

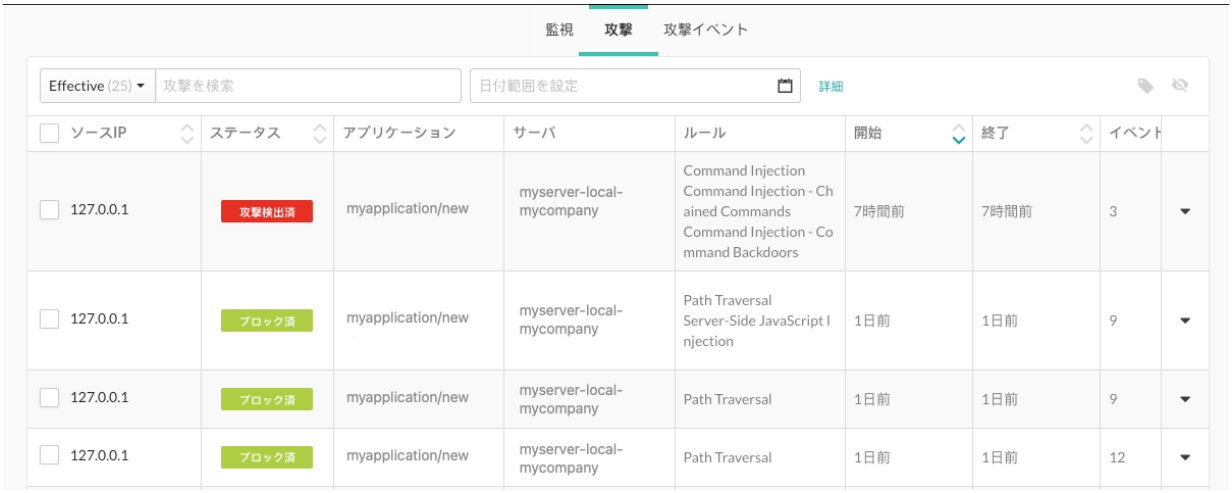
操作

Contrast では、次のような操作ができます。

- 攻撃情報の表示 (708ページ) : 攻撃されたアプリケーションやサーバ、攻撃が発生したコード箇所など、攻撃の情報を確認します。
- 攻撃の管理 (711ページ) : 攻撃や攻撃イベントに対してアクションを実行します。例えば、特定の攻撃イベントに対して Protect ルールを設定することができます。
- 攻撃の監視 (710ページ) : 現在および過去の攻撃について概要画面でモニターします。

攻撃の表示

攻撃の一覧には、組織内で発生した全ての攻撃が表示されます。

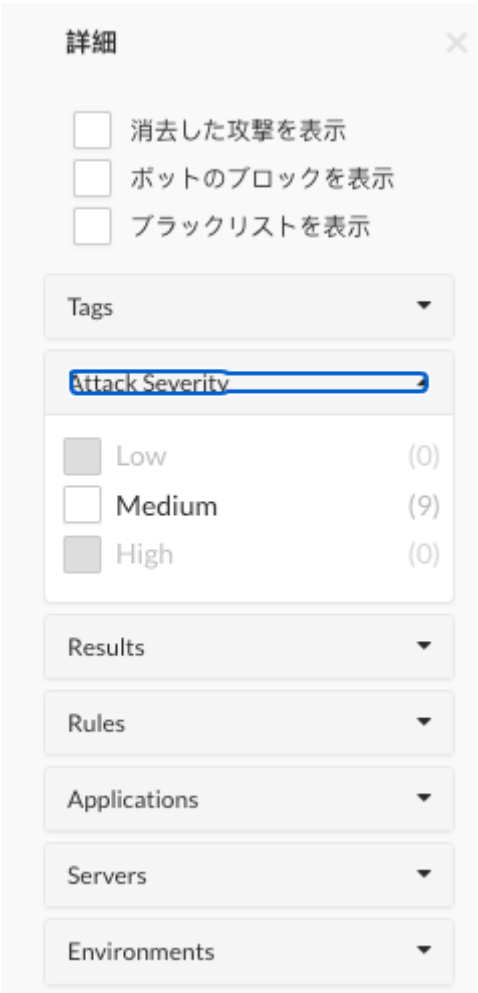


手順

1. Contrast Web インターフェイスのナビゲーションバーで**攻撃**を選択します。
2. 組織内で発生した全ての攻撃を表示するには、**攻撃**タブを選択します。
3. 表示を絞り込むには、次のいずれかのフィルタを選択します。
   - **全て**： 全ての攻撃を表示
   - **効果的**： ステータスが**ブロック済**、**疑わしい**、**攻撃検出済**の攻撃を表示

- **本番：** 本番サーバで発生した攻撃を表示
- **積極的な攻撃：** 現在進行中の攻撃を表示
- **マニュアル：** 1 秒あたりのリクエストが 20 未満の攻撃を表示。人間のマニュアル操作によって攻撃が発生している可能性あり。
- **自動化：** 1 秒あたりのリクエストが 20 を超える攻撃を表示。悪質なボットによって攻撃が発生している可能性あり。

4. 攻撃の詳細を表示するには、ソース IP 列で**ソース名**か **IP アドレス**を選択します。
5. 攻撃内の攻撃イベントを確認するには、**概要**タブを選択します。
6. さらに表示を絞り込むには、日付範囲の横にある**詳細**をクリックするとフィルタを選択できます。



7. 攻撃イベントの詳細を表示するには、**ソース IP** を選択します。
8. 各イベントの時刻を表示するには、**攻撃期間**の下にある**タイムラインを表示**を選択します。

9. **備考**タブを選択すると、イベントの発生率、深刻度、攻撃者などの情報が表示されます。
10. **履歴**タブを選択すると、チーム内でコメントを共有したり参照できます。
    既存のコメントを参照したり、**コメントを追加**をクリックして新規のコメントを入力できます。

攻撃の詳細

Contrast で参照できる攻撃データには、以下の項目があります。

- **ソース IP**：攻撃が発生している IP アドレス。
- **ステータス**：攻撃の現在のステータス。
  攻撃のステータスは、攻撃内の攻撃イベントで最も深刻度が高いものによって決定されます。攻撃内に**攻撃検出済**のステータスのイベントがある場合、その攻撃のステータスは**攻撃検出済**になります。「攻撃検出済」のイベントがない場合、ステータスは次に高い深刻度のイベントのステータスになります。 深刻度の順序は、最も高い順に次のようになります。
  - **攻撃検出済**：Contrast が明らかな攻撃を検知して確定したとしても、該当するルールのモードが**監視**に設定されている場合、そのリクエストはブロックされません。
    このステータスは、Contrast が攻撃が発生したと確定したルール(入力トレース)にのみ適用されます。Contrast で攻撃の発生が確定されるのは、ペリメータで高い信頼性のしきい値に達した攻撃、またはシンクで監視· 確認された攻撃です。
  - **疑わしい**：Contrast が攻撃を検知して、該当するルールによって攻撃が疑わしいと報告されたとしても、そのルールのモードが**監視**に設定されている場合、そのリクエストはブロックされません。このステータスは、Contrast が攻撃が発生したことを確証できないルール(非入力トレース)に適用されます。
  - **ブロック済**：Contrast が攻撃を検知し、該当するルールのモードが**ブロック**に設定されている場合、リクエストがブロックされます。
  - **ブロック済(P)**：このステータスは、「ペリメータでブロック」と「ブロック」の両方のモードをサポートするルールに適用されます。
    アプリケーションでリクエストが処理される前に Contrast が攻撃を検知し、該当するルールが**ペリメータでブロック**に設定されている場合、リクエストがブロックされます。
    ルールのモードが**ペリメータでブロック**に設定されていたとしても、ペリメータではない攻撃を Contrast が検知した場合には、 リクエストはブロックされ、ステータスは**ブロック済**になります。
  - **探査検出**：Contrast が攻撃を検知しても確定せず、該当するルールのモードが**監視**に設定されている場合、攻撃はブロックされません。 確定されない攻撃とは、ペリメータで高い信頼度のしきい値まで達しなかった攻撃であり、シンクで監視はされても検知されなかった攻撃です。
- **アプリケーション**：攻撃が行われている間に、その IP アドレスからの攻撃イベントが確認された全てのアプリケーション。
- **サーバ**：攻撃が行われている間に、その IP アドレスからの攻撃イベントが確認された全てのサーバ。
- **ルール**：攻撃が行われている間に、その IP アドレスから発生した全ての攻撃の種類。
- **開始**：攻撃の時間範囲内で、その IP アドレスから検知された最初の攻撃イベントのタイムスタンプ。
- **終了**：攻撃の時間範囲内で、その IP アドレスから検知された最後の攻撃イベントのタイムスタンプ。
- **イベント**：攻撃を構成する攻撃イベントの数。

攻撃の監視

Contrast Web インターフェイスで、現在および過去の攻撃の概要、攻撃者の IP アドレス、攻撃の種類、攻撃が検出されたアプリケーションなどを確認できます。

1. Contrast Web インターフェイスのナビゲーションバーで**攻撃**を選択します。
2. 攻撃者にソース名 (810ページ)がある場合は、その上にカーソルを合わせると、関連する IP アドレスのリストが表示されます。
   攻撃者にソース名がない場合、攻撃者のアバターにクエスチョンマークが表示されます。攻撃者がアプリケーションの不正利用に成功した場合、アバターが赤色になります。

> **注記**
> 攻撃イベントについて報告されたデータが複数のソース名と一致する場合、最後に更新された名前が適用されます。

攻撃者の情報を参照するには、IP アドレスまたはソース名をクリックします。

3. 攻撃の表示は、以下の方法でフィルターをかけることができます。
   - 日付範囲と環境をフィルターで指定する
   - 攻撃者の IP やソース名で攻撃一覧を検索する
   - 影響を受けたアプリケーション、Assess や Protect のルールで検索する
   - **探査検出も表示**のチェックボックスをオンにして、「探査検出」された攻撃イベントの情報も表示する
4. **攻撃イベント**タブには、検知された攻撃の種類の一覧と、種類ごとの攻撃イベントの合計数が表示されます。
5. **対象アプリケーション**では、攻撃の対象となった各アプリケーションを確認できます。

攻撃の管理

開始する前に

アプリケーションをホストしているサーバで Protect が有効になっていることを確認します。Protect が有効な場合、Contrast Web インターフェイスのナビゲーションバーに「攻撃」タブが表示されます。

手順

以下の手順に従って、攻撃および攻撃イベントに対して各操作を実行します。

1. 攻撃・攻撃イベントを表示する
   a. Contrast Web インターフェイスのナビゲーションバーで**攻撃**を選択します。

   b. **攻撃**タブまたは**攻撃イベント** タブを選択します。
2. (オプション)攻撃・攻撃イベントにタグを付ける
   攻撃や攻撃イベントにタグを付けると、整理ができ検索しやすくなります。
   a. 1 つ以上の攻撃または攻撃イベントを選択します。
   b. 一覧の上にあるタグアイコン( )を選択します。
   c. 表示される画面で、1 つ以上のタグ名を入力します。
3. 攻撃・攻撃イベントを消去する
   攻撃の消去を選択すると、攻撃および関連するイベントがビューから削除されます。攻撃または攻撃イベントを消去するには、以下の手順を行います。
   a. 攻撃や攻撃イベントの一覧より、1 つまたは複数の行のチェックボックスをオンにし、**攻撃を消去**または**イベントを消去**アイコン( )をクリックします。
   または、行の最後にある矢印をクリックして、ドロップダウンメニューを表示し、**攻撃を消去**または**イベントを消去**を選択します。
   b. **消去**をクリックします。
4. IP アドレスをブロックする
   このオプションは、指定した IP アドレスからのアクセスをブロック (809ページ)するものです。IP アドレスをブロックすると、以後はその IP アドレスからの不要なアクセスを防ぐことができます。
   a. 攻撃または攻撃イベントの行の右端にあるドロップダウン( )を選択します。
   b. メニューから **IP を拒否リストに登録**を選択します。
   c. 名前のフィールドに、指定した IP アドレスをブロックするためのポリシー名を入力します。

d. ブロックの有効期限を選択します。
e. **保存**をクリックします。

5. 例外を追加する(攻撃イベント)
   アプリケーションに例外を追加 (804ページ)することにより、特定のアプリケーションまたはその一部をセキュリティ検査の対象から外すことができます。
   これは、Java、.NET Framework、.NET Core、Python、Node.js、Go、Ruby エージェントを使用している場合に利用できます。
   a. 攻撃イベントの一覧で、攻撃イベントの行の右端にあるドロップダウン(˅)を選択します。
   b. **例外を追加**を選択します。
   c. 例外の名前を入力します。
   d. 例外の種類を選択し、その種類に関する詳細情報を入力します。
   e. 例外を適用するルールを選択します。
      対象ルールのフィールドをクリックすると、ルールが一覧表示されます。

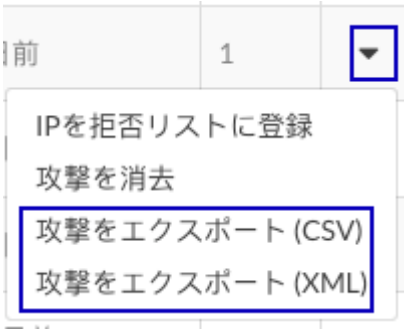
   f. (オプション)例外と一致する全てのイベントを消去するには、チェックボックスをオンにします。
   g. **追加**をクリックします。

6. 仮想パッチを作成する(攻撃イベント)
   仮想パッチ (801ページ)とは短期的なカスタムの防御ルールで、コード内で新たに検出された特定の脆弱性に対して防御するためのものです。
   a. 攻撃イベントの一覧で、攻撃イベントの行の右端にある矢印をクリックします。
   b. **仮想パッチを作成**を選択します。
   c. 仮想パッチを追加 (801ページ)するための画面で、仮想パッチの詳細を入力します。
   d. **保存**をクリックします。

7. Protect ルールのモードを指定する(攻撃イベント)
   Protect ルール (792ページ)により、アプリケーション環境における特定の種類のサイバー攻撃を監視またはブロックすることができます。
   a. 攻撃イベントの一覧で、イベントを選択します。
      イベントを選択すると、攻撃イベントの詳細が表示されます。
   b. イベント名の横にある設定アイコン(⚙)を選択します。
   c. 必要に応じて、**モードを変更**や**現在のモード**、または特定の環境を選択して、モードを変更します。

d. それぞれの環境で必要なモードを選択してください。

8. 攻撃データを保存する
   Contrast では、攻撃イベントのデータは 30 日間保持されてから、削除されます。データの保存には、いくつかの方法があります。
   - Syslog (596ページ)にデータを出力する。
   - Generic Webhook (750ページ)を設定する。
     Generic Webhook は、POST メッセージを受信するあらゆる URL に通知できます。
   - データを CSV や XML ファイルにエクスポートする。
     攻撃の一覧で、攻撃の行の右端にある矢印をクリックし**攻撃をエクスポート(CSV)**または **攻撃をエクスポート(XML)**を選択します。

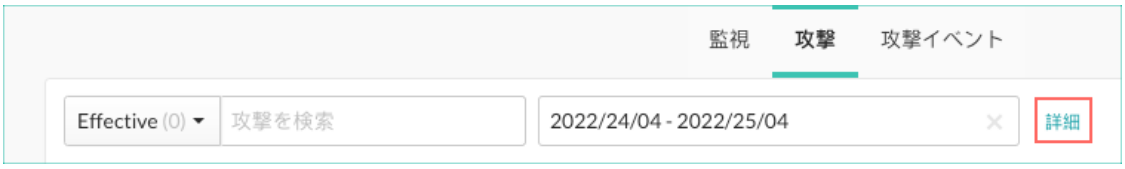


攻撃にタグを付ける

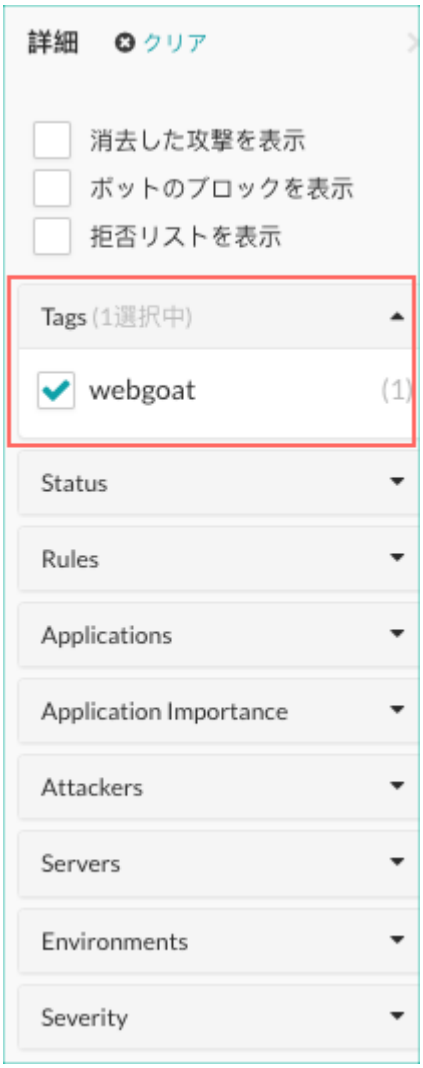
タグを使用すると、特定の攻撃や攻撃イベントを見つけやすくなります。

手順

1. 攻撃または攻撃イベントにタグを付けます。
   a. Contrast Web インターフェイスのナビゲーションバーで**攻撃**を選択します。
   b. **攻撃**または**攻撃イベント**を選択します。
   c. 1 つ以上の攻撃または攻撃イベントの横にあるチェックボックスを選択し、タグのアイコン (🏷)を選択します。
   d. 「攻撃にタグ付け」または「攻撃イベントにタグ付け」の画面で、既存のタグを選択するか新規に作成します。
   e. **保存**を選択します。
2. タグを使用して攻撃や攻撃イベントを検索します。
   a. 攻撃ページまたは攻撃イベントページで、検索ボックスの横にある**詳細**を選択します。



   b. Tags セクションをクリックして展開します。



c. 1 つまたは複数のタグの横にあるチェックボックスを選択します。
表示が変わり、選択したタグのある攻撃または攻撃イベントが表示されます。

攻撃スクリプトの実行

Contrast で攻撃データがどのようにキャプチャされるかを確認したい場合は、オープンソースの Web 脆弱性スキャナである Nikto を使って、攻撃スクリプトを実行してみましょう。

注記

攻撃スクリプトを実行するには、Contrast エージェントがインストール済 (46ページ)で、Contrast Protect が有効な (815ページ)アプリケーションが必要です。

攻撃スクリプトを実行するには：

1. ターミナルで `./nikto.pl` を実行し、Nikto が正しく設定されていることを確認します。正しく設定されていれば、デフォルトのヘルプメッセージが返されます。
2. Contrast で、Nikto を実行するマシンの IP アドレスがブロックリストに登録されていないことを確認します。
3. ターミナルで、*program* ディレクトリに移動します。

4. 以下を実行してスキャンを開始します。

```
./nikto.pl -useragent “MyAgent (Demo/1.0)” -h http://www.your-site.com
```

> **注記**
> Web アプリケーションのファイルが特定のディレクトリにある場合は、-r オプションを使用してディレクトリを追加します。

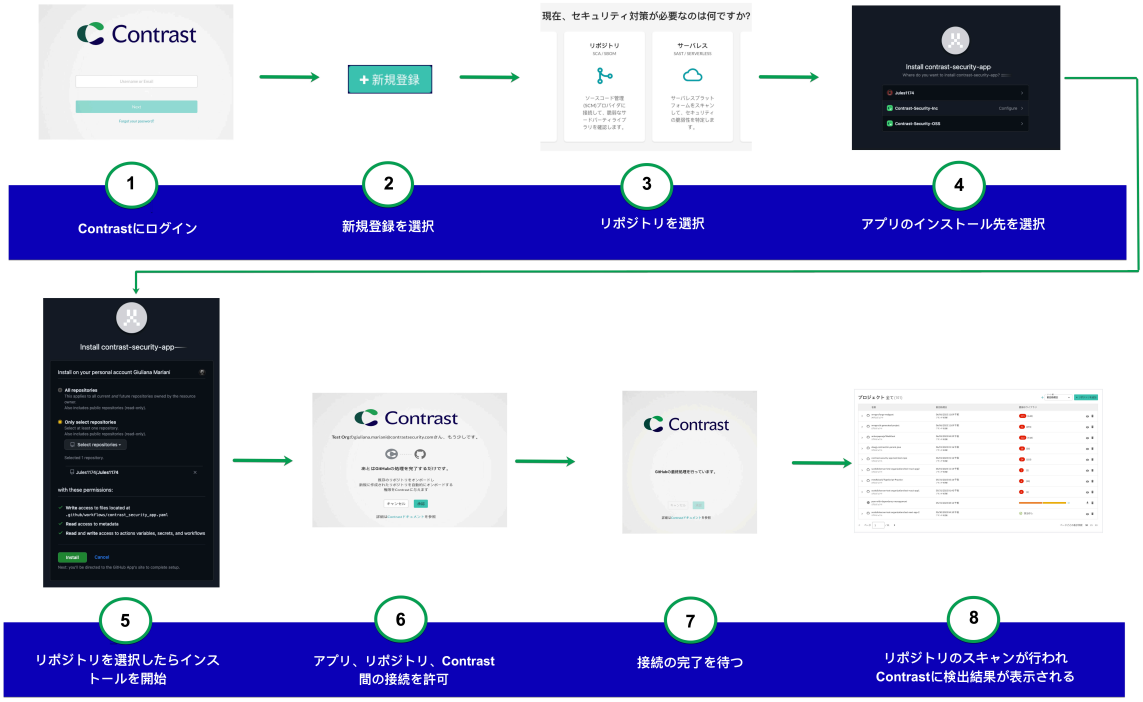
5. スクリプトの実行が完了すると、Contrast Web インターフェイス内や E メールで新しい攻撃に関する警告が通知されます。
6. 警告を選択するか、**攻撃**ページにアクセスし、攻撃の概要を確認します。個々の攻撃の詳細は、**ソース IP** を選択します。

Contrast Security GitHub アプリ

Contrast Security の GitHub アプリ(GitHub Marketplace では **Contrast Security SCA** とも呼ばれます)を使用すると、GitHub リポジトリを Contrast でスキャンできます。脆弱なライブラリの検出と修正方法についてのガイドや、CI/CD の自動化により、コード内のリスクを早い段階で回避できるようになります。

使い方

初めて使用する場合は、Contrast にログインして GitHub アカウントを Contrast に接続します。それから、リポジトリ内のライブラリの脆弱性をスキャンしてください。



接続してスキャンを実行したら、Contrast Web インターフェイスのプロジェクト (524ページ)ページで検出結果を確認できます。

また、GitHub Marketplace から (717ページ)Contrast Security GitHub アプリで接続することもできます。

このアプリでは、以下のことが可能です。

- GitHub リポジトリをスキャンできます。
- 依存関係のセキュリティ解析を自動化して、テスト環境や本番環境での検知や悪用後ではなく、コードレビュー中に脆弱性が検出されて対策を取ることができます。
- デフォルトブランチへのコミットや、デフォルトブランチにマージするために作成された PR があれば、ワークフローのファイルがトリガーされます。また、ワークフローを手動でトリガーすることも可能です。
- 編集(Edit)、ルール管理者(Rules Admin)、または管理者(Admin)権限のあるユーザが、アプリにアクセスできます。

Contrast Security GitHub アプリでサポートされる言語

Contrast Security GitHub アプリは、以下の言語をサポートしています。

- Java
- Node.js
- JavaScript
- Python
- PHP
- Ruby

サポート対象のバージョンの詳細については、Contrast CLI のサポート対象言語とパッケージマネージャ (666ページ)のページをご覧ください。

インストールと認証

GitHub アカウントと接続することで、脆弱なサードパーティのライブラリを確認できます。接続したら、PR のサマリーやトリガーを解析するために監視し、それらを Contrast Web インターフェイスに表示することができます。

注記

Contrast で GitHub の脆弱性を監視する方法は、もう 1 つあります。GitHub と Contrast をインテグレーション (755ページ)することで、プロジェクトの脆弱性を確認できます。

Contrast と GitHub のシークレット

画面に READ/WRITE のシークレットがあっても、Contrast は GitHub のシークレットを読み取ることはできません。あくまでもトークン名の読み出しだけになります。

GitHub でシークレットを作成するには、Contrast アカウントから次の資格情報が必要です。Contrast Web インターフェイスで、**ユーザメニュー** > **ユーザの設定** > **プロファイル**で確認できます。

- API キー(CONTRAST_API_KEY)
- 組織 ID(CONTRAST_ORGANIZATION_ID)
- 認証ヘッダー(CONTRAST_AUTH_HEADER)

また、エージェントが報告する Contrast サーバのアドレス(CONTRAST_API_URL)が必要です。デフォルト：https://app.contrastsecurity.com

Contrast Security GitHub アプリは、ワークフローで使用するリポジトリのシークレットと変数、アクションを作成し、正しい Contrast アカウントに検出結果が送信されるようにします。PR のクローズには、これらのシークレットと変数を手動で削除する必要があります。シークレットと変数は、GitHub アカウントの`/settings/secrets/actions` ページにあります。

開始する前に

- Contrast Web インターフェイスにアクセスできることを確認してください。
- GitHub アカウントにログインしていることを確認してください。

手順

Contrast Web インターフェイスからインストールする場合：

1. Contrast Web インターフェイスにログインし、右上の**新規登録**を選択します。
2. **GitHub に接続**を選択し、**次へ**をクリックします。
3. インストール画面で、すべてのリポジトリに接続するか、特定のリポジトリに接続するかを選択します。
4. Contrast のホストドメインの URL を入力します。例：https://app.contrastsecurity.com
5. **インストール**を選択して、 Contrast と GitHub 間で接続が確立されるのを待ちます。
6. **承認**を選択して接続を確定して、 リポジトリをオンボードします。

完了すると、プロジェクト (524ページ)のページでリポジトリの検査結果を確認できます。

GitHub Marketplace からインストールする場合：

1. GitHub Marketplace で、Contrast Security GitHub App にアクセスします。
2. **Install it for free**(無料でインストール)を選択します。
3. 手順に従って、Contrast の GitHub アプリをインストールし、リポジトリと接続します。

完了すると、プロジェクト (524ページ)のページでリポジトリの検査結果を確認できます。

GitHub リポジトリの追加または削除

Contrast にリポジトリを追加する、または Contrast からリポジトリを削除します。

> **注記**
> リポジトリを追加するには、少なくとも組織の編集(Edit)ロールが必要です。

リポジトリを追加

Contrast Web インターフェイスのプロジェクトページにある**リポジトリを追加**ボタン、または GitHub から Contrast にリポジトリを追加できます。

Contrast からリポジトリを追加するには：

1. Contrast Web インターフェイスのナビゲーションバーで**プロジェクト**を選択します。
2. 画面の右上にある**リポジトリを追加**ボタンをクリックします。

3. GitHub アカウントにログインし、 Contrast に接続するリポジトリを選択します。

リポジトリが追加されると、Contrast でリポジトリのスキャンが実行され、検出結果がプロジェクト (524ページ)の一覧に表示されます。

GitHub からリポジトリを追加するには：

1. Github で、**Settings** > **Applications** > **Integrations** にアクセスし、**Contrast Security GitHub App** を探します。

2. **Repository access**(リポジトリアクセス)で、Contrast に接続するリポジトリを選択します。
3. **Save**(保存)をクリックします。

リポジトリが追加されると、Contrast でリポジトリのスキャンが実行され、検出結果がプロジェクト (524ページ)の一覧に表示されます。

リポジトリを削除

Contrast のプロジェクト一覧からリポジトリを削除するには：

1. Contrast Web インターフェイスのナビゲーションバーで**プロジェクト**を選択し、プロジェクト一覧から削除する行を探します。
2. 処理の列にある**削除**アイコンをクリックします。

この操作によって、リポジトリとその中のすべての関連プロジェクトが削除されます。これは、Contrast Security GitHub アプリからリポジトリを***削除するわけではない***ことに注意してください。このリポジトリに対する Contrast の権限も削除したい場合は、適切にクリーンアップするために、ここでリポジトリを削除した***後に***、GitHub アプリのインストール設定にてアクセスも削除する必要があります。

リポジトリへのアクセスを削除して、GitHub から Contrast Security GitHub アプリをアンインストールします。Github で、**Settings** > **Applications** > **Integrations** にアクセスし、**Contrast Security App** を探します。**Uninstall**(アンインストール)を選択します。

トラブルシューティング

GitHub アプリでインストールとオンボードを完了できない場合は、以下のオプションをお試しください。

- **Cookie を確認します。**`ONBOARDING_SESSION` という Cookie を手動でクリアしてください。
- **アプリを再インストールします。**ブラウザを閉じて、アプリをアンインストールし、その後、アプリを再インストールしてください。

ワークフローを実行できない場合は：

- **GitHub で *Actions permissions* を確認します。**GitHub アカウントの Settings で、**Disable actions** オプションが選択されていないことを確認してください。 このオプションが選択されている場合、ワークフローが実行されず、解析は行われません。

レポート

Contrast の全てのレポートは、ソフトウェア部品表(SBOM)以外はタイムスタンプ付きの PDF として ローカルにダウンロードできます。

以下のレポートを作成できます。

- コンプライアンス対応レポート (718ページ)
- DISA STIG Viewer チェックリスト (720ページ)
- ソフトウェア部品表(SBOM) (721ページ)
- 修復の概要レポート (724ページ)
- 脆弱性の傾向レポート (722ページ)
- 組織の統計値 (723ページ)

コンプライアンス対応レポート

コンプライアンス対応レポートは、脆弱性に対応したことを証明するもので、アプリケーションの最新の情報に基づいて PDF 形式で作成されます。この PDF レポートによって、コンプライアンスや監査要件に対応できます。

2023 年 11 月 7 日以降、本レポートが旧来のセキュリティ基準レポートに代わるものとなります。コンプライアンス対応レポートは、セキュリティ基準レポートと同様の情報を提供します。コンプライアンス要件への対応や緊急な対策が必要な箇所を特定するのに役立ちます。

注記
このレポートは作成してから 7 日後に期限切れになります。この期限が切れるとレポートは Contrast サーバから削除されます。

コンプライアンス対応レポートには、以下の情報が含まれます。

- レポート作成時に使用したフィルタの設定項目の一覧
- アプリケーションのセキュリティ対応状況の概要
- カスタムコードおよびオープンソースライブラリに対する脆弱性の評価。既存の組織で CVSS 3.1 が有効になっていない場合、オープンソースの脆弱性に重大(critical)の深刻度は表示されないことに注意してください。これを有効にするには、サポートにお問い合わせください。
- セキュリティ評価の指標としてのルートカバレッジ
- セキュリティ基準の評価(オプション)、アプリケーションでオープン中の脆弱性の詳細情報(オプション)
- 評価方法や用語を説明する付録

開始する前に

コンプライアンス対応レポートには、以下の制限があります。

- 1,350 件の脆弱性(詳細を含める場合)
- 18,000 件の脆弱性(詳細を含めない場合)
- 15,000 ルート(観測情報を含める場合)
- 30,000 ルート(観測情報を含めない場合)

レポートが上記の制限を超えると、エラーメッセージが表示され、レポートは生成されません。その場合は、レポートの選択項目を変更して、レポートの情報量を減らしてください。

コンプライアンス対応レポートの作成手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーションの一覧からアプリケーション名をクリックします。
3. アプリケーションページの右上にある**レポート**アイコン(📊)をクリックします。
4. ドロップダウンで、**コンプライアンス対応レポートを生成**を選択します。
5. 「コンプライアンス対応レポート」の画面で、レポートに含める**脆弱性**、**環境**およびオプションで**セキュリティ基準**を選択します。

デフォルトで、全ての脆弱性と全ての環境が選択されていますが、フィールドをクリックして選択項目を絞り込むことができます。生成されるレポートにセキュリティ基準のセクションを含める場合は、**セキュリティ基準**から追加したい基準を選択します。
オプションで、オープン中の脆弱性の情報と観測されたルートの情報を含めるかを選択できます。
次の表は、コンプライアンスレポートの作成時に指定できるカテゴリの概要です。

カテゴリ	デフォルト	フィルターオプション
脆弱性	全て	• ステータス(報告済、疑わしい、確認済、問題無し、修復済、修正完了、修復済 - 自動検証) • 深刻度(注意、低、中、高、重大) • Assess ルール
脆弱性の詳細	含めない	脆弱性の詳細を含めるにはチェックボックスを選択
ルートの観測情報	含めない	ルートの観測情報を含めるにはチェックボックスを選択
環境	全て	• 開発環境 • QA • 本番環境
セキュリティ基準	指定なし	• DISA ASD STIG • IPA - 7.0 • OWASP 2013 Top 10 • OWASP 2017 Top 10 • OWASP 2021 Top 10 • OWASP Top10 API 脆弱性 2019 • PCI DSS - 2.0 • PCI DSS - 3.0 • PCI DSS - 3.2.1 • PCI DSS - 4.0

6. **作成**を選択します。
   Contrast でレポートが生成されると、**通知**パネルにダウンロードリンクが表示されます。
   リンクを選択すると、レポートをダウンロードできます。

DISA STIG Viewer チェックリスト

アメリカ国防情報システム局(DISA)のセキュリティ技術導入ガイド (STIG) は、全ての政府機関のアプリケーションセキュリティの評価基準となっています。STIG は、アプリケーションのセキュリティを保証するために、アプリケーションのライフサイクルを通じて使用されることが推奨されています。Contrast のレポート機能で、アプリケーションで検出された脆弱性に関して、STIG の要件に反する脆弱性一覧を作成することができます。

重要
DISA STIG レポートを生成するには、アプリケーションに Assess ライセンスが必要です。

DISA STIG のレポート作成を実行する前に、システム管理者(SuperAdmin 権限のあるユーザ)がオプションを有効にする必要があります。ユーザメニューで **SuperAdmin** を選択して、ナビゲーションバーで**組織**を選択します。DISA STIG のオプションを有効にする組織の名前をクリックすると、組織の編集画面が表示されます。表示される画面で、**DISA STIG チェックリストのレポート作成を有効にする**のボックスをオンにして、**保存**を選択します。

STIG Viewer で、セキュリティ基準レポート用に複数の STIG を指定してカスタムチェックリストを作成します。Contrast でアプリケーションの脆弱性に関する DISA STIG レポートを作成するには、アプリケーションのチェックリストをインポートする必要があります。

STIG Viewer チェックリストを作成するには：

1. **アプリケーション**ページにアクセスして、アプリケーション名をクリックします。
2. アプリケーションの**概要**ページで、レポートアイコンをクリックして、**STIG Viewer チェックリストを作成**を選択します。
3. 表示される画面で、STIG Viewer チェックリスト(.ckl)ファイルをインポートします。このファイルは、STIG Viewer アプリケーションからエクスポートされたチェックリストである必要があります。
4. **作成**をクリックし、更新された STIG Viewer チェックリスト(.ckl)ファイルをダウンロードします。

SBOM(ソフトウェア部品表)

ソフトウェア部品表(SBOM)は、政府のセキュリティ規制に準拠するために必要となる場合があります。

SBOM は、Contrast Web インターフェイスで生成したり、簡単な API や Contrast コマンドラインインターフェイス(CLI)を使用して生成できます。

Contrast の SBOM は、OWASP の CycloneDX SBOM 規格および国際標準の SPDX 形式に準拠しています。Contrast の SBOM には、アプリケーションが使用するソフトウェアに関する次のような情報が含まれます。

- ライブラリ - コード本体に存在するオープンソースおよびサードパーティコンポーネント
- ソフトウェアコンポーネントに適用されているライセンス
- コード本体で使用されているソフトウェアコンポーネントのバージョン

注記
現在、CycloneDX v1.4 と SPDX 2.2 をサポートしています。

また、Contrast の SBOM は、米国商務省電気通信情報局(NTIA)の要件も満たしています。データ作成者、サプライヤー名、コンポーネント名、コンポーネントのバージョン、依存関係、タイムスタンプ、その他のユニーク ID(PURL やパッケージの SPDX 識別子など)が含まれます。

開始する前に

- Contrast Web インターフェイスからエクスポートする場合には Contrast Assess ライセンスが必要

- サポートされている言語：Java、.NET Framework、.NET Core、Node.js、Python、Ruby、Go、PHP

手順

SBOM レポートを生成するには、3 つの方法があります。

1. **Contrast インターフェイスから生成：**
   a. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   b. アプリケーションのページの右上にある**レポート**アイコン( ) をクリックします。
   c. ドロップダウンで、**SBOM(ソフトウェア部品表)を生成**を選択すると、レポートが作成され自動的にダウンロードされます。CycloneDX と SPDX 形式に対応しています。
2. **API を使用してレポートを生成：**
   a. CycloneDX の場合：**GET**`<HOST>/Contrast/api/ng/<ORG_ID>/applications/<APP_ID>/libraries/sbom/cyclonedx` リクエストをします。
   b. SPDX の場合：**GET**`<HOST>/Contrast/api/ng/<ORG_ID>/applications/<APP_ID>/libraries/sbom/spdx` リクエストをします。

   API の使用についての詳細は、REST API を参照してください。
3. **CLI でレポートを生成：**
   - `--save` オプションを使用します。`--save cyclonedx` または`--save spdx` で形式を選択できます。詳細は、CLI コマンド (675ページ)を参照してください。

> **注記**
> - 現在、CLI での.NET のサポートには制限があります。
> - 静的 SCA (600ページ)の検出結果で SBOM を作成するには、CLI を使用します。
> - CLI を使用して作成された SBOM には、CLI で登録されたライブラリデータのあるアプリケーションのクラス利用状況の情報が提供されます。

脆弱性の傾向レポート

脆弱性の傾向レポートを使用して、アプリケーションで直面している脆弱性とそれらがどのように管理されているかを認識できます。

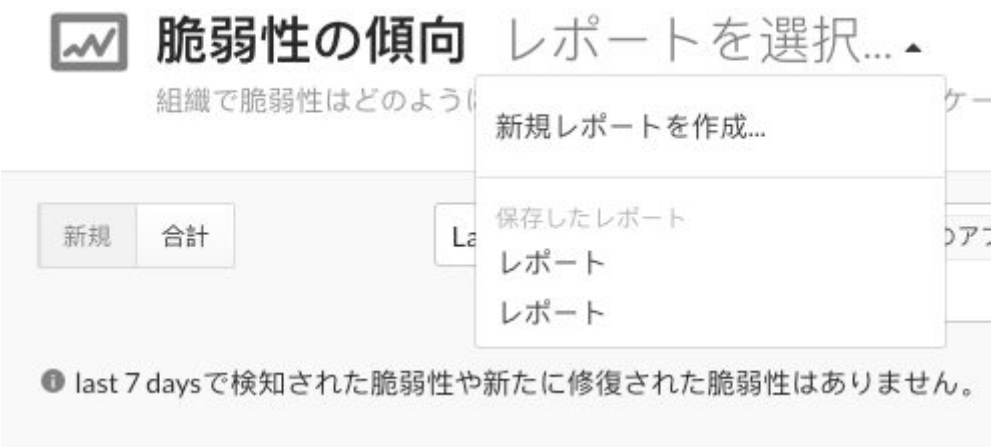
脆弱性の傾向レポートを表示するには：

1. ユーザメニューで、**レポート**を選択します。**表示**を選択すると、詳細がグラフで表示されます。
2. **新規**を選択すると、新規に検出された脆弱性のグラフが表示されます。**合計**を選択すると、報告されているすべての脆弱性と修復されたすべての脆弱性を比較したグラフが表示されます。
   黒色の各データポイントは、その日付で発生した脆弱性の総数を表しますが、ステータスが疑わしい、確認済、報告済の脆弱性の総数になります。緑色の各データポイントは、ステータスが**問題無し**、**修復済**または**修正完了**の脆弱性の総数を表します。各データポイントにカーソルを合わせるとステータスの内訳が表示されます。
3. レポートのデフォルトは、全てのアプリケーション、全てのサーバ、および全てのルールです。グラフの上の各フィールドをクリックすれば、表示する脆弱性にフィルターを適用できます。次の表は、カスタムレポート作成時に指定できるカテゴリの概要です。

フィールド	デフォルト	フィルターオプション
日付	過去 7 日間	過去 30 日間 過去 12 週間 過去 12 か月間
アプリケーション	全て	重要度(重大、高、中、低、重要でない) アプリケーションのタグ ライセンスあり(全アプリケーションの一覧)

フィールド	デフォルト	フィルターオプション
サーバ	全て	環境(開発、QA、本番) サーバのタグ サーバ(全サーバの一覧)
ルール	全て	深刻度(重大、高、中、低、注意) 脆弱性のタグ 脆弱性ルール(全ルールの一覧)

4. **表示**を選択し、右上にある**レポートを保存**アイコンを選択すると、このレポートのフィルター条件が保存されます。表示される画面でレポート名を入力したら、**保存**を選択します。レポートはユーザ別に保存されるため、各ユーザが脆弱性の傾向レポートを独自に定義して保存できます。これらのレポートはいつでも編集や削除ができます。
   保存したレポートを表示中にフィルターオプションを変更すると、星印のアイコンが未保存の状態に変わり、レポート名の横に**編集済**と表示されます。同じアイコンを使用して、**既存レポートを保存**または**新規レポートとして保存**することができます。**既存レポートを保存**を選択すると、保存したレポートが現在のフィルターで更新され、**編集済**のステータスが消えます。**新規レポートとして保存**を選択すると、現在のフィルターで表示中のレポートが別の名前で新しいレポートとして保存されます。
   削除をクリックすると現在参照中のレポートは完全に削除されます。
   既存のレポートで保存していない編集内容を消去してレポートのデフォルト設定からやり直す場合は、ドロップダウンで**新規レポートを作成**を選択します。
   保存したレポートが 5 つを超えると、**保存したレポート**のドロップダウンに**管理**のリンクが表示されます。「管理」を選択してこの画面を開きます。ここではレポート名の変更(名前をクリックして編集)やレポートの検索が行えるほか、各レポートの横にあるチェックボックスを使用(または**全て選択**のチェックボックスを使用)して削除するレポートを選択できます。



5. また、脆弱性の傾向をタイムスタンプ付きの PDF レポートとして作成し、脆弱性管理のスナップショットを取ることができます。ページの右上にある**エクスポート**アイコンを選択します。デスクトップにレポートがダウンロードされます。この PDF レポートには、カスタマイズしたビューに含まれる値の概要、傾向グラフ、各データポイントの指標と内訳の表が含まれます。

組織の統計値

ユーザメニューから**レポート**を選択すると、脆弱性の傾向 (722ページ)や組織レベルで次の情報を確認できます。

- **ライセンス**: ライセンスのグラフには、組織内の Access と Protect の合計ライセンス数、およびライセンスのないアプリケーション数とサーバ数が表示されます。
- **アプリケーション**：アプリケーションのグラフでは、内側の円は言語別の内訳を表します。外側の円は、ドロップダウンから**テクノロジ**または**スコア**を選択して、比較したい分類を選択します。
- **サーバ**：ドロップダウンから**コンテナ**または**環境**を選択し、値の表示方法を選択します。

ドロップダウンのフィルターを使用して、比較するためのデータを簡単に選択できます。

各グラフで**表示**を選択すると、以下のような詳細情報が表示されます。

- **ライセンス：アクティビティ**には、過去 1 年間のライセンス消費に関するアクティビティの傾向グラフが表示されます。
  Assess または Protect の傾向グラフ線のデータポイントにカーソルを合わせると、各月で使用されたライセンス数を確認できます。点線は購入されたライセンス数を示します。
  グラフの縦線をクリックすると、各日の Protect ライセンスの使用時間が表示されます。最大使用時間は、縦線の一番上に明るい緑色が掛かって表示されます。ライセンスアクティビティデータの表示に戻るには、**ライセンスのアクティビティに戻る**を選択します。
  アクティビティグラフの下にある **Protect の使用状況を表示**すれば、Protect に関する当月のデータと使用状況の統計が表示されます。別の月のデータを表示するには、ドロップダウンを使用します。
  使用量には、Assess と Protect のライセンスに関する棒グラフとタイムラインがあります。棒グラフには、購入したライセンスの合計数と比較して、使用中のライセンス数が表示されます。タイムラインには、特定の日付で期限が切れるライセンスの数が表示されます。
  右側の円グラフで、Assess と Protect の割合と使用率の内訳を参照できます。組織に Protect または Assess ライセンスが無い場合は、ライセンスの無いアセットの数がグラフに表示されます。
- **アプリケーション：ステータス**には、ライセンスあり、ライセンスなし、アーカイブ済のアプリケーションごとの合計数、および組織で利用可能なライセン数があります。
  言語別内訳の円グラフは、内側の円は言語別のアプリケーション数を表し、外側の円ではテクノロジまたはスコア別にアプリケーション数が表示されます。詳細を参照するには、カーソルを合わせます。
  高リスクにはオープン中の重大な脆弱性があるアプリケーション数、**有効期限**にはライセンスの期限切れが近いアプリケーション数が表示されます。
  防御範囲には、本番サーバで防御範囲が不完全なアプリケーションの数が表示されます。詳細を参照するには、**内訳を表示**を選択します。
  最近(1 週間以内に)追加されたアプリケーション、オフラインのサーバにあるアプリケーションが、サイドバーにそれぞれ表示されます
- **サーバ：環境**で、デプロイ中の全てのサーバを環境ごとに確認できます。
  コンテナ別内訳には、指定した環境にデプロイされている各言語のサーバ数が表示されます。別の環境のデータを表示するには、ドロップダウンを使用します。
  スナップショットには、指定した環境のサーバ総数と比較して、Assess が有効なサーバ数、Protect が有効なサーバ数、およびオンラインのサーバ数が表示されます。
  右のサイドバーには、新規にオンボードされたサーバ、オフラインのサーバ、削除されたサーバ、およびライセン切れが近いサーバの一覧が表示されます。

修復の概要パッケージ

修復の概要パッケージとは、修復プロセスがどの程度適切に機能しているかを判断するのに役立つ一連のグラフを PDF 形式にして ZIP にまとめたものです。「レポート」ページの「修復の概要」セクションに、このレポートパッケージをダウンロードできるボタンがあります。

パッケージに含まれるグラフは毎月更新されます。その日付は「グラフをダウンロード」ボタンの下に表示されます。

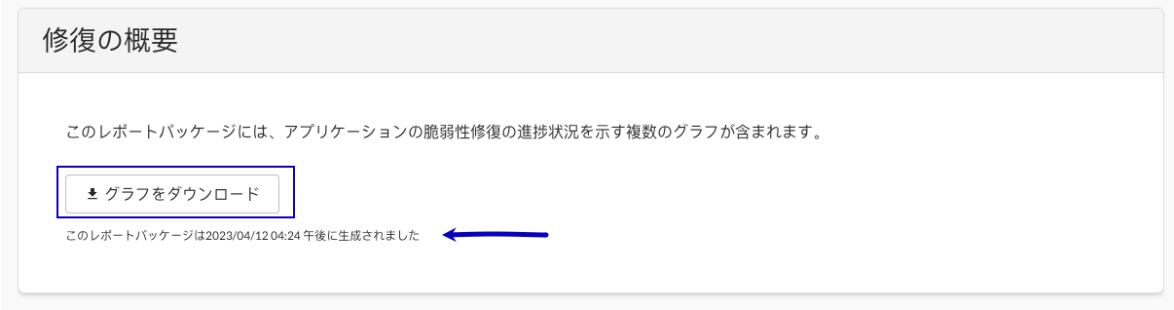
レポートパッケージには、各グラフの解釈の手引きとなる README ファイルも含まれます。

開始する前に

- 診断情報の収集 (938ページ)が有効になっている必要があります。
  診断機能が無効になっている場合、レポートページには「修復の概要」セクションは表示されません。
- SaaS 版をご利用の場合は「アプリケーションの閲覧」アクション、オンプレミス版をご利用の場合は組織の Admin ロールが必要です。

手順

1. ユーザメニューから、**レポート**を選択します。
2. 「修復の概要」セクションで、**グラフをダウンロード**を選択します。



3. ダウンロードしたい場所に、ZIP ファイルを保存します。
4. ファイルを解凍すると、README ファイルと各グラフを参照できます。

インテグレーション

ここでは、サポート対象のインテグレーションに関して、Contrast を使用する推奨方法を記載したドキュメントを提供します。サポート対象外のその他のツールやシナリオでも、Contrast と互換性がある場合があります。サードパーティのツールやテクノロジの具体的な情報については、その製品のドキュメントを参照してください。また、Contrast サポートポータルに特定の使用例や問題に対する回避策などの記事もありますので、そちらもご覧ください。

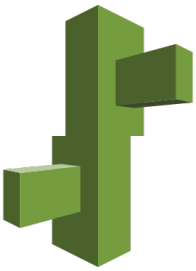
注記

サポート対象のインテグレーションについては、左側のナビゲーションメニューから詳細の説明を参照できます。完全にはサポート対象ではありませんが、互換性の可能性があるインテグレーションは、外部のドキュメントにリンクされています。Contrast サポートポータルやサードパーティサイトのドキュメントへのリンクは、アイコンが付いています。

組織で利用可能なインテグレーションを表示するには、Contrast の組織管理者の口ールが必要です。

クラウドでの連携

お使いの PaaS でアプリケーションをデプロイしたまま、Contrast を有効にしてアプリケーションを実行できます。

	**AWS Elastic Beanstalk**	AWS Elastic Beanstalk での Java エージェント (91ページ)
	AWS Security Hub	AWS Security Hub と Contrast Assess (739ページ)

	AWS Security Lake	AWS Security Lake と Contrast Assess (740ページ)
	Google App Engine	Google App Engine で Java エージェントを設定 Google App Engine で Java エージェントを設定
	Microsoft Azure	.NET エージェントと Azure App Service .NET エージェントと Azure ARM .NET エージェントと Terraform
	Red Hat OpenShift	Contrast Security コンテナ
	VMware Tanzu (旧 Pivotal Cloud Foundry)	VMware Tanzu で Java エージェントを設定 (86ページ) VMware Tanzu で Node.js エージェントを設定 (297ページ)

チャットツール

アプリケーションで新たな脆弱性が発生したり、アプリケーションに対して攻撃が行われている場合に、Contrast はリアルタイムでこれらのセキュリティの問題を検知して、いち早くお知らせします。

	Microsoft Teams	Teams とのインテグレーション (773ページ)
	Slack	Slack とのインテグレーション (775ページ)

コードリポジトリとの連携

リポジトリに GitHub Actions を追加します。

	Amazon Elastic Kubernetes Service のビルド・デプロイ	Contrast Amazon EKS Build Deploy
	Azure Kubernetes Service のビルド・デプロイ	Contrast AKS Build Deploy

	Azure Spring Cloud のデプロイ	Azure Spring Cloud Deploy
	Contrast SCA による解析	Contrast SCA Action
	Contrast Scan による解析	Contrast Scan Analyze
	アプリケーションの検証	Contrast Verify

継続的インテグレーション(CI)とビルドツール

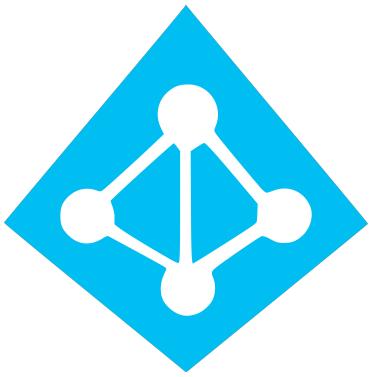
自動化されたパイプラインにアプリケーションのセキュリティゲートを追加することで、脆弱性が本番環境にデプロイされるのを防ぎます。

Azure DevOps	Azure Pipelines の拡張機能 (744ページ) Azure Marketplace
Bamboo	Bamboo プラグイン (748ページ) Atlassian Marketplace Contrast Bamboo プラグインのソースコード
CircleCI	Orb レジストリ Contrast Security Orb ソースコード
Contrast SCA による解析	Contrast SCA Action
Contrast Scan による解析	Contrast Scan Analyze
アプリケーションの検証	Contrast Verify
GitLab	GitLab パイプラインに Contrast を組み込む方法

	Gradle	Gradle プラグイン (756ページ) Gradle.org Contrast Gradle プラグインのソースコード サンプルプロジェクト
	Jenkins	Jenkins プラグイン (758ページ) Contrast Continuous Application Security Jenkins プラグインのソースコード
	Maven	Maven プラグイン (78ページ) Maven Central リポジトリ Maven プラグインのソースコード サンプルプロジェクト

エンタープライズおよび拡張

Contrast の SDK や Webhook を使用してカスタムサービスを作成したり、Webhook を使って新たな脆弱性や攻撃の検出時に通知を送信します。

	**Azure Active Directory**	Azure Active Directory シングルサインオン(SSO)との統合

	Contrast CLI	Contrast CLI (685ページ) NPM
	Java SDK	Contrast Java SDK
	JavaScript SDK	Contrast SDK Javascript NPM
	.NET SDK	Contrast SDK .NET NuGet

	Python SDK	Contrast SDK Python
	Webhook	Generic Webhook (750ページ)

IDE プラグイン

ご利用中の開発環境で、アプリケーションの脆弱性の詳細情報を表示し、修正方法や対応について知ることができます。

	Eclipse	Eclipse Marketplace
	Intellij	Contrast Intellij プラグイン (757ページ) Intellij Marketplace

	Visual Studio	Visual Studio プラグイン (776ページ) Visual Studio Marketplace
	Visual Studio Code	Visual Studio Code プラグイン (777ページ) Visual Studio Code Marketplace
	Visual Studio for Mac	Visual Studio for Mac プラグイン (778ページ) 拡張ファイル

インシデント管理システム

アプリケーションへの攻撃に対して、オンコール担当者が必要な対応を取ることができるようになります。

	PagerDuty	PagerDuty とのインテグレーション (773ページ)

	Splunk On-Call (旧 VictorOps)	VictorOps とのインテグレーション (776ページ)

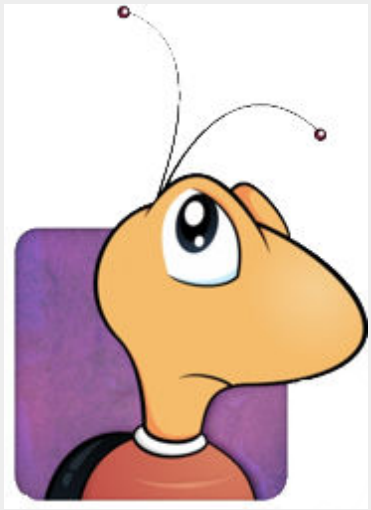
SIEM ツール

Contrast エージェントによるアクティビティから、アプリケーションセキュリティの脅威に関して運用面での新たな情報を SIEM で直接得ることができます。

	Azure Sentinel	Contrast Protect Azure Sentinel Solution Azure Sentinel
	Datadog	ソースコード
	Splunk	Splunk と Contrast を連携する方法

	Sumo Logic	ソースコード

作業管理プラットフォーム

	Azure Boards	Azure Boards とのインテグレーション (741ページ)
	**Bugzilla**	Bugzilla とのインテグレーション (749ページ)
	GitHub	GitHub とのインテグレーション (755ページ)
	Jira	Jira とのインテグレーション (767ページ)
	Kenna Security	Kenna ツールキット Contrast タスクの実行方法

	Rally (旧 CA Agile Central)	Agile Central とのインテグレーション (737ページ)
	ServiceNow	ServiceNow とのインテグレーション (774ページ)
	Solutions Business Manager (旧 Serena)	Serena Business Manager とのインテグレーション (774ページ)
	ThreadFix	Contrast Remote Provider (ThreadFix) Contrast Remote Provider

Agile Central とのインテグレーション

Contrast と Agile Central を連携することで、アプリケーションの脆弱性を自動的に追跡できます。

設定を開始する前に、以下が必要です。

- Agile Central アカウントの URL
- 対象プロジェクトで課題を作成するための権限
- HTTP 経由で Contrast にアクセス可能な Agile Central の実行インスタンス
- Contrast に登録されたアプリケーションと関連付けるプロジェクト

Agile Central と接続するには：

1. **ユーザメニュー**で、**組織の設定 > インテグレーション**にアクセスします。
2. Agile Central の行で、**接続**を選択します。
3. **Agile Central** への接続の設定画面で、バグ管理システムのエントリ名、**URL** および **API キー**を各フィールドに入力します。Agile Central の URL には、設定する Contrast インスタンスからアクセスできる必要があります。

   > **注記**
   > Agile Central の API キーを取得するには、Agile Central Application Manager にログインして、**API Keys** を選択します。Contrast では、設定したユーザ名、パスワード、Agile Central の URL は認証情報セットとして保存されます。

4. フィールドに入力したら、**接続をテスト**を選択します。接続テストでは、Contrast が Agile Central のインスタンスに到達でき、指定したユーザがログインできることを確認します。

5. 接続したら、このインテグレーションを有効にする**アプリケーション**を選択します。
6. ドロップダウンから、**プロジェクト名**と**オーナー**を選択します。
7. **デフォルトの優先順位**のセクションでは、ドロップダウンを使用して、各脆弱性の深刻度に対する優先度レベルを選択します。
8. チケットを作成する**環境**を選択します。
9. **ディフェクトの状態**を選択します。
10. チケット送信者の名前を**以下によりサブミット**で指定します。

> **注記**
> これらの設定フィールドはいずれも必須ではありませんが、空白のままにしたフィールドについては、Agile Central で独自のデフォルト値を使用してチケットが生成される場合があります。

11. Contrast で接続を設定した後に別のインテグレーションを追加するには、Agile Central の行で**設定を追加**を選択します。
12. 新たに検出される脆弱性のチケットを自動的に作成するには、設定画面でで自動作成のオプションをチェックします。表示される複数選択フィールドで、チケットを生成する脆弱性の**ルール**および**深刻度**を選択します。

> **注記**
> このオプションの選択は、過去にさかのぼってチケットを生成するものではありません。

Agile Central の認証情報の管理

Contrast には、Agile Central の設定で入力された最新の認証情報が保存されているので、次の新しい接続を簡単に設定できます。最初の設定で入力した API キーと URL の値が、それ以降のインテグレーション設定でデフォルトの認証情報となります。

以降の設定で、このデフォルトの認証情報がフィールドにあらかじめ入力されますが、必要に応じて値を変更できます。保存されている認証情報を管理して、影響を受ける全ての設定を同時に更新することもできます。

デフォルトの認証情報とは異なる認証情報で、設定を作成または編集するには：

1. **認証情報を管理**のリンクを選択します。
2. **URL** フィールドでドロップダウンを使用して保存済の認証情報セットを選択する、または **URL**、**ユーザ名**、**パスワード**フィールドの値を手動で更新します。
3. フィールドを更新したら、**接続をテスト**を選択します。
4. **保存**を選択します。新しい認証情報を使用する場合、指定した名前で既存の認証情報セットを上書きするか、別の名前で新しい値を新しい認証情報セットとして保存するかを選択する必要があります。
5. 保存済みの既存の認証情報セットを修正する場合は、必要に応じて**名前を変更**を選択します。**接続をテスト**を選択して、**保存**します。

> **注記**
> 認証情報セットを更新すると、そのセットを使用している全ての設定に影響します。

Contrast Assess と AWS Security Hub とのインテグレーション

Contrast Assess と AWS Security Hub を連携して、セキュリティに関する解析情報や検出結果を確実に AWS に直接転送できます。この効率的なインテグレーションによって、セキュリティ体制を維持・強化できます。

開始する前に

設定を開始する前に、以下が必要です。

- AWS のアカウント ID
- AWS リージョン
- 解析情報の送信元となる Contrast アプリケーション

設定

このインテグレーションを設定するには、以下の 2 つの手順を行う必要があります。

- Contrast からの検出結果を受け入れるように AWS Security Hub を設定
- AWS Security Hub に検出結果を送信するように Contrast Assess を設定

Contrast からの検出結果を受け入れるように AWS Security Hub を設定

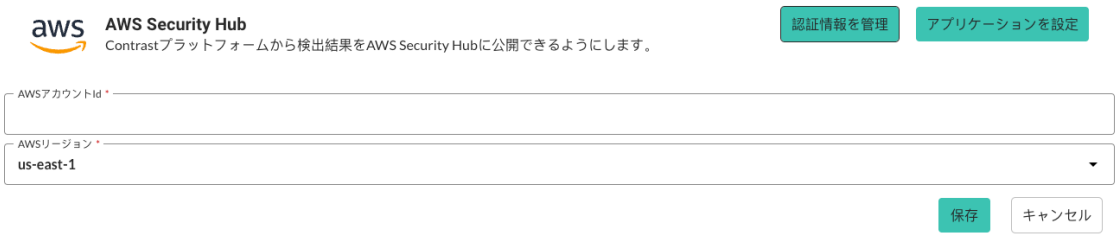
AWS Security Hub が Contrast からの検出結果を受け入れるには、以下を行います。

1. Contrast からの検出結果を受け入れる AWS アカウントとリージョンで、AWS Security Hub コンソールを開きます。
2. **統合**セクションにアクセスして、*Contrast Security* を検索します。
3. Contrast Security タイルを見つけたら、**結果を受け入れる**をクリックし、その後のプロンプトに従って設定を完了します。
4. 続けて、AWS Security Hub に送信するよう Contrast Assess を設定します。

AWS Security Hub に検出結果を送信するように Contrast Assess を設定

AWS Security Hub を設定したら、次は Contrast Assess を設定して、AWS Security Hub に検出結果を送信するようにします。

1. Contrast Web インターフェイスで、**ユーザメニュー**にアクセスし、**組織の設定** > **インテグレーション**を選択します。
2. プラットフォーム連携にある **AWS Security Hub** を選択します。
3. **認証情報を管理**を選択します。

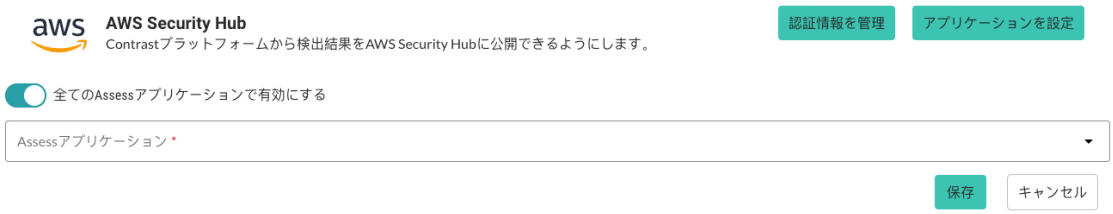


4. **AWS アカウント Id** と **AWS リージョン**を入力します。
5. **保存**を選択します。
6. 続けて、Contrast Assess のアプリケーションを設定します。

Contrast Assess のアプリケーションを設定

認証情報を設定したら、アプリケーションの設定に進みます。

1. Contrast Web インターフェイスの **AWS Security Hub** のインテグレーションセクションで、**アプリケーションを設定**を選択します。
2. 全ての Assess アプリケーションに対して AWS Security Hub とのインテグレーションを有効にするか、一覧から特定のアプリケーション名を選択して、解析情報の送信元を選択します。



3. **保存**を選択します。

再試行の仕組み

Contrast Assess と AWS Security Hub 間の同期が失敗した場合、再試行の仕組みによってデータの信頼性が確保されます。

- イベントの同期に失敗した場合、そのイベントは保存されて、GMT 時間の毎晩午前 0 時(日本標準時は GMT+09 時)に再試行されます。
- 再試行の回数は 1 回から最大 3 回までになり、最大 72 時間行われます。3 回目の再試行に失敗すると、イベントは破棄されます。
- 脆弱性作成イベントが失敗し、保存された場合、失敗したイベントに関連するその後の更新や削除の操作は、正しい状態を維持するために時系列で保存および再生されます。

Contrast Assess と AWS Security Lake とのインテグレーション

Contrast Assess と AWS Security Lake を連携して、検出結果やその他の関連するセキュリティデータを自動的にプッシュできます。

開始する前に

設定を開始する前に、以下が必要です。

- AWS リージョン

AWS でカスタムソースを作成

1. AWS で、AWS Security Lake にアクセスします。
2. **カスタムソース**を選択します。
3. **カスタムソースの作成**をクリックします。
4. 任意の**データソース名**を入力します。
5. **OCSF イベントクラス**を *Security Finding* に設定します。
6. **AWS アカウント ID** と**外部 ID** を入力します。
   これらの ID は、Contrast の**インテグレーション**ページの *AWS Security Lake* セクションで確認できます。Contrast Web インターフェイスでユーザメニューから、**組織の設定** > **インテグレーション**を選択してください。
7. カスタムソースを作成したら、生成された **AWS ロール ARN** と **S3 ARN** を取得してください。これらは、AWS への接続に必要となります。
8. 続けて、AWS Security Lake への接続を設定します。

AWS Security Lake に接続

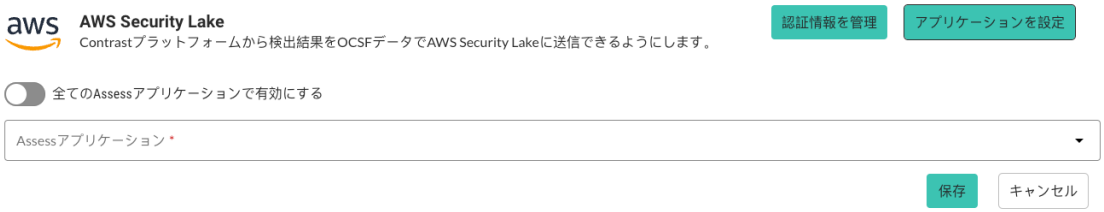
1. Contrast Web インターフェイスで、ユーザメニューにアクセスし、**組織の設定** > **インテグレーション**を選択します。
2. プラットフォーム連携にある **AWS Security Lake** を選択します。
3. **認証情報を管理**を選択します。

4. 先ほど生成された **AWS ロール ARN** と **S3 ARN** を入力します。
5. 一覧から **AWS リージョン**を選択するか、手動で入力します。
6. **保存**を選択します。
7. 続けて、Contrast Assess のアプリケーションを設定します。

Contrast Assess のアプリケーションを設定

認証情報を設定したら、アプリケーションの設定に進みます。

1. Contrast Web インターフェイスの **AWS Security Lake** のインテグレーションセクションで、**アプリケーションを設定**を選択します。



2. 全ての Assess アプリケーションに対して AWS Security Lake とのインテグレーションを有効にするか、一覧から特定のアプリケーション名を選択します。
3. **保存**を選択します。

再試行の仕組み

Contrast Assess と AWS Security Lake 間の同期が失敗した場合、再試行の仕組みによってデータの信頼性が確保されます。

- イベントの同期に失敗した場合、そのイベントは保存されて、GMT 時間の毎晩午前 0 時(日本標準時は GMT+09 時)に再試行されます。
- 再試行の回数は 1 回から最大 3 回までになり、最大 72 時間行われます。3 回目の再試行に失敗すると、イベントは破棄されます。
- 脆弱性作成イベントが失敗し、保存された場合、失敗したイベントに関連するその後の更新や削除の操作は、正しい状態を維持するために時系列で保存および再生されます。

Azure Boards とのインテグレーション

Contrast と Azure Boards を連携し、バグ管理のためのチケットの自動作成、コメントの同期、アプリケーションのプッシュ通知などを可能にします。

インテグレーションには、以下が必要になります。

- Azure Boards または TFS のアカウント認証情報：ユーザ名と個人用アクセストークン(PAT)
- PAT のスコープにワークアイテム(Work Items)の読み取りと書き込み(Read & write)を含める
- Azure Boards または TFS インスタンスがあり、HTTP で Contrast にアクセスできる
- Contrast エージェントが組み込まれたアプリケーションがあり、Azure Boards プロジェクトに関連付けられている

- 詳細については、Microsoft の Azure Boards ドキュメントを参照

関連項目

- Azure Boards への接続 (742ページ)
- チケットの自動作成 (742ページ)
- 双方向インテグレーション (743ページ)
- 個人用アクセストークンの設定 (744ページ)

Azure Boards への接続

手順

1. Contrast Web インターフェイスで、**組織の設定 > インテグレーション**を選択します。
2. Azure Boards の行で、**接続**を選択します。
3. 以下の値を入力します。
   - **名前**：Contrast の検出結果を Azure Boards のバグ管理システムに送信するときに表示されるラベル。
   - **URL**：Azure Boards または TFS の URL。Contrast からこの URL にアクセスできる必要があります。
   - **バージョン**：Contrast は API v2 を使用し、Azure DevOps Services、TFS 2015 および TFS 2017 をサポートします。
   - **パーソナルアクセストークン(PAT)**：パスワードの代わりにホストを認証するために使用します。
4. **URL をテストする**を選択します。Azure Boards または TFS プロジェクトの数によっては、数分かかる場合があります。接続テストは、入力した Azure Boards または TFS インスタンスに Contrast が接続できるか、またユーザの PAT でログインが許可されるかを確認します。
5. Azure Boards と接続したら、表示される設定画面で、バグ管理システムとの連携を有効にする**アプリケーション**を選択します。
6. **プロジェクト**と、デフォルトの**担当先**およびデフォルトの**ワークアイテム**を選択します。
7. **チーム**を選択し、そのチームにある**エリア**を選択します。これにより、特定のバックログにチケットが送信されます。
8. 脆弱性の深刻度に対する**デフォルト Priority** を設定します。この設定により、選択されたアプリケーションの脆弱性を修正するためのチケットに、深刻度に基づいて優先順位が付けられます。ここで、Contrast は API を呼び出し、Azure Boards または TFS のチケットのステータスの一覧が返ります。
9. Contrast で脆弱性のステータスを自動的に更新するための双方向のインテグレーション (743ページ)や Azure Boards でチケットを自動作成 (742ページ)するように設定することもできます。

関連項目

- チケットの自動作成 (742ページ)
- 双方向インテグレーション (743ページ)
- 個人用アクセストークンの設定 (744ページ)

チケットの自動作成

Contrast で脆弱性が新たに検出されるたびに、**チケットを自動的に作成**できます。手順は、以下のとおりです。

手順

1. Azure Boards の接続の設定画面 (741ページ)で、**新たに検知された脆弱性に対してチケットを自動的に作成する**のチェックボックスをオンにします。**ルール**と**深刻度**の複数選択フィールドが表示されます。

2. Azure Boards または TFS で新しいチケット作成のトリガーとなるルールや深刻度を選択します。デフォルトでは、**重大**と**高**が選択されます。

注記
この設定は、この選択を行った後に新規に検出された脆弱性にのみ有効です。

関連項目

- Azure Boards への接続 (742ページ)
- 双方向インテグレーション (743ページ)
- 個人用アクセストークンの設定 (744ページ)

双方向インテグレーション

Azure Boards と双方向のインテグレーションが可能です。双方向のインテグレーションにより、Azure Boards や TFS で脆弱性に関連する課題をクローズや再オープンした際に、Contrast の脆弱性のステータスが自動的に更新されます。

手順

1. Azure Boards の接続の設定画面 (741ページ)で、**双方向のインテグレーションを有効にする**を選択します。脆弱性ステータスのフィールドが表示されます。
2. ドロップダウンメニューを選択して、Azure Boards または TFS チケットの各ステータスに対応する Contrast の脆弱性ステータスを設定します。
3. **保存**を選択します。Azure Boards/TFS チケットの脆弱性ステータスが Contrast によって更新されるようになります。
4. Azure Boards や TFS でチケットのステータスが更新されると、Contrast ではその脆弱性の**アクティビティ**タブにコメントが自動的に作成されます。各コメントには、バグ管理システム名とチケットへのリンクが含まれます。

注記
Azure Boards または TFS のチケットのステータスとして、**問題無し**の脆弱性ステータスを選択する場合、Contrast で**理由**も選択する必要があります。デフォルトの値は、**上記以外**です。

注意
複数の脆弱性を 1 つの問題として、Azure Boards または TFS に送信した場合、チケットのステータスは、そのチケットに関連付けられている全ての脆弱性に適用されます。逆に、複数のチケットを 1 つの脆弱性にリンクした場合は、脆弱性を更新する前に、関連付けられているチケットを全て更新してください。例えば、チケットのステータスを **New** から **Active** に変更した場合、その脆弱性に関連付けられた全てのチケットのステータスが **Active** である場合にのみ、Contrast で脆弱性のステータスが更新されます。

関連項目

- 個人用アクセストークンの設定 (744ページ)
- Azure Boards への接続 (742ページ)
- チケットの自動作成 (742ページ)

Azure Boards の個人用アクセストークンの設定

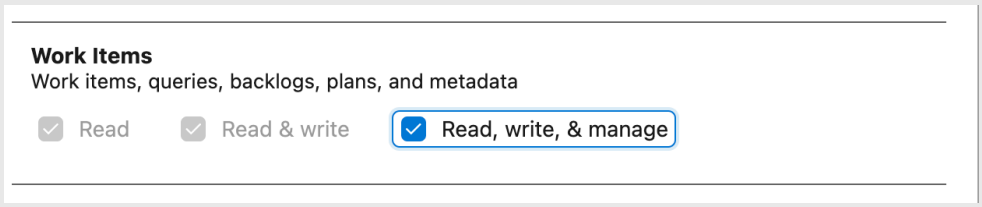
個人用アクセストークン(PAT)は、Contrast が Azure Boards にアクセスするために使用します。PAT には、フルアクセスまたはカスタムアクセスを設定できます。

手順

1. Azure で **User Settings** にアクセスし、「Profile」を選択します。
2. 「Security」で、**Personal access tokens** を選択します。既存の個人用アクセストークンがある場合は、表示されます。
3. **New Token** をクリックします。 **Create a new personal access token** の画面が表示されます。
4. **Create a new personal access token** の画面で、新しい PAT の必須フィールドを入力して、Scopes(アクセス範囲)を選択します。

注記

Contrast で使用するには、**Work Items** の **Read, write, & manage** を必ず選択してください。



5. **Create** をクリックして完了すると、新しい個人用アクセストークンが保存されます。

関連項目

- Azure Boards への接続 (742ページ)
- チケットの自動作成 (742ページ)
- 双方向インテグレーション (743ページ)

Azure Pipelines の拡張機能

Azure Pipelines の拡張機能を使用して、アプリケーションのデプロイのワークフローに Contrast を連携します。本項では、Contrast と連携するための拡張機能を設定する方法と手順を説明します。

拡張機能を設定する前に、Microsoft の拡張機能をインストールする権限があることを確認してください。アクセス権限をお持ちでない場合は、プロジェクトの拡張機能を要求できます。

Azure Pipelines の拡張機能のインストールと設定

Azure Pipelines の拡張機能をインストールして設定するには：

1. Microsoft の手順に従って、 **Contrast Integration** という拡張機能を検索して、インストールします。
   Microsoft の手順に従って、**Contrast Integration** という拡張機能を検索して、インストールします。

2. サイドバーの下にある **Project Settings**(プロジェクトの設定)を選択します。この設定を変更するには、プロジェクト管理グループに属しているか、設定を変更するのに十分な権限を持っている必要があります。
3. プロジェクトの設定ページの **Pipelines**(パイプライン)セクションで、**Service connections**(サービス接続)を選択します。
4. **New Service connection**(新しいサービス接続)を選択し、**Contrast Server Connection** を選択します。
5. 全てのフィールドに、個人のキー (522ページ)から必要なデータを入力します。

注記
Contrast URL には、末尾に*/Contrast* を含めず、ホストのみを指定してください。

Azure Pipelines の拡張機能でタスクを設定

Azure Pipelines の拡張機能で、リリースパイプラインまたはビルドパイプラインにタスクを設定するには：

1. タスクを追加するパイプラインを選択して、**Edit**(編集)を選択します。
2. リリースパイプラインの場合は、タスクを追加するステージを選択します。
3. タスクを追加するために、省略符号(...)メニューをクリックして、**Add an agentless job**(エージェントレスジョブの追加)を選択します。
4. エージェントレスジョブの横にある[+]ボタンをクリックして、**Contrast Assess - Application Vulnerability Detection** タスクを追加します。
5. 接続先とアプリケーションを選択するために、**Contrast Service Connection** メニューで、接続したい**サービス接続**を選択します。**Manage**(管理)を選択して、**Project Settings**(プロジェクトの設定)ページの **Service connections**(サービス接続)の設定画面にアクセスすることもできます。
6. **Application** メニューで、アプリケーションを 1 つ選択します。
7. タスクの設定として、**Allowed Status**(許可するステータス)フィールドと **Build Number**(ビルド番号)フィールドを使用して、Contrast からの結果にフィルターをかけます。結果にフィルターをかけたくない場合は、空欄のままにします。これらのフィールドで設定した値が、以降のフィールドで設定する条件に対して検証されます。
8. 深刻度の設定に進み、深刻度ごとに許容する脆弱性の上限数を設定する必要があります。選択したアプリケーションに、その深刻度で許容する数を超える脆弱性がある場合、タスクは失敗になります。

ビルドパイプラインの場合のみ：タスクが失敗した場合にジョブが実行されないようにするには、Contrast のタスクを含むエージェントレスジョブへの依存をジョブに設定する必要があります。

1. タスク失敗時に実行したくないジョブを選択します。
2. **Dependencies**(依存関係)セクションで、**Agentless job**(エージェントレスジョブ)を追加します。

注記
このタスクは、エージェントレスジョブにのみ使用できます。

Azure Pipelines のパイプラインにリリースゲートを追加

リリースゲートは、特定のアプリケーションの脆弱性が一定のしきい値を超えた場合に、デプロイを止める保護機能です。Azure Pipelines の拡張機能を使用して、リリースゲートを追加するには：

1. ゲートを追加するリリースパイプラインを検索して、**Edit**(編集)を選択します。
2. ゲートを設定するステージとデプロイの条件を選択します。ゲートは、デプロイ前条件またはデプロイ後条件のいずれかになります。同じ条件に複数のゲートを追加することができます。
3. ステージのデプロイ条件アイコンを選択し、トグルボタンを選択して **Gates**(ゲート)を有効にします。
4. **Add**(追加)をクリックし、**Contrast Assess - Application Vulnerability Detection** を選択します。
5. Contrast Service Connection で、サービス接続を選択します。サービス接続を新規に作成するには、サービス接続のドロップダウンメニューの横にある **New**(新規)を選択して、全てのフィールドに入力し、**OK** を選択します。
   Refresh list(リストをリフレッシュ)を選択して、新規に作成した接続を選択します。
6. Application フィールドをクリックするか、**Refresh Application**(アプリケーションをリフレッシュ)を選択すると、アプリケーションの一覧が表示されます。リリースパイプラインで検証するアプリケーションを選択します。
7. 必要に応じて、ゲートでの検証データを取得する際に、フィルターをかける脆弱性のステータスやビルド番号を選択できます。
8. 深刻度ごとに許容される脆弱性の上限数を設定します。パイプラインがこのゲート達して検証が失敗した場合、パイプラインが有効になるか、検証がタイムアウトするまでサンプリングが要求され続けます。
   ステージのゲートの定義方法やゲートの設定方法については、Microsoft ドキュメントに詳しい説明があります。
   ステージのゲートの定義方法やゲートの設定方法については、Microsoft ドキュメントに詳しい説明があります。

ヒント
Evaluation(検証)オプションをカスタマイズして、ゲートの再検証までの時間を設定できます。例えば、この値を 24 時間に設定することで、ゲートが毎日検証されるようになります。これにより、パイプラインの実行を最初からやり直すことなく(または、手動での承認を構成している場合は承認が要求されずに)、脆弱性を修正して、必要なゲート条件を通過できます。

Azure Service Fabric で.NET Framework または.NET Core エージェントを使用する

コンテナイメージを使用する場合は、コンテナにインストール (174ページ)する手順に従ってください。そうでない場合は、Azure Service Fabric サービスに Contrast .NET Framework または.NET Core エージェントを追加します。

ヒント
スタンドアロンの実行サービスの場合、`ServiceManifest.xml` ファイルは最上位の Azure Service Fabric プロジェクト(例、`sfproj` ファイル)にあります。

1. 適切な NuGet パッケージをサービスのメインプロジェクトにインストールします。
   - **.NET Framework の場合：** `Contrast.NET.Azure.AppService` をインストールします。`contrastsecurity` フォルダにあるすべてのファイルは、プロパティがに設定されている必要があります。

- **.NET Core の場合：** `Contrast.SensorsNetCore` をインストールします。`contrast` フォルダにあるすべてのファイルは、プロパティがに設定されている必要があります。

2. `ServiceManifest.xml` の `ServiceManifest/CodePackage/EntryPoint/ExeHost/WorkingDirectory` を `CodePackage` に設定します。

```
<CodePackage Name="Code" Version="1.0.0">
          <EntryPoint>
          <ExeHost>
            <Program>DemoNetFxStatelessService.exe</Program>
            <WorkingFolder>CodePackage</WorkingFolder>
```

3. `ServiceManifest.xml` で環境変数を定義して、プロファイラを指定します。
   - **.NET Framework の場合：**

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="COR_ENABLE_PROFILING" \
Value="1"/>
            <EnvironmentVariable Name="COR_PROFILER" \
Value="{EFEB8EE0-6D39-4347-A5FE-4D0C88BC5BC1}"/>
            <EnvironmentVariable Name="COR_PROFILER_PATH_32" \
Value=".\contrastsecurity\runtimes\win-
x86\native\ContrastProfiler.dll" />
            <EnvironmentVariable Name="COR_PROFILER_PATH_64" \
Value=".\contrastsecurity\runtimes\win-
x64\native\ContrastProfiler.dll" />
            <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

   - **.NET Core の場合：**

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="CORECLR_ENABLE_PROFILING" \
Value="1"/>
            <EnvironmentVariable Name="CORECLR_PROFILER" \
Value="{8B2CE134-0948-48CA-A4B2-80DDAD9F5791}"/>
            <EnvironmentVariable Name="CORECLR_PROFILER_PATH_32" \
Value="contrast\runtimes\win-x86\native\ContrastProfiler.dll"/>
            <EnvironmentVariable Name="CORECLR_PROFILER_PATH_64" \
Value="contrast\runtimes\win-x64\native\ContrastProfiler.dll"/>
            <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

4. 次のいずれかで、エージェントを設定します。
   - **YAML 設定ファイル：**このファイルをサービスのメインプロジェクトに追加します。ファイルのプロパティが、に設定されていることを確認してください。 以下のように、ファイルの場所を指定する環境変数を `ServiceManifest.xml` に追加します。

```
<CodePackage>
              <EnvironmentVariables>
                <EnvironmentVariable Name="CONTRAST_CONFIG_PATH" \
Value="contrast_security.yaml"/>
```

   - **環境変数：** 以下のように、環境変数を `ServiceManifest.xml` に追加します。

```
<CodePackage>
          <EnvironmentVariables>
            <EnvironmentVariable Name="CONTRAST__API__URL" \
Value="https://teamserver-staging.contsec.com"/>
```

```
            <EnvironmentVariable Name="CONTRAST__API__API_KEY" \
Value="aBcD0123"/>
            <EnvironmentVariable Name="CONTRAST__API__SERVICE_KEY" \
Value="ABCD0123"/>
            <EnvironmentVariable Name="CONTRAST__API__USER_NAME" \
Value="agent_123@Team"/>
```

5. 通常どおり、Azure Service Fabric アプリケーションをデプロイします。

Contrast Bamboo プラグイン

このプラグインは、Bamboo に機能を追加して、Contrast に接続するためのプロファイルを設定し、脆弱性のしきい値に対してビルドを検証できるようにします。

インストールと設定

Bamboo プラグインをインストールして設定するには：

1. Contrast Bamboo プラグイン(*contrast-bamboo-plugin-#.#.#.jar*)を Bamboo Marketplace(マーケットプレイス)からダウンロードします。
2. Bamboo インスタンスにログインし、左上の設定メニューから、**Add-Ons** を選択します。
3. **Upload add-on** を選択します。
4. ファイルのアップロードを促すメッセージが表示されたら、*contrast-bamboo-plugin-#.#.#-SNAPSHOT.jar* を選択します。
5. **User-installed add-ons** にプラグインが表示されていることを確認します。
6. プラグインがインストールされたら、Contrast のプロファイルを設定します。サイドナビゲーションバーにある **Add-Ons** で、**Contrast Profiles** を選択します。
7. **Profile Configuration** ページで **New Profile** を選択して、フォームに入力します。

> **注記**
> SaaS 版をご利用のお客様の場合は、Contrast URL を入力する必要はありません。

8. **Test Connection** を選択して、設定が正しいことを確認します。接続できると、成功(Success!)のメッセージが表示されます。

脆弱性のしきい値の設定

Bamboo プラグインは、ビルドジョブに対して、設定した脆弱性の条件をチェックするためのタスクとして追加できます。これにより、アプリケーションに存在する脆弱性の数や種類を Contrast で確認できます。

ビルドジョブにタスクを追加するには：

1. **Create a New Build Plan** を選択します(既存のプランを使用することもできます)。
2. プロジェクト(Project)、プラン名(Plan name)、リポジトリホスト(Repository host)を入力または選択します。プロジェクトキーとプランキーは自動生成されます。
3. プランを作成したら、ビルドプロセスにタスクを追加するために **Add Task**(タスクの追加)を選択します。
4. 表示される画面で、**Contrast CI for Assess** というタスクを検索して選択します。
   Contrast CI for Assess configuration 画面では、Contrast Profile(プロファイル)だけでなく、Server Name(サーバ名)、Application Name(アプリケーション名)、**Passive**(パッシブ)パラメータを指定します。サーバ名は必須ではありませんが、指定する場合は Contrast 内のサーバと一致している必要があります。アプリケーション名は指定されたサーバ上にある必要があります。
   Passive パラメータを選択した場合、プラグインは(ビルド固有の脆弱性だけではなく)アプリケーションに対して全ての脆弱性のクエリを実行します。これを行うと、Bamboo ビルドでビルド後の Contrast の処理を行う前に、アプリケーションの統合テストを実行する必要がなくなります。
5. 次に、ビルドを失敗させるための条件を定義します。

- **Threshold Count**(しきい値の数)： ビルドを失敗させるのに必要な検出数の最小値。
- **Threshold Severity**(しきい値の深刻度)： 検出結果をしきい値の数として数えるための最小レベルの深刻度。
- **Threshold Vulnerability Type**(しきい値の脆弱性の種類)： 検出結果をしきい値の数として数える対象とする脆弱性の種類。

> **注記**
> **Any** オプションを使用すると、全ての深刻度や全ての脆弱性の種類がしきい値の数にカウントされます。

6. タスクごとに複数の条件を設定するには、**Add New Threshold Condition**(新しいしきい値条件の追加)を選択します。
7. **Save**(保存)を選択します。
8. 左下のチェックボックスを選択して、ビルドプランを有効にします。

Bugzilla とのインテグレーション

Bugzilla とインテグレーションするには：

1. **ユーザメニュー**で、**組織の設定 > インテグレーション**にアクセスします。
2. Bugzilla の行で、**接続**を選択します。
3. 表示される画面で、Bugzilla のプロパティフィールドに入力します。

フィールド	説明
名前	バグ管理システムのエントリ名、Contrast の検出結果を送信するときに表示されるラベル
ユーザ名	Bugzilla に接続するアカウントのユーザ名
パスワード	指定したユーザ名のパスワード
ホスト	Bugzilla インスタンスの URL
アプリケーション	Bugzilla のプロダクト(product)/コンポーネント(component)にマップするアプリケーション
プロダクト	アプリケーションにマップする Bugzilla プロダクト(product)
コンポーネント	アプリケーションにマップする Bugzilla コンポーネント(component)
バージョン	アプリケーションにマップする Bugzilla バージョン番号(version)
優先度	Bugzilla に検出結果をエクスポートする際に使用する優先度

4. **接続をテスト**を選択して、接続を確認します。接続テストによって、Contrast が Bugzilla インスタンスとの通信と認証を行い、指定した**プロダクト**(product)、**コンポーネント**(component)、**バージョン**(version)の存在を確認します。

Eclipse

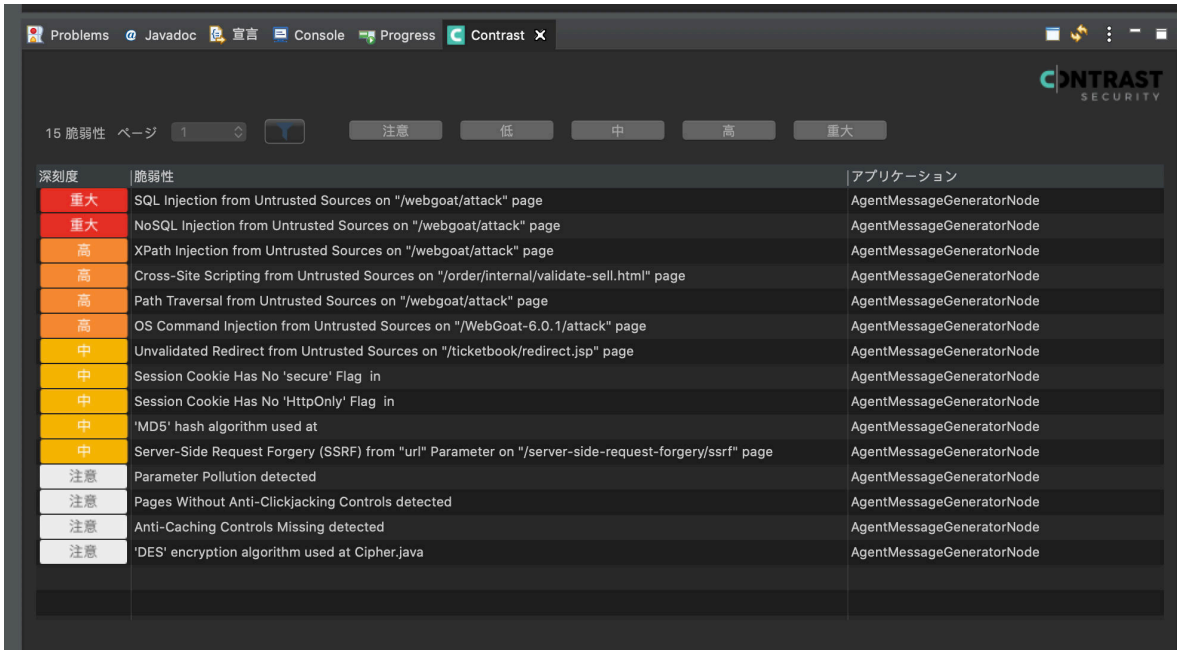
Contrast エージェントが組み込まれているアプリケーションでは、Contrast からの脆弱性情報を開発中の Eclipse IDE で直接表示することができ、迅速な修正が可能になります。影響を受けるコード行を確認し、脆弱性について Contrast Web インターフェイスでさらに詳細を参照できます。

このプラグインは Mac/OS、Linux および Windows で利用でき、最新バージョンの Eclipse をサポートします。

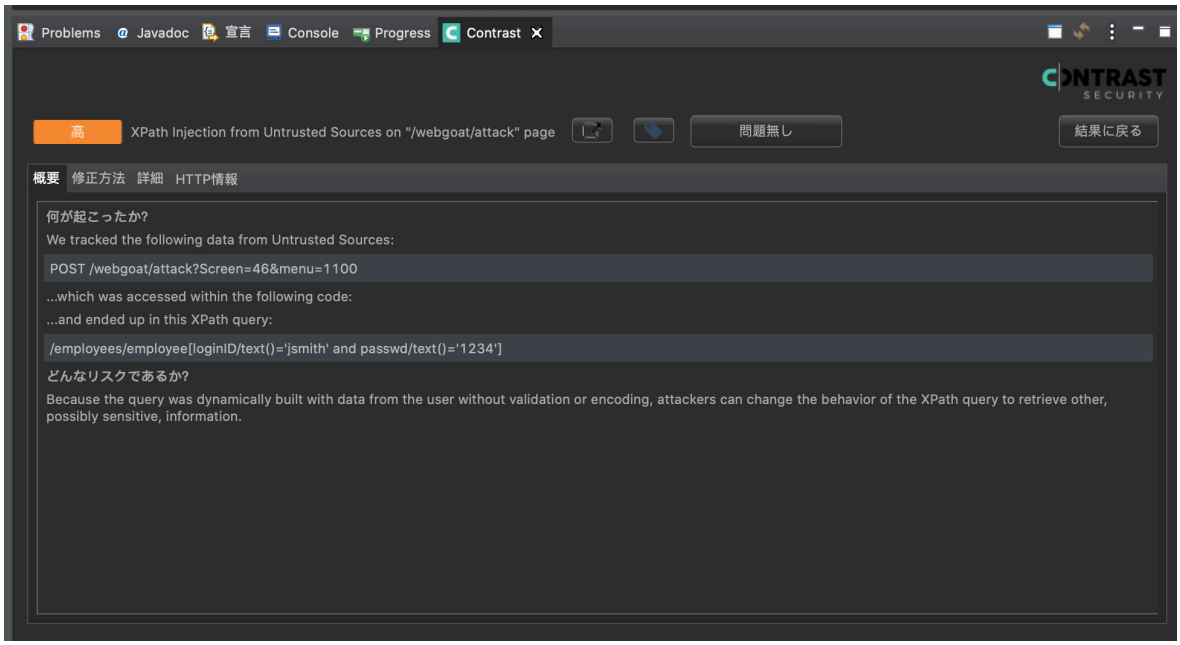
Eclipse プラグインをインストールするには、Eclipse マーケットプレイスにアクセスするか、以下の操作を行います。

1. Eclipse で、**ヘルプ > Eclipse マーケットプレイス**を選択します。
2. 「Contrast Security」を検索します。
3. **インストール**を選択します。
4. **ウィンドウ > ビューの表示 > その他**を選択し、プラグインを設定します。
5. 「Contrast」を検索し、検索で表示されたビューを追加します。
6. 設定ページで、**ユーザ名**、**API キー**、**組織 ID**、および**サービスキー**を入力します。これらのキー(522ページ)は Contrast Web インターフェイスのユーザの設定画面で確認できます。

7. **追加**を選択します。
8. **脆弱性**ビューに、Contrast からの全ての脆弱性の一覧が表示されます。これらの脆弱性はソートしたり、フィルターで絞り込んだりできます。



9. 特定の脆弱性に関する情報を参照するには、その脆弱性のタイトルを選択します。そこから、**修正方法**のタブを選択すると、その脆弱性の対処方法が表示されます。**詳細**を選択し、Java スタックトレースをダブルクリックすると、特定のソースコード行にフォーカスできます。ここでは、脆弱性のステータスを変更することもできます。



10. **ページに移動**アイコンを選択すると、Contrast Web インターフェイスが開き、より詳細な脆弱性の情報を確認できます。

Generic Webhook

Contrast では汎用の Webhook によるインテグレーションをサポートしており、POST メッセージを受信するあらゆる URL で通知 (831ページ)を受信することができます。Contrast のイベント (754ページ)でトリガーされる、`$ApplicationName` や`$ServerId` などのカスタム変数をペイロードに追加 (752ページ)できます。

接続

Generic Webhook に接続するために、以下を行います。

1. Contrast からの通知を受信する URL を取得します。
2. ユーザメニューで**組織の設定 > インテグレーション**を選択します。
3. **Generic Webhook** の行で、**接続**を選択します。
4. Webhook に名前を付け、URL を指定のフィールドに貼り付けます。
5. インテグレーションするアプリケーションを選択します(複数選択可)。
6. **ペイロード**フィールドに 変数 を入力します。例：

```
{
   "title": "$Title",
   "message": "$Message"
}
```

7. **追加**を選択します。

Webhook をテストするには：

1. **組織の設定** > **通知**を選択します。
2. **インテグレーション**のドロップダウンで、webhook のインテグレーション名を選択します。
3. 通知を受けたい**イベントタイプ**(通知の登録)ごとに、**インテグレーション**列のトグルボタンをクリックします。
4. イベントを発生させて、指定した URL で通知を受信していることを確認してください。

注記

この Webhook は、5 回試行しても応答しない場合、切断されます。設定を復元するには、接続を再テストし、再保存する必要があります。

Contrast で Generic Webhook が切断された時に、すべての管理者に Contrast 内と E メールで通知するよう、インテグレーションを設定できます。

この設定を行うには、**組織の設定 > インテグレーション > Generic Webhook > 設定を表示**にアクセスします。 設定を行う接続名を選択します。そして、**接続失敗時に通知**のチェックボックスを選択し、**保存**をクリックします。

Generic Webhook の変数

NEW_VULNERABILITY や SERVER_OFFLINE などの Contrast イベントからのデータを使用して、Generic Webhook (750ページ)のレスポンスをカスタマイズできます。各イベントには、ペイロードのリクエストで呼び出すことができる変数があります。変数には、一般的な共通情報として利用するもの、もしくは、アプリケーション用、サーバ用、脆弱性用があります。

変数	説明
共通変数	

変数	説明
`$EventType`	Webhook のトリガーとなるイベントタイプ 例：SERVER_OFFLINE
`$Message`	Webhook のトリガーとなったイベントの概要を表すメッセージ
`$OrganizationId`	Contrast で組織の作成時に、組織に割り当てられた一意の ID
`$OrganizationName`	組織の名前
`$Title`	常に“Contrast Security”を返す
アプリケーション変数	
`$ApplicationChild`	アプリケーションが子アプリケーションの場合は真(true)を返し、そうでない場合は偽(false)を返す
`$ApplicationCode`	アプリケーションのタイトルに表示されるアプリケーション名の省略形、デフォルトでは空白 例：TEST
`$ApplicationContextPath`	アプリケーションのコンテキストパス 例： `/example/somethingelse`
`$ApplicationFirstSeen`	アプリケーションが最初に検知された日時(Unix 時間) 例：1572033840000
`$ApplicationHasParentApp`	アプリケーションに親がある場合は真(true)を返し、そうでない場合は偽(false)を返す
`$ApplicationImportance`	アプリケーションの重要度の列挙値(enum 値) 例：MEDIUM
`$ApplicationId`	Contrast でアプリケーションの登録時に、アプリケーションに割り当てられた一意の ID 例： 49fe2978-1833-4441-83db-2b7o486d9413
`$ApplicationImportanceDescription`	アプリケーションに割り当てられた重要度、例：Medium
`$ApplicationLanguage`	アプリケーションのプログラミング言語
`$ApplicationLastSeen`	アプリケーションが最後に検知された日時(Unix 時間)、例：1572033840000
`$ApplicationLicenseLevel`	アプリケーションに Assess ライセンスがあるかどうか、値：Licensed、Unlicensed
`$ApplicationMaster`	アプリケーションがマスターアプリケーションである場合は真(true)を返し、そうでない場合は偽(false)を返す
`$ApplicationName`	アプリケーションの名前
`$ApplicationParentAppId`	Contrast でアプリケーション(この場合は親アプリケーション)の登録時に、アプリケーションに割り当てられた一意の ID(存在する場合) 例： 49fe2978-1833-4441-83db-2b7o486d9413
`$ApplicationTags`	アプリケーションのタグのカンマ区切りリスト
`$ApplicationTotalModules`	アプリケーションにあるモジュールの数
サーバ変数	
`$Environment`	サーバの環境、値：DEVELOPMENT、QA、PRODUCTION
`$ServerId`	イベントに関与しているサーバの ID 複数サーバが関与している場合、カンマ区切りのサーバ ID リスト
`$ServerName`	イベントに関与しているサーバの名前 複数サーバが関与している場合、カンマ区切りのサーバ名リスト
脆弱性変数	
`$Severity`	イベントのトリガーが脆弱性の場合、脆弱性の深刻度を返す
`$Status`	イベントのトリガーが脆弱性の場合、脆弱性のステータスを返す
`$TraceId`	イベントのトリガーが脆弱性の場合、脆弱性 ID を返す
`$VulnerabilityAgentLanguage`	脆弱性が検知されたアプリケーションの言語またはフレームワーク名（例：.Java、.NET、Ruby など）

変数	説明
$VulnerabilityAppVersionTags	脆弱性が検出されたアプリケーションのバージョン 例：v1.2.3
$VulnerabilityAutoRemediatedExpirationPeriod	脆弱性が自動修復される有効期間(Unix 時間) 例：1572033840000
$VulnerabilityBugTrackerTickets	脆弱性情報がバグ管理システムに送信されて、作成されたチケットのリスト(カンマ区切り) 例：ticket1, ticket2, ticket3
$VulnerabilityCategory	検出された脆弱性のカテゴリー、例：Injection
$VulnerabilityClosedTime	脆弱性がクローズされた日時(Unix 時間) 例：1572033840000
$VulnerabilityConfidence	脆弱性の信頼度
$VulnerabilityDefaultSeverity	脆弱性のデフォルトの深刻度
$VulnerabilityDiscovered	脆弱性が最初に検出された日時(Unix 時間) 例：1572033840000
$VulnerabilityEvidence	脆弱性のエビデンス
$VulnerabilityInstanceUuid	Contrast で脆弱性インスタンスの作成時に、脆弱性インスタンスに割り当てられた一意の ID 例： R33T-N00B-TGIF-RM6P
$VulnerabilityFirstTimeSeen	脆弱性が最初に検出された日時(Unix 時間)、例： 1572033840000
$VulnerabilityImpact	脆弱性の影響度レベル、値：Low、Medium、High
$VulnerabilityLastTimeSeen	脆弱性が最後に確認された日時(Unix 時間)、例： 1572033840000
$VulnerabilityInstanceLastTimeSeen	脆弱性が最後に確認された日時(Unix 時間)、例：1572033840000
$VulnerabilityLicenseLevel	脆弱性のライセンスレベル
$VulnerabilityLikelihood	脆弱性の可能性レベル 値：Low、Medium、High
$VulnerabilityReportedToBugTracker	脆弱性がバグ管理システムに送信された日時(Unix 時間) 例：1572033840000
$VulnerabilityReportedToBugTrackerTime	脆弱性がバグ管理システムに送信された場合、真(true)を返す
$VulnerabilityRule	脆弱性に関連付けられたルール
$VulnerabilityRuleName	脆弱性に関連付けられたルールの名前
$VulnerabilityRuleTitle	脆弱性に関連付けられたルールのタイトル
$VulnerabilitySubStatus	脆弱性のサブステータス
$VulnerabilityTags	脆弱性に関連付けられたカスタムタグ 例：my-custom-tag
$VulnerabilityTitle	脆弱性のタイトル
$VulnerabilitySubStatusKeyCode	脆弱性サブステータスのキーコード
$VulnerabilityTotalTracesReceived	脆弱性が受信された合計回数
$VulnerabilityUuid	脆弱性を検索するために使用される一意の ID
$VulnerabilityVisible	脆弱性にライセンスがあり参照可能である場合は真(true)を返し、そうでない場合は偽(false)を返す
$VulnerabilityRule	イベントのトリガーが脆弱性の場合、脆弱性が違反したルールを返す
$VulnerabilityTags	イベントのトリガーが脆弱性の場合、脆弱性に関連付けられたタグのリスト(カンマ区切り)を返す

イベントと Generic Webhook 変数

NEW_VULNERABILITY や SERVER_OFFLINE などの Contrast のイベントからのデータを使って、Generic Webhook (750ページ)のレスポンスをカスタマイズできます。各イベントには、共通 (752ページ)変数、アプリケーション (753ページ)変数、サーバ (753ページ)変数、または脆弱性 (753ページ)変数があり、ペイロードのリクエストで呼び出すことができます。

イベント	変数
ATTACK_END	共通 (752ページ)、アプリケーション (752ページ)、サーバ (753ページ)
ATTACK_EVENT_COMMENT	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
ATTACK_UPDATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
EXPIRING_LICENSE	共通 (752ページ)、アプリケーション (753ページ)
NEW_ASSET (新規アプリケーションの場合)	共通 (752ページ)、アプリケーション (753ページ)およびサーバ (753ページ)(新規アプリケーションの場合)
NEW_ATTACK_APPLICATION	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_ATTACK_UPDATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_ATTACK	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_VULNERABILITY_COMMENT	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
NEW_VULNERABILITY	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
NEW_VULNERABLE_LIBRARY	共通 (752ページ)、アプリケーション (753ページ)
SERVER_OFFLINE	共通 (752ページ)、サーバ (753ページ)
VULNERABILITY_CHANGESTATUS_CLOSED	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
VULNERABILITY_CHANGESTATUS_OPEN	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
VULNERABILITY_DUPLICATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)

GitHub とのインテグレーション

インテグレーションを設定して、Contrast でアプリケーションの脆弱性が検出された際に、自動で GitHub に問題が送信されるようにします。

設定を開始する前に、以下を準備してください。

- GitHub アカウントの認証情報(ユーザ名と個人用アクセストークン)。個人用アクセストークンを作成する際に、必ず **repo** の権限を有効にしてください。
- アプリケーションに対する GitHub 組織とリポジトリへのアクセス
- リポジトリへの書込み権限(push アクセス)。設定フォームで、ラベル、マイルストーン、担当者を設定するために必要です。
- HTTP 経由で Contrast にアクセス可能な GitHub の実行インスタンス

GitHub に接続するために、以下を行います。

1. **ユーザメニュー**で、**組織の設定 > インテグレーション**にアクセスします。
2. GitHub の行にある**接続**をクリックします。
3. **GitHub** への接続の設定画面で、インテグレーション名、GitHub に接続するアカウントのユーザ名、個人用アクセストークンを各フィールドに入力します。GitHub の URL は、設定する Contrast インスタンスからアクセスできる必要があります。
4. 設定画面の下部にあるチェックボックスをオンにすると、新たに検出された脆弱性に対して GitHub でチケットが自動的に作成されます。表示される複数選択フィールドで、チケットを作成する脆弱性の**ルール**と**深刻度**を選択します。デフォルトでは、**重大**と**高**が選択されます。
5. フィールドに入力したら、**接続をテスト**を選択します。この処理は、GitHub の組織とリポジトリの数によっては、数分かかる場合があります。接続テストでは、Contrast から GitHub インスタンスにアクセスでき、指定したユーザがログインできることを確認します。
6. 接続できたら、このバグ管理システムとの連携を有効にする**アプリケーション**を選択します。
7. ドロップダウンで **GitHub の組織**と**リポジトリ**フィールドの値を選択します。

> **注記**
> **GitHub の組織**または**リポジトリ**の値を変更した場合は、オプションフィールドの値を再入力する必要があります。

8. 必要に応じて、GitHub の課題の**ラベル**(Labels)、**担当者**(Assignees)、**マイルストーン**(Milestone)を各フィールドを使用して指定できます。

> **注記**
> 複数の脆弱性を 1 つの課題として GitHub に一括送信した場合、GitHub でのチケットのステータスは、そのチケットに関連付けられている全ての脆弱性に反映されます。複数の課題が 1 つの脆弱性に関連付けられている場合は、全てのチケットをクローズするまでその脆弱性をクローズできません。

Gradle プラグイン

Contrast Gradle プラグインは、*Contrast.jar* とビルドを統合するのに使用されます。Contrast で認証を行い、最新の Java エージェントをダウンロードしてビルドを検証することができます。

> **注記**
> Gradle は、`build.gradle` ファイルを活用してアプリケーションを設定するビルドツールです。さまざまな種類のアプリケーションをビルド、パッケージ化、テストするのに使用されます。

サンプル Web アプリケーションの複製

プロジェクトを設定する最も簡単な方法は、Gradle ベースの Web アプリケーションのサンプルを複製することです。このアプリケーションは Maven から Gradle へと移行され、MongoDB に依存します。

1. インストールしてデータベースパスを設定します。

```
git clone https://github.com/Contrast-Security-OSS/Contrast-Sample-
Gradle-Application
brew install mongodb
sudo mkdir -p /data/db
brew services start mongodb
```

2. アプリケーションの実行準備が整います。*Contrast-Sample-Gradle-Application/build.gradle* ファイルを開きます。スクロールして `contrastConfiguration` 拡張機能を見つけます。`appName` と `serverName` 以外の値は、全て個人のキー (522ページ)にあります。

```
contrastConfiguration {
    username = "username"
    apiKey = "apiKey"
    serviceKey = "serviceKey"
    apiUrl = "apiUrl"
    orgUuid = "orgUuid"
    appName = "editLATER"
```

```
    serverName = "editLATER"
}
```

3. `contrastInstall` タスクを呼び出して、Contrast JAR ファイルをインストールします。これにより、プロジェクトのビルドディレクトリに Contrast JAR がインストールされます。

```
cd path/to/Contrast-Sample-Gradle-Application
gradle build -x test contrastInstall
```

4. Java エージェントでアプリケーションを実行します。サーバが起動します。

```
cd path/to/Contrast-Sample-Gradle-Application/build
java -Dcontrast.agent.java.standalone_app_name=mytestapp -
Dcontrast.server=mytestserver -jar libs/Contrast-Sample-Gradle-
Application-0.0.1-SNAPSHOT.jar
```

5. localhost:8080 でアプリケーションが実行されていることと、Contrast にアプリケーションが表示されることを確認します。
6. Contrast 内で、前述のコマンドで指定された `appname` のアプリケーションが表示されていることを検証します。
7. Contrast-Sample-Gradle-Application プロジェクトの *build.gradle* で `contrastConfiguration` を編集して、前のステップで Java エージェントのオプションとして指定した `appName` と `serverName` を指定します。

```
contrastConfiguration {
    username = "alreadySetup"
    apiKey = "alreadySetup"
    serviceKey = "alreadySetup"
    apiUrl = "alreadySetup"
    orgUuid = "alreadySetup"
    appName = "mytestapp"
    serverName = "mytestserver"
}
```

8. いつでも検証タスクを実行して脆弱性を確認できます。

```
gradle build contrastVerify -x test
```

プラグインの使用

プラグインのコードは GitHub リポジトリで参照できます。プラグインによって追加される 2 つのタスク(`contrastInstall` および `contrastVerify`)とその仕組みを確認できます。

プラグインの最新バージョンは、Gradle プラグインの Web ページにあります。

タスク	説明
`contrastInstall`	ローカルプロジェクトに Contrast Java エージェントをインストールします。Contrast エージェントで JVM が起動するよう、プラグインによって *gradle.properties* ファイルの `org.gradle.jvmargs` プロパティが編集されます。`contrastVerify` タスクで脆弱性を絞り込むためのアプリケーションバージョンは、このタスク中に生成されます。プラグインでは、以下の順番でアプリケーションバージョンが生成されます。 • TravisCI でビルドが実行されている場合、Contrast は `appName-$TRAVIS_BUILD_NUMBER` を使用します。 • CircleCI でビルドが実行されている場合、Contrast は `appName-$CIRCLE_BUILD_NUM` を使用します。 • TravisCI と CircleCI 以外でビルドが実行されている場合、Contrast は `appName-yyyyMMddHHmm` 形式でアプリケーションバージョンを生成します(`yyyyMMddHHmm` は、ビルドの生成時刻です)。
`contrastVerify`	Web アプリケーションで新規の脆弱性をチェックします。

Contrast IntelliJ プラグイン

IntelliJ プラグインを使用すると、Contrast エージェントが組み込まれたアプリケーションの脆弱性情報を IntelliJ IDE から確認できます。

このプラグインによって、脆弱性の影響を受けるコード行が IntelliJ 内で表示され、詳細を Contrast Web インターフェイスで確認することができます。このようにして、開発者は開発中にアプリケーションセキュリティに関するフィードバックを得ることができ、脆弱性に迅速に対応することができます。

このプラグインは、IntelliJ バージョン 2017.1.5 以降をサポートします。

IntelliJ プラグインをインストールして設定し、使用するには：

1. Windows の場合は、**File** > **Settings** > **Plugins** > **Browse Repositories** にアクセスします。
   OSX の場合は、**Preferences** > **Plugins** > **Search in Repositories** にアクセスします。
2. **Contrast Security** を検索します。
3. **Install** を選択します。
4. Windows の場合は、**File** > **Settings** > **Contrast** にアクセスします。
   OSX の場合は、**Preferences** > **Other settings** > **Contrast** にアクセスします。
5. **Contrast URL**、**Username**(ユーザ名)、**Service Key**(サービスキー)、**API Key**(API キー)、および **Organization ID**(組織 ID)を入力します。これらの情報はプロファイル (522ページ)で確認できます。
6. **Add** を選択して、新しい組織を追加します。
7. Contrast の画面で **Refresh**(リフレッシュ)を選択すると、脆弱性の一覧が更新されます。IntelliJ の **脆弱性**ビューに、Contrast からの全ての脆弱性が一覧で表示されます。
   脆弱性をソートするには、列のヘッダを選択します。フィルターを使用するには、ファネルアイコンを選択します。詳細を表示するには、脆弱性の名前を選択します。

IntelliJ で Java エージェントを設定

IntelliJ IDE のサポート対象のアプリケーションサーバを使用するアプリケーションに Contrast エージェントを追加するには：

1. アプリケーションツールバーで **Run** をクリックして、ドロップダウンメニューから **Edit Configuration...**をクリックします。
2. IntelliJ サーバの設定インスタンスを選択します。
3. **サーバの設定**タブを選択して、**VM Options** に Contrast ランチャー文字列となる -`javaagent:<YourContrastJarPath>`を入力します。`<YourContrastJarPath>`をお使いの Contrast JAR (75ページ)ファイルへのパスに置き換えます。
4. **Apply** をクリックして、**OK** をクリックします。
5. **サーバ**を起動します。
   サーバのコンソールに、Contrast の起動メッセージが表示されます(サーバの起動には、1〜2 分ほど時間がかかります)。
6. アプリケーションにアクセスし、起動するまでにさらに 1 分ほどお待ちください。

Jenkins とのインテグレーション

Jenkins は、アプリケーションのビルド、テスト、デプロイ、実行のプロセスを自動化する継続的インテグレーション(CI)ツールです。

Contrast の Jenkins プラグインを使用すると、このパイプラインにアプリケーションセキュリティゲートを追加できます。セキュリティゲートとして、脆弱なアプリケーションの Jenkins ジョブを「FAILURE(失敗)」や「UNSTABLE(不安定)」などのビルド結果で失敗させる基準を指定できます。

ヒント
このプラグインのソースコードは、Jenkins プラグインの Github リポジトリで参照できます。

互換性を保つために、次のバージョンを使用してください。

Jenkins	Contrast-Jenkins プラグイン	Contrast
2.60.3	3.4	3.7.6
2.60.3	3.7	3.7.10
2.60.3	3.8	3.8.0

Jenkins プラグインのインストールと使用

1. Contrast と Jenkins 間の接続を定義 (759ページ)します。
2. お使いの環境に合わせて、Jenkins をどのように利用するかを決めます。
   - フリースタイルのジョブを使用している場合、システムレベル (760ページ)かビルド後の処理 (761ページ)として、脆弱性に対するセキュリティ制御を定義できます。
   - パイプラインのステップ (761ページ)に脆弱性のセキュリティ制御を定義します。
   - Contrast の組織管理者がジョブ結果ポリシー (763ページ)を定義することもできます。
3. ビルドを実行 (767ページ)します。

関連項目

- Jenkins から Contrast へ接続 (759ページ)
- Jenkins でセキュリティ制御を定義 (763ページ)
- Contrast でジョブ結果ポリシーを定義 (763ページ)
- Jenkins でのビルドの実行 (767ページ)

接続

開始する前に

- Contrast Security の全てのインテグレーションで、 接続を設定するために組織管理者のロール (955ページ)が必要です。
- Jenkins をインストールしてください。
- Jenkins パイプラインが既にセットアップ済みであること。

接続を定義

Jenkins で Contrast への接続を定義するには：

1. Jenkins のダッシュボードで、左のサイドバーの **Jenkins の管理**を選択します。
2. **System Configuration** で**プラグインの管理**を選択します。
3. **インストール済み**タブで、**Contrast Continuous Application Security** プラグインを有効にするためにチェックします。
4. 再度、**Jenkins の管理**を選択します。
5. **システムの設定**を選択し、**Contrast Connections** のセクションを探してください。
6. **Contrast Username** の項目に、Contrast のユーザ名を入力します。このユーザ名は、Contrast のアカウントに使用する E メールアドレスです。
7. 以下を入力します。
   - **Contrast API Key**(Contrast API キー)
   - **Contrast Service Key**(Contrast サービスキー)
   - **Contrast URL**
   - **Organization ID**(組織 ID)

   これらの情報はプロファイル (522ページ)にあり、**ユーザの設定 > プロファイル > あなたのキー**でアクセスできます。
8. **Result of a vulnerable build** を選択して、 アプリケーションが脆弱である場合に Contrast から Jenkins ジョブの結果をどのように返すかを選択します。指定できるオプションは以下の通りです。

- **FAILURE**(失敗)
- **UNSTABLE**(不安定)
- **SUCCESS**(成功)
- **NOT_BUILT**(ビルドしない)
- **ABORTED**(停止)

9. Jenkins インスタンスがアプリケーションを見つけられないときに Jenins ジョブを自動的に失敗させたい場合は、**Apply this vulnerable build result to the job when Jenkins encounters an error with Contrast** のボックスにチェックをします。
10. アプリケーションが Jenkins のシステムレベルで脆弱であるかを判断するために、Contrast プラグインで使用する基準を定義できます。ジョブレベルの制御をシステムレベルの制御より優先したい場合は、**Allow job level application vulnerability security controls to override those controls set here at the system level** の横にあるボックスにチェックをします。 インスタンス内のすべての Jenkins ジョブで基準の一貫性を持たせたい場合は、このボックスをオフのままにします。

> **注記**
> ジョブ結果ポリシー (763ページ) を使用してセキュリティ制御を設定している場合、それらのポリシーはジョブレベルまたはシステムレベルで設定されたポリシーよりも優先されます。

11. プラグインが Contrast に認証され、アプリケーションの脆弱性に関する情報を取得できることを確認するために、**Test Contrast connection** をクリックします。
    - プラグインが認証されると成功のメッセージが表示されます。
    - 認証に失敗した場合は、Contrast の認証情報(URL など)を確認してください。

関連項目

- Jenkins でセキュリティ制御を定義 (763ページ)
- Contrast でジョブ結果ポリシーを定義 (763ページ)
- Jenkins でのビルドの実行 (767ページ)

システムレベルのセキュリティ制御を定義

Jenkins で接続を定義 (759ページ)した後、フリースタイルのジョブを使用している場合は、システムレベルで Contrast の脆弱性によるセキュリティ制御を設定することができます。または、 ジョブレベルでセキュリティ制御を設定 (761ページ)することもできますし、ジョブ結果ポリシー (763ページ)を使用して、それらのセキュリティ制御より優先させることもできます。

1. **Contrast Connections > Contrast Vulnerability Security Controls** で、ドロップダウンから以前作成した **Connection** を選択します。
2. **Number of Allowed Vulnerabilities** を設定します。この数字は除かれるので、「5」に設定すると、6 件以上の脆弱性がある場合に Jenkins が失敗します。このフィールドは必須です。
3. ドロップダウンのオプションから **Vulnerability Severity** を選択します(これらは、Contrast の脆弱性の深刻度オプション (707ページ)と同じです)。プラグインにより、このフィールドに指定された深刻度以上の全ての脆弱性について API コールにフィルターが設定されます。このフィールドは必須ではありませんが、選択すると結果を絞り込むことができます。従って、この数値を「5」に設定し、深刻度に **High**(高)を選択した場合、深刻度の高い脆弱性が 6 件以上あると Jenkins は失敗します。
4. ドロップダウンから **Vulnerability Type**(ルール名)を選択します。プラグインは、選択したルール名で脆弱性の数をチェックし、そのルールで許容される脆弱性の数と比較します。このフィールドは必須ではありませんが、選択すると結果を絞り込むことができます。セキュリティ制御ごとに 1 つの深刻度と 1 つのルール名を選択できます。
5. **Vulnerability Statuses** を選択します。これらの脆弱性のステータスは必須ではありませんが、便利な場合があります。例えば、**Confirmed** と **Suspicious** を選択すると、ステータスがオープン中

の脆弱性のみが返されます。ステータスで脆弱性をフィルタリングしない場合は、選択しないままにします。
脆弱性のセキュリティ制御は複数登録することができますが、プラグインは最初に合致した悪条件でジョブを失敗させます。最初に違反した脆弱性のセキュリティ制御でビルド結果が返されます。

ビルド後の処理としてセキュリティ制御を定義

Jenkins のシステムレベル (760ページ)でセキュリティ制御を設定した後に、Jenkins のパイプラインに含まれないフリースタイルのジョブに対して、ジョブレベルでセキュリティ制御を追加することもできます。これを行うには：

1. Jenkins でジョブを定義する際に、**ビルド後の処理**セクションを探します。
2. ドロップダウンから、以前作成した**接続**を選択します。
3. アプリケーションを選択します。このフィールドは必須です。
   - アプリケーションで既に Contrast エージェントを使用している場合は、**Choose your application** ドロップダウンからアプリケーションを選択します。
   - アプリケーションでまだ Contrast エージェントを使用していない場合は、**Application Name** と **Application Language** フィールドを使用して、アプリケーションを指定します。Contrast で検査するアプリケーションと同じアプリケーション名を Jenkins に指定してください。アプリケーションで Contrast エージェントが有効になったら、Contrast によるビルド後の処理時に、そのアプリケーションと同じ名前と同じ言語が使用されます。
4. システムレベルの脆弱性セキュリティ制御を上書きできる (760ページ)ように接続が設定されている場合は、**Override Vulnerability Security Controls at the Jenkins system level** の横にあるチェックボックスをオンにすることで、その設定を上書きできます。
   この場合、このジョブの **Number of Allowed Vulnerabilities**、**Vulnerability Severity**、**Vulnerability Type**、**Vulnerability Statuses** も指定する必要があります。
5. **Query vulnerabilities by** でオプションを選択して、脆弱性のクエリ方法を選択します。そうすると、そのジョブで検出される脆弱性のみが対象となります。デフォルトでは、最初のオプションである、`appVersionTag`、`format: applicationId-buildNumber` がプラグインで使用されます。

Jenkins のパイプラインで脆弱性のセキュリティ制御を定義

`contrastAgent` というパイプラインステップを使用すると、Contrast エージェントをダウンロードして、アプリケーションにエージェントを組み込み、実行することができます。
`contrastVerification` というパイプラインステップを使用すると、アプリケーションを検証し、セキュリティ制御のパラメータを設定できます。

contrastAgent でダウンロード

`contrastAgent` という名前のパイプラインステップは、最新の Contrast エージェントをダウンロードします。

パラメータ	必須項目	説明	例
`profile`	必須	Contrast との通信に使用する Contrast 接続プロファイル	`MyConnection`
`outputDirectory`	必須	ダウンロードしたエージェントを配置する場所を定義	`env.WORKSPACE`
`agentType`	`applicationId` が定義されていない場合は必須	アプリケーションに組み込んだエージェントのタイプ(大文字と小文字の区別なし) オプション： `Java`、 `.NET`、 `.NET_Core`、 `Node`、 `Ruby`、 `Python`	`Java`

`contrastAgent` という名前のパイプラインステップを追加する例を次に示します。

```
node{
  stage('Download Latest Contrast Agent'){
    contrastAgent profile:'MyConnection', outputDirectory: \
env.WORKSPACE, agentType: 'Java'
```

```
  }
}
```

contrastVerification でアプリケーションを検証

`contrastVerification` という名前のパイプラインステップを使用して、アプリケーションが脆弱かどうか検証できます。

パラメータ	必須項目	説明	例
`profile`	必須	`profile` を使用して、Contrast との通信に使用する接続を指定します。	Contrast Connection
`queryBy`	必須	`queryBy` を使用して、ビルドに関連する脆弱性にフィルタをかけます。オプション 1、2、4 の場合、この値は、アプリケーションの実行中に Contrast エージェントに渡された `contrast.override.appversion` パラメータと一致する必要があります。 脆弱性をクエリする方法をオプション番号で指定します(デフォルトの値は 1) : 1. `appVersionTag, format: applicationId-${BUILD_NUMBER}` 2. `appVersionTag, format: applicationId-${JOB_NAME}-${BUILD_NUMBER}` 3. `startDate` (これはビルドのタイムスタンプです。ビルド開始後に検出された脆弱性のみを調べます。) 4. `APPVERSIONTAG` (これは、ジョブパラメータか環境変数です。独自のテキストを指定する場合は、このオプションを選択し、Jenkins ジョブ内の環境変数として APPVERSIONTAG をエクスポートしてください。`JOB_NAME` と `BUILD_NUMBER` の両方は、既に Jenkins の環境変数として利用可能です。)	1
`applicationId`	`applicationName` と `agentType` が定義されていない場合は必須	検証しようとしているアプリケーションまたはアプリケーションモジュールの ID	`cb3ea678-38c8-4487-ba94-692a117e7966`
`applicationName`	`applicationId` が定義されていない場合は必須	検証しようとしているアプリケーションの名前(大文字と小文字の区別なし)	MyApp
`count`	オプション	許容する脆弱性の総数、デフォルトは 0	10
`rule`	オプション	デフォルトは **All**	`xss`
`severity`	オプション	デフォルトは **All**。その他のオプション : **Critical**、 **High**、 **Medium**、**Low**	**High**
`appVersionTag`	オプション	Contrast エージェントの `contrast.override.appversion` パラメータに渡された値	v1.2.3

`contrastVerification` という名前でパイプラインステップを追加する方法の例をいくつか示します。

- **`queryBy startDate` を使用 :**

```
contrastVerification applicationId: '1e6ad9c6-89d4-4f06-
bdf6-92c569ec89de', count: 1, profile: 'new-profile', queryBy: 3, \
rule: 'cache-controls-missing', severity: 'High'
```

- **`queryBy` のカスタム `appVersionTag` パラメータを使用 :**

```
contrastVerification applicationId: '1e6ad9c6-89d4-4f06-
bdf6-92c569ec89de', count: 1, profile: 'new-profile', queryBy: 4, \
appVersionTag: 'v1.2.3' rule: 'cache-controls-missing', severity: 'High'
```

- **`applicationName` と `AgentType` を使用してアプリケーションを定義：**

```
contrastVerification applicationName: 'MyApp', agentType: 'Java', count: \
1, profile: 'new-profile', queryBy: 3, rule: 'cache-controls-missing', \
severity: 'High'
```

- **事前設定またはオーバライドされている脆弱性のセキュリティ制御でアプリケーションを検証します。**
  Contrast (763ページ)で脆弱性のセキュリティ制御が事前に設定されていることがわかっている場合は、プロファイルおよび、`applicationId` か(`applicationName` と `agentType`)のいずれかを定義するだけで良いです。

```
contrastVerification applicationId: '35ae7b89-1c76-414b-b317-
c444ce27608b', profile: 'ContrastConnection'
```

Jenkins でのセキュリティ制御

Jenkins で、以下のセキュリティ制御を定義できます。

- システムレベルでの制御 (760ページ)
- ビルド後の処理としての制御 (761ページ)
- パイプラインでの制御 (761ページ)

ジョブ結果ポリシーの定義

ジョブ結果ポリシー(Contrast Jenkins プラグインのバージョン 3.3 以降でサポート)は、Contrast プラグインを使用する Jenkins ジョブにビルド結果を割り当てます。ジョブ結果ポリシーにより、設定した基準に基づいて、ジョブのビルド結果に **FAILURE**(失敗)、**UNSTABLE**(不安定)、**SUCCESS**(成功)などのステータスが付けられます。

開始する前に

Contrast でジョブ結果ポリシーを定義するには、組織の管理者である必要があります。

ポリシーを定義

ジョブ結果ポリシーを定義するには：

1. Contrast Web インターフェイスで組織の設定 (818ページ)を選択し、左のナビゲーションで**インテグレーション**を選択します。
2. **Jenkins** の行で、**ジョブ結果ポリシーを追加**を選択します。

3. ジョブ結果ポリシーの名前を定義します(必須)。
4. **アプリケーション**で、ポリシーを適用するアプリケーションを指定します。アプリケーションは、名前や重要度、またはタグで指定できます。アプリケーション名で選択する場合、個々のアプリケーションやマージされたアプリケーション (532ページ)を選択できます。
5. **脆弱性プロパティ**では、ポリシーの対象とする脆弱性の制限や環境などを定義します。**環境**、**脆弱性ステータス**、**最初に検知された脆弱性**を使用して、ポリシーの対象とする脆弱性を絞り込みます。 **脆弱性ルールと深刻度**を使用して、ジョブ結果のステータス変更のトリガーとなるルール(特定の深刻度)とその数をしきい値として指定します。
   - **環境：** ポリシーを適用する環境を選択します(複数選択可)。
     例えば、テスト(QA)環境にある脆弱性が本番環境で発生してしまうのを防ぐには、**QA** を選択します。 (ただし、この場合、開発環境の脆弱性は考慮されません。それらの脆弱性を対象とするには、**開発環境**も選択します。または、すべての環境の脆弱性を対象とする場合は、**全ての環境**を選択します。)
   - **脆弱性ステータス：** どのステータスを対象とするかを選択します(複数選択可)。脆弱性のステータス (703ページ)は、Contrast で決定されます。

     > **ヒント**
     > ほとんどの場合、**報告済**、**確認済**、**疑わしい**などのオープン中のステータスのみを選択します(**問題無し**、 **修復済**や**修正完了**などのクローズのステータスではなく)。そうすれば、既に開発者が対応した脆弱性によって、Jenkins のジョブが失敗したり不安定になることがありません。

   - **最初に検知された脆弱性：** 脆弱性が最初に検知された時間に基づいて、ポリシーの対象とする脆弱性の時間範囲を設定します。

開始と**終了**フィールドを使用して、時間範囲の開始と終了を設定します。時間範囲の開始に、**ジョブ開始時**か、Contrast ステップを実行する前の所定の日数、または**アプリケーションをオンボード**した日を選択します。 時間範囲の終了に、Contrast ステップを実行する前の所定の日数か、Jenkins で Contrast ステップを実行するまで、または特定の期間のオプションを選択します。**カスタマイズ**を選択して、いずれかのフィールドに特定の日付を選択することもできます。
脆弱性がこの時間範囲外で最初に検出された場合、その脆弱性はポリシーの対象にはなりません。

ヒント
開発者に一定期間(例えば、1 週間)内に脆弱性を修正させるためには、7 日以上前に検出された脆弱性のみをポリシー違反とさせるよう、ポリシーを定義します。これを行うには、**開始**に**アプリケーションのオンボード時**を、**終了**に**過去 7 日間**を設定します。

- **脆弱性ルールと深刻度：** このセクションを使用して、ポリシーで許可する脆弱性の件数、種類および深刻度によるしきい値を設定します。
  ルールごとに、**深刻度**か **Assess ルール**を選択したら、**許容する脆弱性数**を選択します。**別のルールを追加**をクリックすれば、複数のルールを登録できます。
  許容される脆弱性数は、この深刻度の脆弱性を、このビルドに影響を与えずに許容する脆弱性の数を決定します。「0」に設定すると、1 つの脆弱性でビルド結果のステータスが変更されます。「10」に設定した場合は、指定した種類の脆弱性が 11 件検出されるまで、ビルド結果のステータスは変わりません。特定のルールや深刻度に対して、**許容される脆弱性数**を空欄のままにすると、そのルールや深刻度の*全ての脆弱性*が許容されます。
  例えば、**全てのルール**と 1 件の脆弱性を設定した場合、1 つの脆弱性がポリシーのトリガーとなります。また、別のルールを追加して、このポリシーを 5 つの重大な脆弱性のルールと 2 つのクロスサイトスクリプティングの脆弱性に制限することもできます。
- **脆弱性にフィルターをかける時にプラグインの"Query vulnerabilites by"で選択したオプションを適用します**の横のチェックボックスにチェックをします。Contrast Assess のビルド後の処理 (761ページ)またはパイプラインステップ (762ページ)を使用して、Jenkins ジョブの脆弱性をクエリするよう定義できます。例えば、`AppVersionTag` や、脆弱性が最後に検出された日などを使用できます。このチェックボックスを選択すると、ジョブ結果ポリシーの評価時にこのクエリが含まれるようになります。

以下は、ジョブ結果ポリシーの可能なルールと設定で、これらの条件が満たされた場合に Jenkins での結果ステータスを変更する例です。

この例では、次のような脆弱性がポリシー違反の対象となります。
- QA 環境として指定されたサーバで検出された全ての脆弱性
- ステータスが**報告済**、 **確認済**、**疑わしい**全ての脆弱性
- アプリケーションのオンボード時から Jenkins で Contrast ステップが実行されるまでの間に初めて検出された全ての脆弱性

次のいずれかのうち少なくとも 1 つが発生すると、ポリシー違反となり、結果ステータスが変更されます。
- **重大**な脆弱性が 1 つ以上ある
- SQL インジェクションを除く*全*てのルールで、**高**、**中**、**低**または**注意**の深刻度の脆弱性が合計で 4 つ以上ある

> **注記**
> 脆弱性は 1 回だけカウントされ、最も具体性が高い設定(例、特定の脆弱性ルール)が、最も具体性が低い設定(**全てのルール**)より優先されます。ルールの種類と深刻度の両方で脆弱性の制限が設定されている場合、その脆弱性はルールの種類の数としてカウントされますが、深刻度の脆弱性としてカウントされません。従って、この例では、**重大**な脆弱性は深刻度の脆弱性数としてカウントされますが、**高**、**中**、**低**および**注意**の深刻度は**全てのルール**としてまとめられます。

6. **ポリシー結果**で、ポリシーの結果を選択します。選択した基準に一致するジョブを Contrast で**失敗**(FAILURE)、 **不安定**(UNSTABLE)または**成功**(SUCCESS)のステータスにします。複数のジョブ結果ポリシーが指定されているアプリケーションには、違反しているすべてのポリシーで最も重大な結果が適用されます。
7. **このジョブ結果ポリシーを有効化**の横にあるチェックボックスをオフにすると、個々のポリシーを削除しなくても、Jenkins ジョブに Contrast のポリシーが適用されるのを一時停止できます。

Jenkins でのビルドの実行

初めてビルドを実行するには：

1. Jenkins で、実行するジョブまたはプロジェクトを選択します。
2. 左側のメニューで、**Build Now** を選択します。
3. 詳細を見るには、ログ出力を表示します。
4. フリースタイルのジョブを使用している場合は、タスクからのデータを表示できます。実行を選択して、左側のメニューで **Vulnerability Report** を選択します。

> **重要**
> ジョブまたはプロジェクトの概要でグラフを見ることもできます。ただし、Contrast パイプラインステップを使用した場合、または 1 つ以上の選択されたアプリケーションがジョブ結果ポリシーで上書きされている場合は、グラフが表示されません。

Jira とのインテグレーション

Contrast と Jira を連携し、チケットの自動生成、コメントの同期、アプリケーションのプッシュ通知などを可能にします。

設定を開始する前に、以下が必要です。

- Jira Cloud または Jira 8。
- Jira アカウントの認証情報。Jira Cloud の場合は、これはユーザ名と API キーになります。オンプレミスの Jira をインストールしている場合は、ユーザ名とパスワードです。
- 対象プロジェクトで課題を作成するための権限。
- HTTP 経由で Contrast にアクセス可能な Jira の実行インスタンス。
- Contrast に登録済みのアプリケーションと関連付けるプロジェクト

関連項目

- Jira に接続する (767ページ)
- Assess と Jira の設定 (768ページ)
- サーバレスと Jira の設定 (769ページ)
- Jira 認証情報の管理 (771ページ)

Jira に接続する

Jira とインテグレーションするには：

Contrast で Jira を設定

1. Contrast Web インターフェイスで、**ユーザメニュー > 組織の設定 > インテグレーション**を選択します。
2. Jira の行にある**接続**を選択します。
3. **Jira への接続の設定**画面で、Jira インテグレーションの名前、ユーザ名および API キー(もしくは、オンプレミスの場合は Jira のパスワード)を入力します。Jira インスタンスの URL も入力したら、Contrast が URL にアクセスできるかを確認します。
   - Jira インテグレーションの名前を**接続名**に入力します。
   - Jira インスタンスの **URL** を入力します。
   - **Assess** か **Serverless** を選択します。

> **注記**
> Contrast では、ユーザ名、API キー(またはパスワード)および URL が認証情報セットとして、このインテグレーション用に保存されます。

4. **接続をテスト**を選択します。Jira プロジェクトが多数ある場合は、接続テストに数分かかることがあります。接続テストでは、Contrast が指定の Jira インスタンスにアクセスでき、ユーザがログインできることを確認します。
5. 接続をテストしたら、Assess と Jira の設定 (768ページ)または サーバレスと Jira の設定 (769ページ)を行います。

関連項目

- Assess と Jira の設定 (768ページ)
- サーバレスと Jira の設定 (769ページ)
- 認証情報の管理 (771ページ)

Assess と Jira の設定

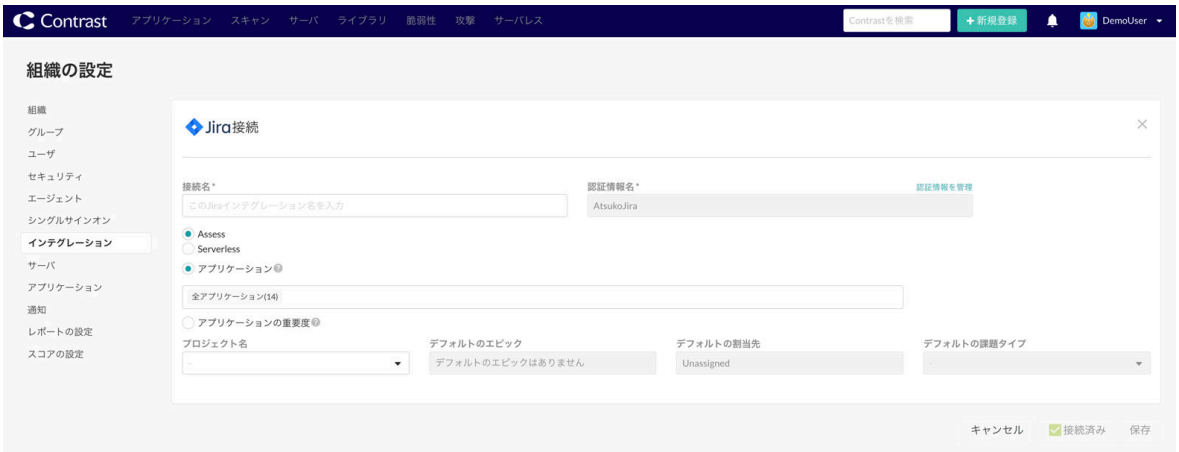
Jira との接続をテストしたら、指定したトリガーに基づいて Jira チケットを作成するように設定できます。

開始する前に

- Jira と接続 (767ページ)ができることを確認してください。

手順

1. Contrast から Jira に接続できたら、**アプリケーション**フィールドをクリックして、セキュリティの問題に対して Jira チケットのトリガーとなるアプリケーションを指定します。また、Contrast で特定の重要度が設定されているアプリケーションに対してのみ、Jira チケットをトリガーすることもできます。その場合は、**アプリケーションの重要度**フィールドを選択して、Jira チケット作成のフィルターとして使用する重要度を指定します。



2. **プロジェクト名**、**デフォルトのエピック**、**デフォルトの割当先**、**デフォルトの課題タイプ**フィールドを使用して、Contrast 側で作成する Jira チケットの値を選択します。Contrast での脆弱性の深刻度レベルと Jira の優先度をマップすることもでき、担当者間でセキュリティチケットが管理しやすくなります。その他の Jira フィールドに事前入力したい場合は、**JIRA フィールドを追加**を選択します。ドロップダウンから、追加するフィールドとフィールドのデフォルト値を選択します。

> **注記**
> **プロジェクト名**や**デフォルトの課題タイプ**を変更すると、関連する Jira のフィールドや選択できる値も変わります。Contrast Web インターフェイスでは選択済みの値が保持され、新たに選択したプロジェクトや課題タイプに対しても適用されます。

3. Jira 側で課題がクローズや再オープンされるたびに Contrast の脆弱性ステータスも更新したい場合は、**双方向のインテグレーションを有効にする**のオプションを選択してください。このオプションを選択すると、チェックボックスの下に URL が生成されます。Jira 管理者は、この URL を使用して Jira に Webhook を登録してください。
   脆弱性ステータスのドロップダウンを使用して、Jira チケットのステータス更新によって Contrast の脆弱性ステータスをどのように更新するかを設定します。

   > **注記**
   > **問題無し**のステータスを選択する場合、ドロップダウンで**理由**を選択する必要があります。デフォルトでは、ドロップダウンの**上記以外**が選択されます。

   双方向のインテグレーションを保存すると、関連する Jira チケットのステータス変更が Contrast で自動的に追跡されます。追跡情報は、脆弱性の**アクティビティ**タブにコメントとして表示されます。各コメントには、Jira インテグレーション名とチケットへのリンクが含まれます。

   > **注記**
   > Atlassian は、https 以外の URL で Webhook を登録する機能を廃止しました。そのため、Contrast のオンプレミス版をご利用のお客様は、Jira の双方向インテグレーションを有効にする前に HTTPS を設定 (892ページ)する必要があります。

4. Contrast で脆弱性の検出時に Jira チケットを新規に作成したい場合は、**新たに検知された脆弱性に対してチケットを自動的に作成する**のオプションを選択してください。そして、どの**深刻度**や**ルール**を Jira チケットのトリガーとするかを選択します。
   複数の脆弱性を Jira の 1 つのチケットとして作成した場合、チケットのステータスは、そのチケットに関連付けられている全ての脆弱性に適用されます。1 つの脆弱性に対して複数のチケットを作成した場合は、Contrast で脆弱性をクローズする前に、全ての Jira チケットをクローズする必要があります。

   > **注記**
   > 自動作成のオプションは遡及しないため、過去の脆弱性に対する Jira チケットは生成されません。

5. **保存**を選択して、Jira インテグレーションの使用を開始します。インテグレーションを削除するには、**設定を削除**を選択します。

サーバレスと Jira の設定

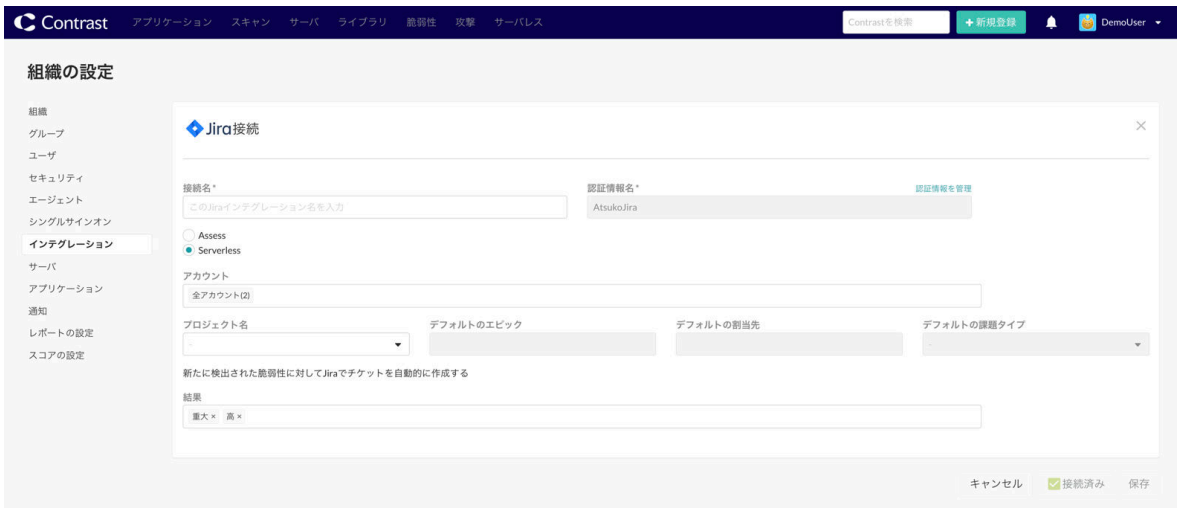
Jira の接続をテストした後に、インテグレーションによって Jira チケットを自動作成するために、脆弱性の深刻度と種類のサブセットを設定することができます。

開始する前に

- Jira に接続 (767ページ)できることと、 Contrast に AWS アカウントがあることを確認してください。

手順

1. Jira との接続を確認したら、**設定を追加**をクリックして、検査結果から Jira チケットを作成する AWS アカウントを設定していきます。



2. 以下の情報を入力します。
   - **接続名：** 作成する Jira インテグレーション名を入力します。
   - **認証情報名：** 認証に使用するログイン情報を指定します。
3. 認証情報を編集または新規に作成するには、*認証情報名*フィールドの上にある**認証情報を管理**をクリックします。
4. 以下の情報を入力します。
   - 認証情報名：前の手順で入力した名前
   - URL：Jira インスタンスの URL
   - ユーザ名
5. **API キー**を入力します。Contrast からのコールを認証するために、Jira の API キーを指定する必要があります。
   - API キーフィールド名の横にある情報アイコンをクリックして、*開始する*のリンクをクリックします。
   - 「API トークンを作成する」を選択します。
   - ラベルを入力します。
   - 作成をクリックします。
   - 「コピー」をクリックして、API トークンをクリップボードにコピーします。
   - Contrast Web インターフェイスに戻り、その API トークンを API キーフィールドに貼り付けます。
6. **接続をテスト**をクリックして、接続を確認します。
7. **保存**または**完了**をクリックします。
8. 特定のアプリケーションやアカウントに関連する脆弱性の Jira チケットを作成するために、**Serverless** オプションを選択します。
9. Jira チケットを作成したい**アカウント**を選択します。
10. チケットを作成する Jira の**プロジェクト名**を選択します。
11. **デフォルトのエピック**を設定します。新しいチケットを作成した場合のエピックを指定します(該当する場合)。
12. **デフォルトの割当先**を設定します。チケットを割り当てる担当者を指定します。
13. **デフォルトの課題タイプ**を設定します。作成するチケットのデフォルトの課題タイプ(タスク、ストーリー、バグなど)を選択します。

> **注記**
> **プロジェクト名**または**デフォルトの課題タイプ**を変更すると、関連する Jira のフィールドや選択できる値も変わります。 Contrast Web インターフェイスでは選択済みの値が保持され、新たに選択したプロジェクトや課題タイプに対しても適用されます。

14. チケットを作成する**結果**の種類を選択します。
    - Permissions(権限)
    - Exploits(脆弱性)
    - Dependencies(CVE)
    - 深刻度(注意、低、中、高、重大)
15. チケットを作成する**結果**の種類を選択します。
    - **Jira フィールドを追加**を選択します。
    - 報告者フィールドやワークタイプフィールドなどを追加します。
16. 必要に応じて、Contrast での脆弱性の深刻度レベルを、該当する Jira の優先度レベル(Critical、Major、Standard など)にマッピングします。
17. **保存**を選択して、Jira インテグレーションの使用を開始します。インテグレーションを削除するには、**設定を削除**を選択します。

以下のビデオで、インテグレーション手順のデモをご覧になれます。

https://player.vimeo.com/video/827314961?h=f049a72b04

新規に脆弱性を検出した場合に関しては、以下のビデオを参照してください。

https://player.vimeo.com/video/827318792?h=78a2e50f7f

Jira 認証情報の管理

Contrast には、Jira インテグレーションの設定で入力された最新の認証情報が保存されているので、次の新しい接続を簡単に設定できます。最初の設定で入力したユーザ名、API キー(またはパスワード)と Jira の URL が、それ以降の Jira インテグレーション設定でデフォルトの認証情報となります。以降の Jira 設定では、このデフォルトの認証情報がフィールドにあらかじめ入力されますが、必要に応じて値を変更できます。保存されている認証情報を管理して、影響を受ける全ての設定を同時に更新することもできます。

デフォルト設定とは異なる認証情報にするために、1 つの設定を作成・編集するには：

1. **ユーザメニュー > 組織の設定 > インテグレーション**にアクセスします。
2. **設定を表示**を選択し、既存の Jira インテグレーションの一覧を表示します。更新するインテグレーション名を選択します。
3. **認証情報を管理**を選択して、Jira 接続の設定情報を表示します。
4. **URL** フィールドでドロップダウンメニューを使用して保存されている認証情報セットを選択するか、**URL**、**ユーザ名**、**API キー**(または**パスワード**)を手動で更新します。
5. フィールドを更新したら、**接続をテスト**を選択し、変更した情報で問題なく接続できることを確認します。
6. **保存**を選択します。

> **注記**
> 新しい認証情報を使用する場合、指定した名前で既存の認証情報セットを上書きするか、別の名前で新しい認証情報セットとして保存するかを選択する必要があります。

複数の Jira 設定を同時に編集するには：

1. Contrast Web インターフェイスで、**ユーザメニュー > 組織の設定 > インテグレーション**を選択します。
2. Jira インテグレーションの**認証情報を管理**を選択します。
3. **管理 Jira 認証情報**の画面で、ドロップダウンを使用して保存されている認証情報セットを選択します。
4. ユーザ名、API キー(またはパスワード)、Jira の URL を編集します。
5. 編集した認証情報に別の名前を使用する場合は、**名前を変更**を選択します。

> **注記**
> 認証情報セットを更新すると、そのセットを使用している全ての設定に影響します。

6. **接続をテスト**を選択して、インテグレーションが機能することを確認します。
7. **保存**を選択します。

> **注記**
> 認証情報セットを更新すると、このセットを使用する全ての設定が更新されます。

関連項目

- Jira とのインテグレーション (767ページ)
- Jira に接続する (767ページ)
- Assess と Jira の設定 (768ページ)
- サーバレスと Jira の設定 (768ページ)

Contrast Maven プラグイン

Maven は、Java アプリケーションのビルド、パッケージ化、およびテストを行うビルドツールです。

Contrast Maven プラグインにより、Contrast Asses や Contrast Scan をプロジェクトの Maven ビルドに連携できます。

以下の詳細については、Contrast Maven Plugin Reference Documentation を参照してください。

- ゴール
- 使用方法

ゴール

- **スキャン**(scan)： scan ゴールは、Contrast Scan を使用して Maven プロジェクトのアーティファクトを解析し、静的分析により脆弱性を検出します。
- **インストール**(install)：install ゴールは、統合テストに Contrast Java エージェントを組み込み、Contrast Assess によるランタイムのセキュリティ分析を行います。
- **検証**(verify)：verify ゴールは、統合テスト中に Contrast Assess で検出される脆弱性により、プロジェクトのセキュリティポリシーに違反していないことを検証します(違反が検知された場合はビルドは失敗します)。

関連項目

- Contrast Scan (546ページ)
- Java エージェントのインストール (77ページ)

Microsoft Teams とのインテグレーション

インテグレーションを設定して、Microsoft Teams のインスタンスで Contrast からの通知を受け取るようにします。

Microsoft Teams に接続するために、以下を行います。

1. Microsoft Teams アカウントで自分のチームにアクセスし、通知を受信するチャネルを選択します。
2. **その他のオプション**アイコンを選択します。
3. メニューから**コネクタ**を選択します。
4. **Incoming Webhook** の**構成**を選択します。
5. 着信 Web フック(Incoming Webhook)の名前を入力して、**作成**をクリックします。
6. Webhook の URL をクリップボードにコピーします。この URL を使用して、Contrast のインテグレーションを設定します。
7. **完了**を選択します。
8. Contrast Web インターフェイスにログインし、**ユーザメニュー**から**組織の設定 > インテグレーション**を選択します。
9. Microsoft Teams の行で、**接続**を選択します。
10. インテグレーション名を入力します。
11. URL のフィールドに、Microsoft Teams からコピーした Webhook の URL をペーストします。
12. 通知を有効にするアプリケーションを選択します。
13. **保存**を選択します。

重要
このインテグレーションによる接続は、Contrast で 5 回試行しても応答しない場合、切断されます。

PagerDuty とのインテグレーション

インテグレーションを設定して、PagerDuty のインシデント管理で Contrast から攻撃の通知を受け取るようにします。

PagerDuty に接続するために、以下を行います。

1. Contrast Web インターフェイスにログインし、**ユーザメニュー**から**組織の設定 > インテグレーション**を選択します。
2. PagerDuty の行で、**接続**を選択します。
3. 設定画面で、インテグレーションの**名前**を入力します。この名前が、Contrast からの通知に表示されます。
4. **メッセージの重大度**のドロップダウンで、警告の深刻度レベルを選択します。デフォルトでは、「Critical」が選択されます。インシデントの深刻度レベルの詳細については、PagerDuty のドキュメントを参照してください。
5. **インテグレーションキー**を入力します。このフィールドに入力するインテグレーションキーを取得するには、PagerDuty のドキュメントの手順に従ってください。
6. Contrast から PagerDuty でインシデントを自動生成するアプリケーションを**アプリケーション**で選択します。 デフォルトでは、「全てのアプリケーション」が選択されます。
7. 全てのフィールドに入力したら、**接続をテスト**を選択します。この処理は、PagerDuty プロジェクトの数によっては、数分かかる場合があります。接続テストでは、Contrast が PagerDuty のインスタンスに到達できて、メッセージが送信できるかを確認します。

組織の通知 (831ページ)で、PagerDuty と連携させる通知を管理してください。

重要
このインテグレーションによる接続は、Contrast で 5 回試行しても応答しない場合、切断されます。

Solutions Business Manager とのインテグレーション

インテグレーションを設定して、Contrast からの通知を Solutions Business Manager で受け取るようにします。

設定を開始する前に、以下が必要です。

- Solutions Business Manager アカウントの認証情報(ユーザ名とパスワード)
- HTTP 経由で Contrast にアクセス可能な Solutions Business Manager の実行インスタンス
- Contrast に登録されたアプリケーションと関連付けるプロジェクト

Solutions Business Manager に接続するために、以下を行います。

1. **ユーザメニュー**で、**組織の設定 > インテグレーション**にアクセスします。
2. Solutions Business Manager の行で、**接続**をクリックします。
3. **Serena** への接続ページで、インテグレーションの名前を入力します。この名前は、バグ管理システムに検出結果を送信する際に表示されます。
4. Solutions Business Manager インスタンスに接続するアカウントのユーザ名を入力します。
5. 指定したユーザ名のパスワードを入力します。
6. **ホスト**フィールドに、Solutions Business Manager インスタンスへの URL を入力します。
7. Solutions Business Manager インスタンスにマップするアプリケーションを選択します。
8. そのアプリケーションと関連付ける **Solutions Business Manager のプロジェクト ID** を入力します。
9. **接続をテスト**を選択して、通信を確認します。接続テストは、指定したプロジェクトの存在を確認するだけではなく、Contrast がインスタンスと通信し認証されるかを確認します。

ServiceNow とのインテグレーション

ServiceNow と Contrast を連携させて、ServiceNow でインシデントを自動的に生成することができます。

設定を開始する前に、以下が必要です。

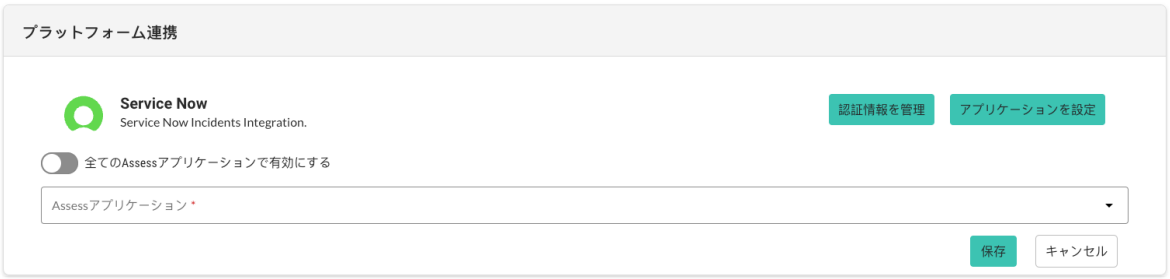
- ServiceNow の URL
- ServiceNow のユーザ名
- ServiceNow のパスワード

ServiceNow に接続

ServiceNow とインテグレーションするには：

1. Contrast Web インターフェイスで、**ユーザメニュー** > **組織の設定** > **インテグレーション**を選択します。
2. ServiceNow の行にある**認証情報を管理**を選択します。
3. ServiceNow の URL、ユーザ名、パスワードを入力します。

4. **保存**を選択します。
5. **アプリケーションを設定**を選択します。
6. 全ての Assess アプリケーションに対してインテグレーションを有効にするか、ドロップダウンリストから特定のアプリケーション名を選択します。



7. **保存**を選択します。

再試行の仕組み

イベントが失敗した場合、そのイベントは保存されて、GMT 時間の毎晩午前 0 時(日本標準時は GMT+09 時)頃に再試行されます。再試行の回数は 1 回から最大 3 回までになり、最大 72 時間行われます。

`create vulnerability` イベントが失敗し、保存された場合、失敗したイベントに関連する更新や削除の操作は、正しい状態を維持するために時系列で保存および再生されます。

Slack とのインテグレーション

Slack とのインテグレーションにより、Slack 側で Contrast からの通知 (831ページ)をアプリ内通知と同様の形式で受け取ることができます。

Slack に接続するために、以下を行います。

1. Slack で、チームの**アプリの管理**画面にアクセスします。
2. **Incoming Webhook** を検索し、Slack に追加します。
3. Contrast からの通知を受信するチャネルを選択して追加します。
4. **Webhook URL** をコピーします。
5. Contrast Web インターフェイスにログインし、**ユーザメニュー**から**組織の設定 > インテグレーション**にアクセスします。
6. Slack の行で、**接続**を選択します。
7. インテグレーションの名前を入力して、URL を貼り付けます。
8. 通知を有効にするアプリケーションを選択します。
9. **保存**を選択します。

次の手順でインテグレーションを確認します。

1. **組織の設定 > 通知**を選択します。
2. **インテグレーション**のドロップダウンで、Slack のインテグレーション名を選択します。

3. 通知を受けたい**イベントタイプ**(通知の登録)ごとに、**インテグレーション**列のトグルボタンをクリックします。
4. イベントを発生させて、指定した Slack チャネルで通知を受信していることを確認してください。

重要
このインテグレーションによる接続は、Contrast で 5 回試行しても応答しない場合、切断されます。

Splunk On-Call(旧 VictorOps)とのインテグレーション

インテグレーションを設定して、VictorOps のインシデント管理で Contrast から攻撃の通知を受け取るようにします。

VictorOps に接続するために、以下を行います。

1. Contrast Web インターフェイスにログインし、**ユーザメニュー**から**組織の設定 > インテグレーション**を選択します。
2. VictorOps の行で、**接続**を選択します。
3. インテグレーションの**名前**を入力します。この名前が、Contrast からの通知に表示されます。
4. ドロップダウンを使用して、警告の**メッセージタイプ**を選択します。デフォルトでは、「Critical」が選択されます。メッセージタイプの詳細については、VictorOps のドキュメントの Incident Fields を参照してください。
5. 接続の **URL** を入力します。VictorOps の URL は、REST API エンドポイントを使用して生成できます。URL の取得やその他の情報については、VictorOps のドキュメントの REST endpoint を参照してください。
6. **URL をテストする**を選択します。この処理は、VictorOps プロジェクトの数によっては、数分かかる場合があります。接続テストでは、Contrast が VictorOps のインスタンスに到達でき、指定したユーザがログインできるかを確認します。
7. 接続が確立したら、複数選択フィールドをクリックして、通知を送信する**アプリケーション**を選択します。デフォルトでは、「全てのアプリケーション」が選択されます。

重要
このインテグレーションによる接続は、Contrast で 5 回試行しても応答しない場合、切断されます。

Contrast Visual Studio プラグイン

Visual Studio プラグインを使用すると、Contrast エージェントが組み込まれたアプリケーションの脆弱性情報を Visual Studio IDE から確認できます。

このプラグインによって、脆弱性の影響を受けるコード行が Visual Studio 内で表示され、関連する情報を Contrast Web インターフェイスで確認することができます。この方法により、開発者は開発時にアプリケーションのセキュリティに関するフィードバックを得ることができ、迅速な修正が可能になります。

このプラグインは、Visual Studio のバージョン 2017 (15.0 以降)、2019、2022 をサポートしています。

Visual Studio プラグインをインストールして設定し、使用するには：

1. Visual Studio で、**拡張機能**を選択します。
2. 拡張機能の管理画面で、左ナビゲーションパネルから**オンライン**を選択します。
3. 「Contrast」を検索して、**ダウンロード**を選択します。
4. ダウンロードが完了したら、IDE を再起動します。
5. Visual Studio で、**ツール > オプション**の順に移動します。
6. 検索フィールドに「Contrast Security」と入力して、**Contrast Security - Connection** を選択します。
7. **Contrast Connection** 画面で、適切なフィールドに **Contrast URL**、**ユーザ名**、**サービスキー**、**API キー**および**組織 ID** を入力します。これらの情報はプロファイル (522ページ)で確認できます。

> **注記**
> API キーは、アクセスしようとする組織に属している必要があります。そうでない場合は、認証エラーが発生します。試行に何度も失敗すると、アカウントがロックされます。

8. **追加**を選択します。Visual Studio が自動的に接続をテストして、Contrast からの組織の取得を試みます。
9. **Organizations** フィールドで組織を選択して、**OK** を選択します。
10. Visual Studio で、**表示 > その他のウィンドウ > Contrast Security Integration** にアクセスします。「Contrast Security Integration」を検索することもできます。このビューには、Contrast からの全ての脆弱性の一覧が表示されます。
11. 一覧を絞り込むには、ページの右上にある**フィルター**アイコンをクリックします。
12. 表示される画面で、サーバ、アプリケーション、深刻度レベル、状態、最後の検出日などの複数のフィルターから選択します。
13. 脆弱性の一覧が表示されない場合は、**Refresh** を選択します。選択済フィルターを全てクリアするには、クリアアイコンをクリックします。これは、サーバおよびアプリケーションの一覧にも適用されます。

> **注記**
> 一覧を更新しても脆弱性が表示されない場合は、脆弱性を絞り込む必要があります。フィルターおよび脆弱性が適切に更新されるよう、**Connection** 設定で異なる組織を選択して、このプロセスを繰り返す必要があります。

14. **Actions** 列で、虫眼鏡アイコンをクリックすると、脆弱性の詳細情報を確認できます。このアイコンを使用すると、Contrast 内の**脆弱性**ページに移動します。

Contrast Visual Studio Code プラグイン

Visual Studio Code プラグインを使用すると、機能テスト中に Contrast でセキュリティの問題が検出されたときに、検査対象のアプリケーションの脆弱性情報を Visual Studio Code 環境から確認できます。

このプラグインによって、アプリケーションで検出された全ての脆弱性の概要に加え、脆弱性にさらされている HTTP リクエストなど、各脆弱性の詳細情報が表示されます。

このプラグインは Visual Studio Code のバージョン 1.42.1 以降をサポートしています。

Visual Studio Code プラグインをインストールして設定し、使用するには：

1. Visual Studio Code で、**拡張機能**の画面にアクセスし、「Contrast Security」を検索します。
2. **インストール**を選択します。インストールが完了したら、Visual Studio Code を再起動します。
3. **Contrast Security** のビューから**設定**アイコンを選択して、Contrast アカウントを認証します。
4. **ワークスペース**を選択し、**API キー**、**組織 ID**、**Contrast URL**、**認証ヘッダ**を各フィールドに入力します。これらの情報はプロファイル (522ページ)で確認できます。

5. **Test Contrast connection** アイコンを選択して、認証情報を確認します。接続に成功したか、または無効な認証情報についてのメッセージが表示されます。
6. **リフレッシュ**アイコンを選択すると、脆弱性情報が更新されます。**CONTRAST SECURITY** 下に、**深刻度** 別にグループ化された脆弱性が**ステータス**順に表示されます。脆弱性を選択すると、**概要**、**詳細**、**HTTP 情報**、**修正方法** などの詳細が表示されます。
また、最後に脆弱性が検出された日時や現在のステータスも表示されます。脆弱性の詳細は、**出力**のコードエディターに表示されます。

ヒント

プラグインを使用すると、以下の項目で脆弱性を絞り込めます。

- 脆弱性メタデータ：
  - アプリケーション名
  - ステータス(報告済、問題無し、修復済など)
  - 環境(開発、テスト、本番)
  - タグ(脆弱性に適用されたカスタムラベル)
  - 検出日(特に、最初と最後の検出日)
- セッションメタデータ：
  - コミッタ
  - コミットハッシュ
  - ブランチ名
  - Git タグ
  - リポジトリ
  - テスト実行
  - バージョン
  - ビルド番号

例えば、ユーザは特定の機能ブランチ(ブランチ名)で検出され、自分(コミッタ)が直接コミットした脆弱性のみを表示するよう選択して、別の機能ブランチで別の開発者によってもたらされた脆弱性を除外することができます。

他のユーザは、脆弱性を絞り込んで、自分のセキュリティチームによってブロックされた、特定のビルド(ビルド番号)からの結果のみを表示することができます。マージされた機能ブランチをデプロイする前に解決する必要のある脆弱性のサブセットを直ちに特定できます。

Contrast Visual Studio for Mac プラグイン

Visual Studio for Mac プラグインを使用すると、Contrast エージェントが組み込まれたアプリケーションの脆弱性情報を Visual Studio for Mac IDE から確認できます。

このプラグインにより、コード行が表示され、ユーザは Contrast Security 用のウィンドウで詳細を確認できます。この方法により、開発者はアプリケーションのセキュリティに関するフィードバックを開発時に取得でき、迅速な修復につながります。

このプラグインは Visual Studio for Mac のバージョン 8.3.0 以降をサポートしています。

Visual Studio for Mac プラグインをインストールして設定し、使用するには：

1. Contrast の配布リポジトリのリリースから、Visual Studio for Mac プラグインのファイル(*.mpack*)をダウンロードします。

2. Visual Studio for Mac で、**Visual Studio > 拡張機能**にアクセスします。
3. **ファイルからインストール...**をクリックして、ダウンロードした*.mpack* ファイルを選択します。プラグインのインストールを行い、Visual Studio for Mac を再起動します。
4. **表示 > その他のウィンドウ > Contrast** を選択します。**設定**アイコンを選択して、**Contrast URL**、**ユーザ名**、**サービスキー**、**API キー**および**組織 ID** を各フィールドに入力します。これらの情報はプロファイル (522ページ)で確認できます。
5. **Test connection** を選択して、Contrast への接続を確認します。接続できたら、**Save** を選択します。
6. プラグインをインストールしたら、**表示 > その他のウィンドウ > Contrast** に戻り、虫眼鏡アイコンを選択して、脆弱性を参照したいアプリケーションを選択します。これにより、選択したアプリケーションに対して Contrast で検出された脆弱性が読み込まれます。脆弱性の一覧が表示されない場合は、**リフレッシュ**アイコンを選択します。脆弱性が深刻度順、ステータスなどの情報と一緒に表示されます。
   脆弱性を選択すると、その概要や詳細情報を確認できます。**General Information** のセクションには、深刻度、アプリケーション、ステータス、履歴などがあります。**Details** セクションには、その脆弱性の詳細や備考、アクティビティなど、Contrast から取得された情報があります。
   選択済フィルターを全てクリアするには、ほうきのアイコンを選択します。この方法で**サーバ**および**アプリケーション**の一覧も表示できます。

管理ガイド

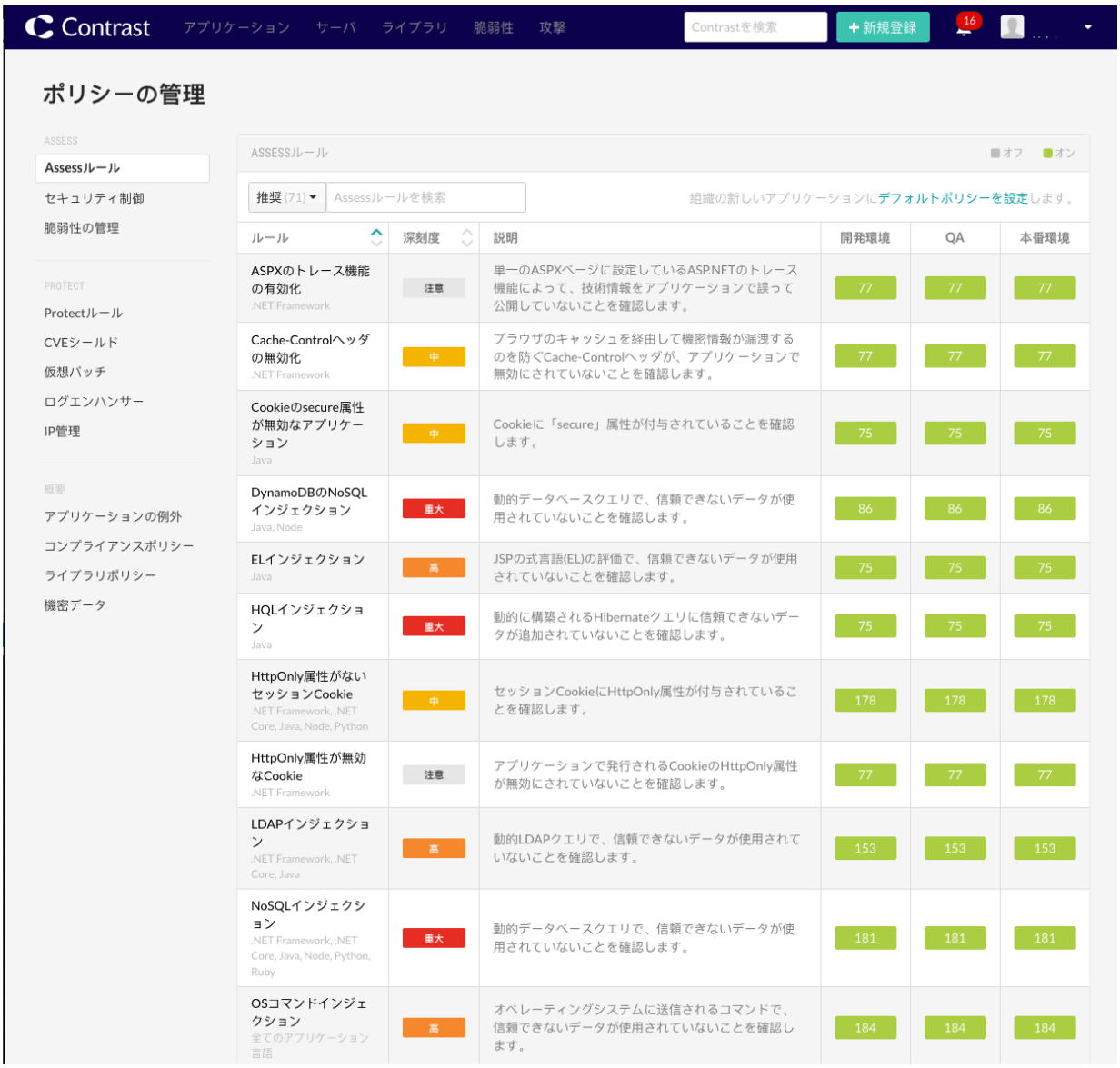
ユーザにはそれぞれ異なるロールや権限 (953ページ)があり、Contrast でのアクセスレベルが異なります。この異なるアクセスレベルにより、企業内の様々なユーザやチームが、それぞれの職務に応じて Contrast を最適に利用することができます。

- ルールとポリシーの管理： (780ページ)RulesAdmin ロール(組織のルール管理者)により、ポリシーの作成と編集ができます。Edit ロール(組織の編集者)では、アプリケーションを追加したり、スコアや通知などのコンテンツ情報を管理できます。
- 組織の管理： (813ページ)Admin ロール(組織管理者)は、特定の組織の設定を構成できます。
- システム管理： (874ページ)オンプレミス版のお客様の場合、SuperAdmin ロール(スーパー管理者)が、Contrast のインストール、設定、管理をシステムレベルで行います。また、ServerAdmin ロール(サーバ管理者)や SystemAdmin ロール(システム管理者)もシステム管理業務をサポートします。

ルールとポリシーの管理

Contrast でアプリケーションをメンテナンスするには、実行する操作に応じて異なるロールや権限が必要です。

RulesAdmin 権限のあるユーザで、**ユーザメニュー**の**ポリシーの管理**を選択します。



管理できる項目は、以下の通りです。

- Assess ルール (782ページ)
- セキュリティ制御 (783ページ)
- 脆弱性ポリシー (786ページ)
- Protect ルールの設定 (793ページ)
- CVE シールド (796ページ)
- 仮想パッチ (801ページ)
- ログエンハンサー (802ページ)
- IP アドレスのブロック/許可 (809ページ)
- ソース名の編集 (810ページ)
- アプリケーションの例外 (804ページ)
- コンプライアンスポリシー (807ページ)
- ライブラリポリシー (810ページ)
- 機密データ (812ページ)

RulesAdmin 権限があるユーザは、次の操作も実行できます。

- エージェントのインストールと設定 (46ページ)

- Protect の有効化 (815ページ)
- 保留中の脆弱性ステータスの変更の承認/拒否 (706ページ)
- 通知の有効化 (813ページ)
- デフォルトの評価方法の設定 (834ページ)

Assess ルールの設定

適用されている全てのルールの一覧を表示するには、 **アプリケーション > アプリケーション名 > ポリシー > ASSESS** を選択するか、ユーザメニューで **ポリシーの管理 > Assess ルール**を選択します。 各ルールには、深刻度と説明があり、そのルールがどの環境に適用されているかも表示されます。

また、組織のデフォルトの Assess ルール (782ページ)も設定できます。

設定を行う前に

- 組織管理者(Admin ロール)またはルール管理者(Rules Admin ロール)であることを確認してください。
- ログインして、該当する組織を選択します。

手順

Assess ルールの適用と更新：

1. アプリケーションの特定の環境に Assess ルールを適用するには：
   a. **アプリケーション**ページでルールの一覧を表示している場合は、トグルボタンを使用して各環境の各ルールをオンまたはオフにします。また、複数のルールに変更を適用するには、左側の列のチェックボックスを使用して複数のルールを選択し、**モードを変更**を選択します。表示される画面で、各環境のルールのオンとオフを切り替え、**保存**を選択します。
   b. または、**ポリシーの管理 > Assess ルール**の場合は、ルールを選択すると、そのルールに関連付けられているアプリケーションの一覧が表示されます。トグルボタンを使用して、アプリケーションごとにルールをオンまたはオフにします。
2. 個々の Assess ルールの設定を更新するには：
   a. **ポリシー管理**にて、ルール名を選択すると、そのルールに関連付けられているアプリケーションの一覧が表示されます。
   b. 1 つまたは複数のアプリケーションを選択するには、各アプリケーションの横にあるチェックボックスをオンにします。
   全てのアプリケーションを選択する場合は、**アプリケーション**のチェックボックスを選択します。
   c. 右上にある**設定**アイコン( )を選択します。
   d. 表示される画面で、このルールの対象となる脆弱性の**可能性**、 **影響度**、**信頼度**を選択します。
   e. 必要に応じて**上書き**チェックボックスをオンにして、設定を保存後にこれらのフィールドを更新するオプションを有効にします。
   f. **リスクの説明**フィールドには、この脆弱性がどんなリスクであるかの追加情報を入力します。また、修正方法として**推奨**する情報を入力することもできます。
   g. **参照**フィールドには、ルールに関する詳細な情報として、特定の脆弱性に関する外部参照へのリンクを入力します。
   h. **保存**を選択します。

組織の Assess ルールの設定

アプリケーションにエージェントを追加· 設定したり、新しい組織を作成すると、デフォルトの Assess ルールセットが適用されます。

組織レベルで Assess ルールのデフォルト設定を変更するには、以下の手順を実行してください。ここでの設定は、Contrast の組織に新規に追加される全てのアプリケーションに適用され、組織内の既存のアプリケーションには影響しません。

設定を行う前に

- 組織管理者(Admin ロール)またはルール管理者(Rules Admin ロール)であることを確認してください。
- ログインして、該当する組織を選択します。

手順

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **デフォルトポリシーを設定**を選択します。

3. フィルターで**全て**を選択します。

4. 各 Assess ルールで、トグルボタンを使用して環境ごとに各ルールをオンまたはオフにします。

セキュリティ制御

セキュリティ制御とは、コード内でデータを安全に使用するためのメソッドです。Contrast で認識される組み込みのセキュリティ制御を通過した情報は信頼できる情報とされます。Contrast は、サードパーティライブラリの多くのメソッドを監視して、データフローが安全かどうかを判断します。

組織によっては、Contrast で認識されない独自のセキュリティ制御の構築が必要な場合があります。その場合は、カスタムメソッドを Contrast に認識させるためにセキュリティ制御を作成します。カスタムのセキュリティ制御を追加すると、Contrast はより正確な結果を報告します。

セキュリティ制御の種類

- **入力バリデーション**：バリデーションは、アプリケーションの他の部分にデータが渡される前に、適切に形成され、フォーマットされたデータのみが入力として受け入れられることを確認するためのメソッドです。入力フィールドで特定の文字を受け入れたり拒否したりできるように定義されます。入力バリデータは、SQLi、XSS、およびその他の入力検証に関連する攻撃を防ぐための主要な方法です。

- **無害化**：入力を検証する代わりに、データベースなどのアプリケーションの他の部分に渡される前に、入力を安全にレンダリングします。例えば、潜在的なインジェクション攻撃で使用される可能性のある一重引用符を取り、それを二重引用符に変更します。

セキュリティ制御を行うタイミング

アプリケーションで使用しているメソッド、クラス、ライブラリを Contrast が正確に把握できない場合にセキュリティ制御を作成し、アプリケーションを入力検証の問題(無害化や入力バリデーション)から守ります。

アプリケーションで使用するバリデーションや無害化が分かっている場合は、それらを手動で追加したり、Contrast が自動的に検出した推奨のセキュリティ制御をアプリケーションに適用するセキュリティ制御として追加することができます。

セキュリティ制御を追加して有効にすると、Contrast Assess で、そのセキュリティ制御で緩和するように定義した入力に関する脆弱性が表示されなくなります。セキュリティ制御を適切な脆弱性ルールに適用することが非常に重要です。

セキュリティ制御の影響

セキュリティ制御は、指定されたルールの脆弱性と検出に影響を与えます。入力バリデーションと無害化は、通常、特定の種類のデータ、特定のフィールドやページ、または特定のアプリケーションのために定義· 適用されます。すべてのルールに対してセキュリティ制御を有効にすると、検知漏れとなる可能性があります。

セキュリティ制御の使用は、慎重に行って下さい。セキュリティ制御の適用は、特定のルールにのみに必要な場合があります。例えば、バリデーションや無害化によって XSS からアプリケーションを保護できても、SQL インジェクションに対しては効果的ではない可能性があります。すべてのルールにセキュリティ制御を適用すると、SQLi の脆弱性が非表示になり、結果として検知漏れとなる可能性があります。

セキュリティ制御は、さまざまな攻撃からアプリケーションを保護することを保証するためのものであるべきです。

セキュリティ制御の管理ロール

RulesAdmin 以上の組織ロールが割り当てられたユーザーのみが、セキュリティ制御を表示または変更できます。

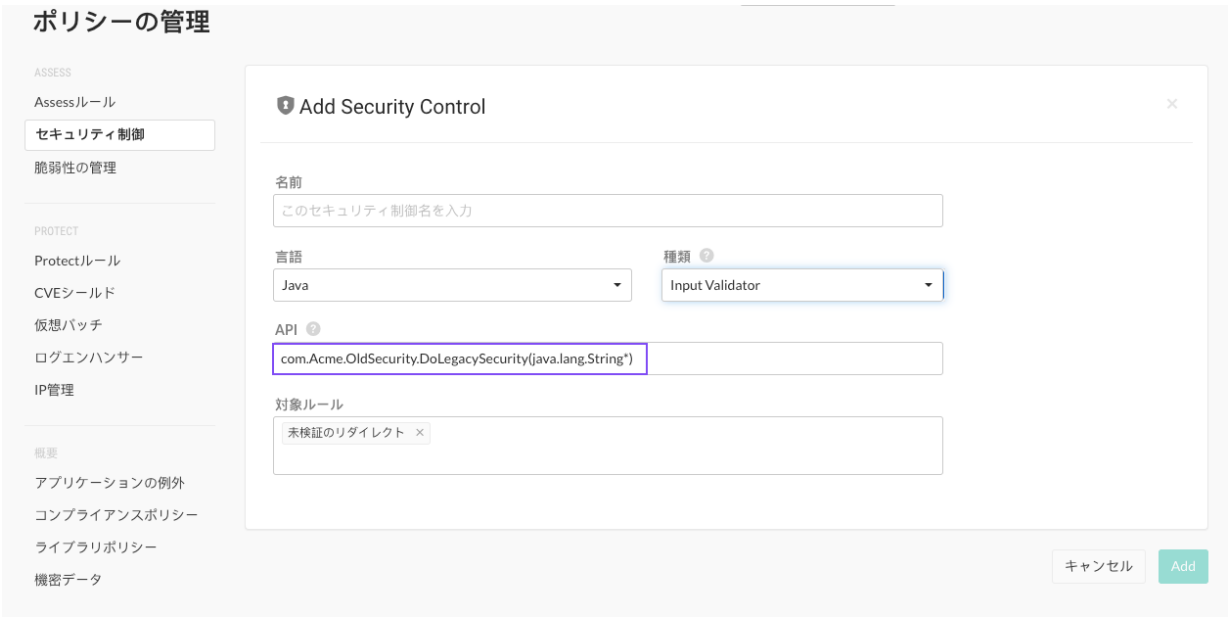
サポートされる言語

セキュリティ制御が適用される言語は、Java、.NET Framework、.NET Core のみです。

セキュリティ制御の例

安全でない暗号化アルゴリズムの使用について報告されている `com.Acme.OldSecurity` というクラス内に `DoLegacySecurity()`というメソッドがあると仮定します。セキュリティ制御を作成し、次のメソッドシグネチャを指定します。

```
com.Acme.OldSecurity.DoLegacySecurity(java.lang.String*)
```



この例では、`Java.lang.String*`がマークされたパラメータで検査対象となります。

セキュリティ制御の作成時には、末尾のパラメータ定義や余分な文字が含まれないように注意します。

Contrast では、検出された脆弱性のスタックトレースとこのメソッドシグネチャが照合され、マッチする脆弱性が非表示になります。

セキュリティ制御の追加、編集、削除

セキュリティ制御が適用される言語は、Java、.NET Framework、.NET Core のみです。

手順

1. **ユーザメニュー > ポリシーの管理**を選択し、**セキュリティ制御**を選択します。
   既にセキュリティ制御がある場合は、セキュリティ制御の一覧に表示されます。
2. 編集する場合は既存のセキュリティ制御の名前を選択します。新規に作成する場合は、**セキュリティ制御を追加**を選択します。
3. 表示される画面で、以下の項目を入力します。
   - **名前**
   - **言語**： Java、.NET Framework、.NET Core から選択します。
   - **種類**：以下のいずれかのメソッドを選択します。
     - **入力バリデーション**(Input Validator)は、ユーザ入力を受け付けて、安全でないデータを受信した場合に修正処理を行います。
     - **無害化**(Sanitizer)は、渡されたデータをクリーンにして、どのインタプリタでも安全に使用できるようにします。無害化によって、ある種類の攻撃を防ぐことができても、他の種類の攻撃が防げるわけではありません。
   - **API**：API を指定する際は、以下の規約を考慮してください。
     - Java の場合には、メソッド名とパラメータを含める必要があります。完全修飾型を使用し、`java.lang.String` パラメータ(boolean、int、long、short double、float などでない)のみを対象とします。
     - **.NET Framework および.NET Core** の場合：
       - 戻り値の型(または void)、メソッド名、パラメータを含めます。完全修飾型を使用し、`System.String` パラメータのみを対象とします。
       - パラメータ間にスペースが無いことを確認してください。
     - バリデーションまたは無害化されるパラメータにアスタリスク(*)を付けます。
   - **対象ルール**：**全て**の脆弱性ルールを選択することも、個別の脆弱性ルールを 1 つ以上選択することもできます。

4. **保存**を選択して新しいセキュリティ制御を作成します。既存のセキュリティ制御を編集する場合は、このパネルから**削除**アイコンを使用してセキュリティ制御を削除することもできます。
5. 表の下部に、Contrast が検出した潜在的なセキュリティ制御の候補が**推奨**として、クラスとメソッドとともに表示されます(このセクションは、ヘッダ行のキャレットをクリックして非表示にできます)。
   セキュリティ制御が初めて自動的に検出された場合は、該当するアプリケーションの View 権限以上を持つ全てのユーザに通知されます。
   API にカーソルを合わせて、この推奨が検出された場所を確認します。また、必要に応じてアプリケーション名を選択し、そのアプリケーションのコンテキストで脆弱性を確認します。

   推奨されるセキュリティ制御の行の端にある**プラス記号**のアイコン(+)を使用して、推奨を新しいセキュリティ制御として追加し、上の表に含めます。追加する前に、名前、API、種類フィールドを直接編集できます。セキュリティ制御の追加後、名前を選択し、セキュリティ制御が正しいアプリケーションルールに適用されることを確認します。

   推奨を削除するには、**ゴミ箱**アイコン(🗑)を使用します。Contrast からの推奨は繰り返されないので、一度削除するとその API が再度推奨されることはありません。過去の推奨を表示したり、戻す方法はありません。

注記
サーバの再起動が必要な場合があります。選択した内容によって、影響を受けるサーバの一覧が表示されます。

特定の脆弱性に対応したセキュリティ制御の作成

タグイベントを使用して、特定の脆弱性のコンテキストでセキュリティ制御を作成することもできます。

Contrast によって脆弱性のランタイムのデータフローがキャプチャされた場合、**脆弱性 > 脆弱性の名前 > 詳細**を選択して、その脆弱性の詳細情報を確認できます。検出された潜在的なセキュリティ制御によってタグイベントがトリガーされ、それが深刻度の低い(緑色の)イベントとして表示されます。イベントを展開すると、**セキュリティ制御の追加**を選択できます。

また、「内部のセキュリティ制御を通過」という理由で脆弱性を**問題無し**とマークした場合は、そのセキュリティ制御をその時点で定義できます。

脆弱性の管理ポリシー

脆弱性ポリシーを使用して、組織の RulesAdmin または Admin ロールのある管理者が、ポリシーのトリガー時に脆弱性のステータスを変更したり、確認のためにフラグを立てる基準を定義することができます。ポリシーを定義する基準には、脆弱性のルール、深刻度、アプリケーション、ルートなどを指定できます。

脆弱性ポリシーによって、どの脆弱性に注意が必要であるか、またどの脆弱性が修復済みでクローズしたと見なされているかをより正確に把握できます。自動検証ポリシーと違反ポリシーを設定できます。

脆弱性がこれらのポリシーに違反する場合、アプリケーション内で通知 (832ページ)するように設定できます。管理者には、Contrast アプリケーション内と E メールの両方で違反が通知されます。

自動検証ポリシーのタイプ

自動検証 (788ページ)ポリシーは、特定の条件を満たす脆弱性のステータスを自動的に**修復済 - 自動検証**に変更します。脆弱性の修正・検証後のステータス変更を手動で行うことに依存するのではなく、注意が必要な脆弱性のステータスをより正確に認識したい場合、自動検証ポリシーは便利です。

自動検証ポリシーのタイプには、セッションベース、ルートベース、または時間ベースがあります。

- **セッションベースの自動検証(推奨)**：エージェントの設定ファイルに指定したメタデータ値の組み合わせによって、セッションを定義します。テストランの最後に Contrast API を呼び出すことで、セッション終了のタイミングを制御します。
  このタイプの自動検証には、自動化されたテストスイートが必要です。
- **ルートベースの自動検証**：エージェントの設定ファイルに指定したメタデータ値の組み合わせによって、セッションを定義します。セッションベースの自動検証を使用できない場合は、この自動検証タイプを使用して下さい。
  このタイプの自動検証には、自動化されたテストスイートが必要です。
- **時間ベースの自動検証**：指定された期間内でアプリケーションの全てのルートを疎通できることが確実な場合は、この自動検証タイプを使用して下さい。
  このタイプの自動検証では、自動テストまたは手動テストを利用できます。

自動検証の動き方

- Contrast で、ある脆弱性が 2 つの異なるセッションに渡って同じルートで検出されない場合、この脆弱性は**修復済 - 自動検証**というステータスになります。2 つのセッションで全く同じセッションメタデータ値が報告された場合、2 つのセッションは 1 つの単独セッションとして見なされます。
  定義した値に応じて、各エージェントの実行を単独のセッションの 1 部にすることもできますし、各エージェントの実行に独自のセッション情報を持たせることもできます。Contrast を CI/CD パイプラインに組み込んでいる場合は、アプリケーションの新しいバージョンをデプロイするたびに、一意であるセッションメタデータのキーと値のペアを少なくとも 1 つ送信して下さい。例えば、コミットハッシュ(commitHash)やビルド番号(buildNumber)、バージョン(version)などのメタデータは、アプリケーションのデプロイごとに更新される場合が多いため、これらの値がエージェントから送信されるように設定すると良いです。
- Contrast で以前に**修復済 - 自動検証**のステータスにされた脆弱性が、同じルートを実行した際に再度検出された場合は、その脆弱性のステータスは**報告済**に変わります。その場合、脆弱性の詳細ページにある「アクティビティ」タブに情報が更新されます。
- 自動検証ポリシーを無効または削除した後に、Contrast で以前に**修復済 - 自動検証**のステータスにされた脆弱性が、同じルートを疎通した際に再度検出された場合は、その脆弱性のステータスは**報告済**に変わります。その場合、脆弱性の詳細ページにある「アクティビティ」タブに情報が更新されます。

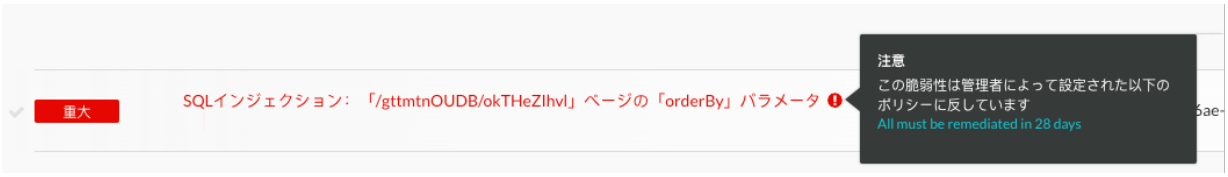
セッションベースおよびルートベースの自動検証のためのセッションメタデータ

セッションベースまたはルートベースの脆弱性ポリシーには、エージェントの設定ファイルにセッションメタデータを追加 (537ページ)して下さい。

- 一意のセッションメタデータを指定することによって、検出結果のベースラインができ、ルート比較に基づいて脆弱性が修復されたかどうかを検証できるようになります。
- テストラン(testRun)のセッションメタデータフィールドを使用すると、テスト実行中にエージェントやアプリケーションを何度も再起動しても、1 つのテスト実行内でのルートと脆弱性を確実に追跡できます。
  Contrast では、一意のメタデータのキーと値のペアごとに、一意のセッション ID が作成されます。この方法でセッションメタデータを使用すれば、複数のテスト実行が 1 つのテストセッションとして統合されます。この操作は、同じルートで異なるコードパスをテストする際などに便利です。
- コミットハッシュ(commitHash)やビルド番号(buildNumber)、バージョン(version)などのメタデータは、アプリケーションのデプロイごとに値が更新される場合が多いため、利用すると便利です。

違反ポリシー

違反ポリシー (791ページ)は、脆弱性が特定の基準に一致した場合に、違反の通知をトリガーします。トリガーされると、脆弱性一覧で脆弱性が赤字で表示されます。脆弱性のフィルタを使用すると、ポリシー違反の脆弱性のみを表示できます。



ポリシーのトリガー

以下のトリガータイプが、脆弱性ポリシーに有効です。

- **セッションベース(推奨)**：セッション中の特定のルートで、脆弱性が検出された場合、または検出されなかった場合に、自動検証ポリシーをトリガーします。このトリガーを使用する場合は、Contrast API を使用してセッションを終了させます。この機能では、テストランの結果をすぐに取得できるよう、セッションの終了タイミングを定義できます。
- **ルートベース**：特定のルートで、脆弱性が検出された場合、または検出されなかった場合に、自動検証ポリシーがトリガーされます。このトリガータイプは、Contrast がルート判定できるサポート対象のテクノロジで利用できます。
- **時間**：指定した日数が経過すると、違反ポリシーまたは自動検証ポリシーがトリガーされます。

環境

最適な結果を得るためには、テストの自動化を行っている環境に脆弱性ポリシーを適用するよう設定します。複数のサーバで同じアプリケーションを実行している場合は、各サーバが開発、QA または本番環境用に設定されていることを確認してください。

複数ポリシーの処理

複数のポリシーが同じ脆弱性に影響を与える場合、以下のルールによって Contrast の処理が決定されます。

- 自動検証ポリシーは、違反ポリシーよりも優先されます。例えば、最初に自動検証の期限が適用された場合、脆弱性はクローズされフラグは立てられません。
- 2 つの時間ベースのトリガーが影響する場合は、期限が最も近い処理が先に適用されます。例えば、最初に違反期限が適用された場合、脆弱性にフラグが立てられ、後の期限が適用された時に脆弱性が自動検証されます。

脆弱性の自動検証ポリシーの設定

脆弱性の自動検証ポリシー (786ページ)の条件を満たす脆弱性のステータスを自動的に**修復済 - 自動検証**に変更します。自動検証ポリシーには、セッションベース、ルートベース、または時間ベースを設定できます。

ポリシーを追加すると、デフォルトでポリシーが有効になります。ポリシーの有効/無効は、自動検証タブの「有効」の列で切り替えることができます。

開始する前に

- Contrast のバージョン 3.7.2 以降を使用してください。
- Contrast エージェントでサポート対象となる最小バージョン以降を使用していることを確認してください。
  - .NET Framework 20.4.1
  - .NET Core 1.0
  - Java 3.7.3.14895
  - Node.js 2.11.0
  - Python 3.4.0
  - Ruby 3.8.4
- サポート対象のフレームワーク (791ページ)を使用していることを確認してください。
- セッションベースまたはルートベースの自動検証を使用する場合は、エージェントの設定ファイルに一意のセッションメタデータを設定 (537ページ)して下さい(例、コミットハッシュ、ビルド番号、バージョンなど)。

自動検証ポリシーの設定

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **脆弱性の管理**を選択します。
3. 脆弱性ポリシーにある**自動検証**タブを選択します。
4. **ポリシーを追加**を選択します。
5. **名前**に、ポリシーの名前を入力します。
6. **脆弱性ルール**に、ポリシーに関連付ける 1 つ以上の深刻度や Assess ルールを選択します。
7. **アプリケーション**に、ポリシーに関連付ける 1 つ以上の重要性やアプリケーションを選択します。具体的な重要性やアプリケーションを検索する場合、「アプリケーション」欄に値を入力してみて下さい。
8. **環境**には、ポリシーを適用する 1 つ以上のサーバ環境を選択します(全ての環境、開発環境、QA、本番環境)。
9. **トリガー**では、ポリシーに使用するトリガーの種類を選択します(トリガーのタイプを 1 つまたは両方選択)。
   - 時間ベースのトリガーを設定するには、**指定日数以降に全ての脆弱性を自動検証によって修復済のステータスにする**を選択し、脆弱性ポリシーが自動検証されたステータスになるまでの日数を選択します。
     このトリガーは、選択した期間内に脆弱性を修正し、全てのルートを疎通できることが確実な場合に便利です。Contrast で再びその脆弱性が確認された場合、その脆弱性は再オープンされます。

     ヒント
     時間ベースの自動検証を、セッションベースまたはルートベースの自動検証と併用することで、ビルドごとにルートが大きく変化する状況を検知できます。例:

     - 新しいルートを追加して、古いルートを削除するなど、コードの大幅なリファクタリングを行った場合。
     - ルートが有効ではなくなり、疎通されなくなった場合。

   - セッションベースまたはルートベースの自動検証を設定するには :
     a. **ルートまたはセッションに基づく自動検証**を選択します。
        セッションまたはルートベースの自動検証は、時間ベースのトリガーよりも優先されます。
     b. 以下のオプションのいずれか**ひとつ**を選択します(両方のオプションを同時に使用することはできません)。
        - **セッションベースの自動検証(推奨)**: このタイプの自動検証ポリシーでは、セッションの終了タイミングを定義でき、ビルドの合否も判断できるなど、テストランの結果をすぐに

取得できます。自動検証として推奨されるのは、セッションベースの自動検証です。このオプションを選択する場合、テストランの最後に Contrast API を呼び出すことで、セッションを終了します。

- **ルートベースの自動検証**：セッションベースの自動検証を使用できない場合に、ルートベースの自動検証の使用を検討します。この場合、エージェントに設定したセッションメタデータによって、セッションが定義されます。
  ルートベースのトリガーは、ルートが識別できる特定のテクノロジに対してのみ機能します。

10. **保存**を選択します。

セッションベースの自動検証のためのテストランの設定

セッションベースの自動検証は、自動検証の方法として推奨されますが、テストランの最後に Contrast SBAVRouteSession API を呼び出す必要があります。以下に、セッションをクローズする例をいくつか示します。

認証ヘッダーと API キーを検索するには、Contrast Web インターフェイスにログインし、ユーザメニューより**ユーザの設定**を選択します。

- セッション ID とアプリケーション ID でセッションを終了する場合。以下の例のようなコマンドを使用して下さい。

```
curl --location --request POST 'https://<HOST>/Contrast/api/ng/
organizations/<ORG-UUID>/agent-sessions/sbav' \
--header 'Authorization: <Your-Auth-Header-Value>' \
--header 'API-Key: <API-KEY>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "sessionId":"0",
    "appId":"5b4960b3-a111-4f2a-bf24-7367be7c8302"
}'
```

- セッション ID、アプリケーション名、アプリケーション言語でセッションを終了する場合。以下の例のようなコマンドを使用して下さい。

```
curl --location --request POST 'https://<HOST>/Contrast/api/ng/
organizations/<ORG-UUID>/agent-sessions/sbav' \
--header 'Authorization: <Your-Auth-Header-Value>' \
--header 'API-Key: <API-KEY>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "sessionId":"0",
    "appName":"FakeRubyApp",
    "appLanguage":"JAVA"
}'
```

- アプリケーションに設定されているメタデータのキーと値のペア、およびアプリケーション ID でセッションを終了する場合。以下の例のようなコマンドを使用して下さい。

```
curl --location --request POST 'https://<HOST>/Contrast/api/ng/
organizations/<ORG-UUID>/agent-sessions/sbav' \
--header 'Authorization: <Your-Auth-Header-Value>' \
--header 'API-Key: <API-KEY>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "appId":"abc",
    "metadata":[
        {"label":"developer", "value":"carlos"},
        {"label":"repo", "value":"ts"}
```

```
    ]
}'
```

- アプリケーションに設定されているメタデータのキーと値のペア、およびアプリケーション名とアプリケーション言語でセッションを終了する場合。以下の例のようなコマンドを使用して下さい。

```
curl --location --request POST 'https://<HOST>/Contrast/api/ng/
organizations/<ORG-UUID>/agent-sessions/sbav' \
--header 'Authorization: <Your-Auth-Header-Value>' \
--header 'API-Key: <API-KEY>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "appName":"FakeJavaApp",
    "appLanguage":"JAVA",
    "metadata":[
        {"label":"developer", "value":"carlos"},
        {"label":"repo", "value":"ts"}
    ]
}'
```

脆弱性ポリシーの更新

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **脆弱性の管理**を選択します。
3. 脆弱性ポリシーにある**自動検証**タブを選択します。
4. 自動検証タブで、ポリシーを選択します。
5. 必要に応じて、値を更新します。
6. **更新**を選択します。

自動検証のサポート対象フレームワーク

次のフレームワークが、自動検証ポリシーに対応しています。

- **Java：**Jersey 2、Spring MVC 4、Struts 1、Struts 2、Servlet(ベータ)
- **.NET Framework：** ASP.NET MVC(バージョン 4 と 5)、WebForms、WebAPI、WCF
- **.NET Core：**ASP.NET Core MVC(バージョン 2.1、2.2、3.0、3.1)、ASP.NET Core Razor Pages(バージョン 2.1、2.2、3.0、3.1)
- **Node.js：** Express、Hapi 17 以降、Koa、Kraken
- **Python:** Django、Pyramid、Flask
- **Ruby：**Rails、Sinatra

脆弱性の違反ポリシーの設定

違反ポリシーは、脆弱性がポリシーに違反していることをマークするものです。このポリシーが適用されると、脆弱性の一覧で違反が赤字で表示されます。

ポリシーを追加すると、デフォルトでポリシーが有効になります。ポリシーの有効/無効は、違反タブの「有効」の列で切り替えることができます。

開始する前に

- 組織の Rules Admin ロールか組織の Admin ロールが必要です。

手順

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **脆弱性の管理**を選択します。
3. **違反**タブを選択します。
4. ポリシーを追加するには：
   a. **ポリシーを追加**を選択します。
   b. 名前にポリシーの名前を入力します。
   c. 脆弱性のルールで、ポリシーに関連付ける深刻度や Assess ルールを選択します。
   d. アプリケーションで、ポリシーに関連付ける重要性やアプリケーションを選択します。
   e. 環境で、ポリシーを適用するアプリケーションがホストされているサーバ環境を選択します。
   f. トリガーでは、**指定日数以降に既存の脆弱性にフラグを立てる**を選択し、日数を選択します。
   g. **保存**を選択します。
5. ポリシーを更新するには：
   a. 違反タブで、ポリシーを選択します。
   b. 必要に応じて、値を更新します。
   c. **更新**を選択します。

Protect ルール

Protect ルールを適用することで､アプリケーション環境における特定の種類のサイバー攻撃を監視またはブロックします。各ルールは、SQL インジェクションやクロスサイトスクリプティングなど、カスタムコードやオープンソースライブラリの脆弱性を悪用する攻撃の種類を表しています。

Contrast には多数の Protect ルールがあり、以下のような攻撃を監視したりブロックするために使用できます。

- **コマンドインジェクション**： 巧妙に作成された入力により、オペレーティングシステムレベルで不正なコマンドを実行されてしまう脆弱性。
- **クロスサイトスクリプティング**： ユーザが他のユーザのブラウザで任意の JavaScript を実行できてしまう Web アプリケーションの脆弱性。
- **EL インジェクション**：多くのフレームワーク やカスタムコードに存在する脆弱性で、アプリケーションがユーザ入力を OGNL、SpEL、JSP EL などの式言語(EL)として誤って評価した時に発生。
- **メソッドの改ざん**：セキュリティ設定で、暗黙的に「全てを許可する」ような設定がある認証･ 承認システムに対する攻撃のこと。
- **パストラバーサル/ローカルファイルインクルード**： アプリケーションで開いて読み込むファイルをユーザが制御できてしまう脆弱性。

- **SQL インジェクション· NoSQL インジェクション**：巧妙に作成された入力により、アプリケーションで使用される SQL クエリや NoSQL クエリが変更され、データが盗まれたりコードが実行されてしまう脆弱性。
- **安全でないファイルのアップロード**：悪意のあるファイルがアップロードの保護をバイパスして、悪意のある処理を実行できてしまう、アップロード処理の脆弱性。このルールは、一般的に使用される次のような拡張子を持つファイルに適用されます(ただし、これらに限定されるものではありません): SVG、ASP、ASPX、*SH、JAR、JAVA など。このルールによって、監視モードの場合には、安全でない可能性のあるファイルのアップロードが報告されます。ブロックモードの場合は、これらのファイルはブロックされます。
- **信頼できないデータのデシリアライゼーション**：ユーザが任意のオブジェクトをデシリアライズ処理に渡し、リモートからのコード実行が可能になる、Web アプリケーションの脆弱性。
- **XML 外部実体参照処理(XXE)**：ユーザがファイルに対してファイルを読み書きしたり、リモートコードを実行できる可能性がある、XML 処理の脆弱性。

Protect ルールの設定

アプリケーション環境で攻撃を監視· ブロックする Protect ルール (792ページ)を設定することができます。

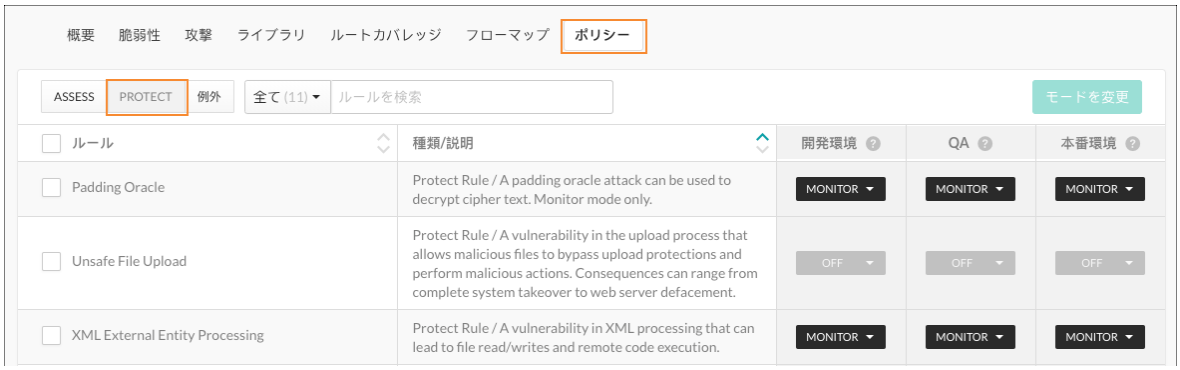
新しいアプリケーションを追加すると、Contrast はデフォルトの Protect ルールを適用します。組織のデフォルトの Protect ルール (795ページ)のモードは変更できます。

設定を行う前に

- Contrast Security(SaaS 版のお客様の場合)、もしくは SuperAdmin(オンプレミス版のお客様の場合)によって、組織に Protect 権限が付与 (915ページ)されていることを確認してください。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
2. アプリケーション名を選択し、**ポリシー**タブを選択します。
3. **Protect** を選択します。



4. 特定のルールを検索するには、検索ボックスにルール名を入力します。
5. ルールごとに、各環境のモードを設定します。
   a. 各環境のドロップダウンを選択します。
   b. 次のいずれかのモードを選択します。
      - **オフ**：ルールは完全に無効になります。
      - **監視**：エージェントは攻撃を特定し、報告します。
      - **ブロック**：エージェントは攻撃を特定して報告し、ブロックします。

重要
攻撃がルールと一致し、そのルールが**ブロック**モードに設定されている場合、エージェント(Java、.NET Framework、.NET Core)は `AttackBlockedException` をスローします。

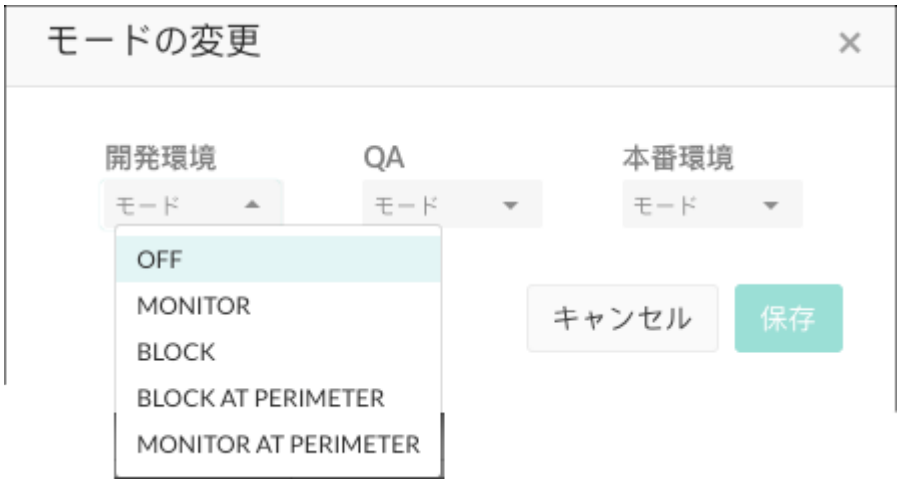
アプリケーションがクラッシュしないように、`AttackBlockedException` を処理するようにアプリケーションを設定してください。

- **ペリメータでブロック**：エージェントは、アプリケーションが攻撃を処理する前に、攻撃の可能性をブロックします。このオプションは全てのルールで利用できるわけではありません。
- **ペリメータで監視**：エージェントは、アプリケーションが攻撃を処理する前に、攻撃の可能性の特定と報告を試みます。このオプションは全てのルールで利用できるわけではありません。
  ペリメータでブロックまたは監視をする場合、エージェントはシンクで攻撃を検証しません。この処理は、誤検知の結果となる可能性があります。

ヒント
環境ごとに Protect ルールを異なるモードに設定することで、ポリシーをテストできます。これにより、本番前にさまざまなオプションがどのように機能するかを確認でき、本番環境での防御を妨げることはありません。

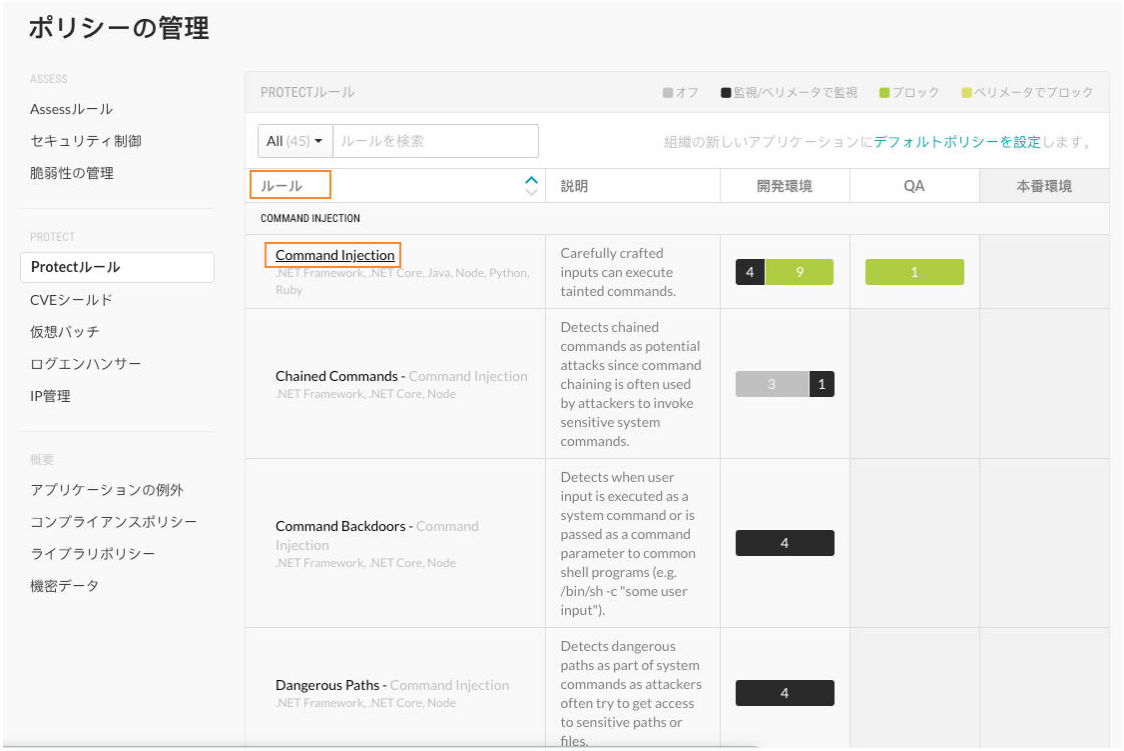
6. 複数のルールに設定を適用するには、次のいずれかの方法を使用します。
   a. 変更したい各ルールの横にあるチェックボックスを選択し、**モードを変更**を選択します。
   b. 全てのルールの設定を変更するには、**ルール**の横にあるチェックボックスを選択し、**モードを変更**を選択します。

   c. 「モードの変更」画面で、各環境のモードを選択し、**保存**を選択します。

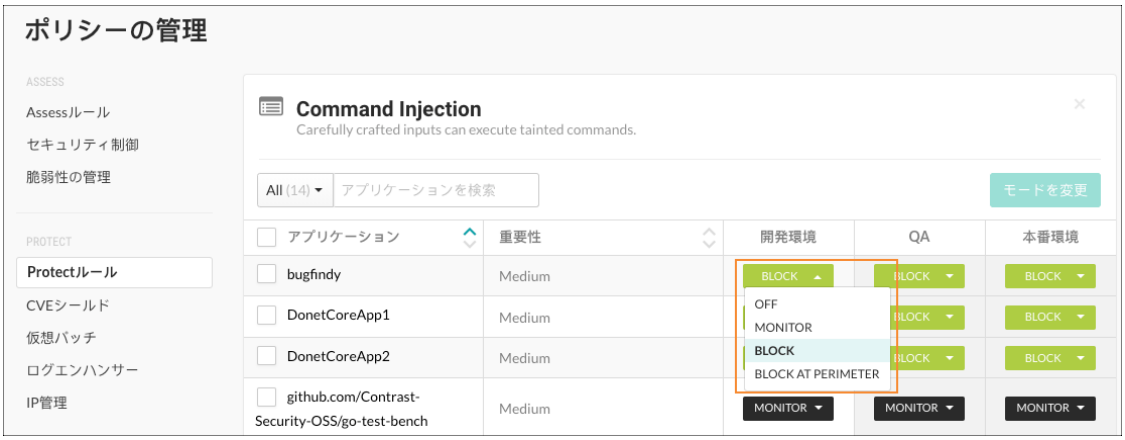


7. 特定のルールを使用する組織内の全てのアプリケーションに Protect ルールを設定するには：
   この手順には、組織ロールとしてルール管理者(Rules Admin)ロールが必要です。
   a. **ユーザメニュー > ポリシーの管理 > Protect ルール**を選択します。
   b. ルールの一覧にフィルタを適用するには、ドロップダウンを使用して言語でフィルタをかけるか、検索フィールドを使用して名前でルールを検索します。

c. ルール名を選択すると、現在そのルールを使用している全てのアプリケーションの設定を管理できます。



d. ドロップダウンを使用して、各環境の Protect モードを設定します。



組織の Protect ルールの設定

アプリケーションにエージェントを追加· 設定したり、新しい組織を作成すると、デフォルトの Protect ルールセットが適用されます。

注記

2021 年 8 月から、新しい組織には最適化された Protect ルールセットが含まれます。この構成は、パフォーマンスの向上など、Contrast をご利用の皆様に最高の価値を提供するためのものです。

組織レベルで Protect ルールのデフォルト設定を変更するには、以下の手順を実行してください。 ここでの設定の変更は、Contrast の組織に追加する新しいアプリケーションに適用されます。組織内の既存のアプリケーションには影響しません。

設定を行う前に

- 組織管理者(Admin ロール)またはルール管理者(Rules Admin ロール)であることを確認してください。
- ログインして、該当する組織を選択します。

手順

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **Protect ルール**を選択します。
3. **デフォルトポリシーを設定**を選択します。

4. Protect ルールごとに、アプリケーションがホストされている環境(開発環境、QA、本番環境)のドロップダウンを選択します。
5. 次のいずれかのモードを選択します。
   - **オフ**：ルールは完全に無効になります。
   - **監視**：エージェントは攻撃を特定し、報告します。
   - **ブロック**：エージェントは攻撃を特定して報告し、ブロックします。
   - **ペリメータでブロック**： エージェントは、アプリケーションが攻撃を処理する前に、攻撃の可能性をブロックします。このオプションは全てのルールで利用できるわけではありません。
   - **ペリメータで監視**：エージェントは、アプリケーションが攻撃を処理する前に、攻撃の可能性の特定と報告を試みます。このオプションは全てのルールで利用できるわけではありません。

CVE シールド

共通脆弱性識別子(CVE)は、特定の脆弱性やリスクを標準化した識別子を表します。CVE は、ツールのカバレッジを評価するためのベースラインにもなります。

Contrast では、CVE があるアプリケーションを保護するために役立つ CVE シールドをいくつか提供しています。CVE シールドは、アップデートするのが難しい脆弱なライブラリを使用しているレガシーアプリケーションに有効です。

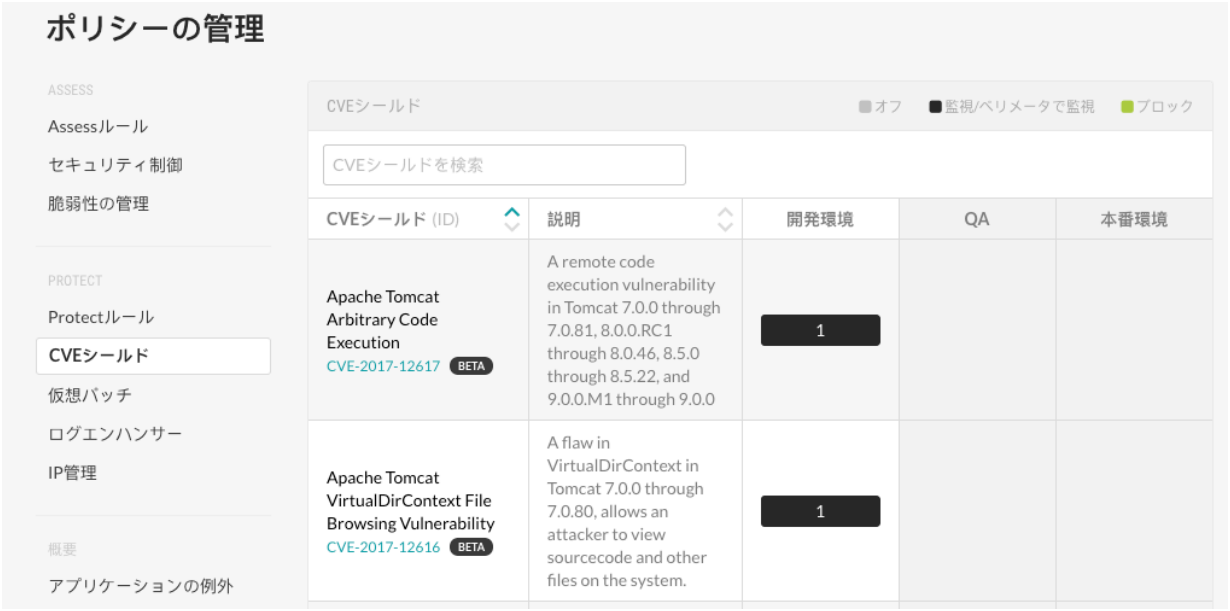
CVE シールドが必要となるのは、脆弱性が SQL インジェクションや信頼されていないデシリアライゼーションのような一般的な攻撃でない場合のみです。攻撃の発生を防ぐ既存の Protect ルールがある場合でも、特定の脅威に特化したより多くのデータを得るために Contrast で CVE シールドを作成することもあります。これにより、攻撃の状況をより詳しく把握しやすくなり、進行中の攻撃をセキュリティエコシステム全体の傾向にマッピングするのにも役立ちます。

CVE シールドを表示する (796ページ)

CVE シールドのモードを設定する (798ページ)

CVE シールドを表示

CVE シールドの一覧には、次の情報が表示されます。

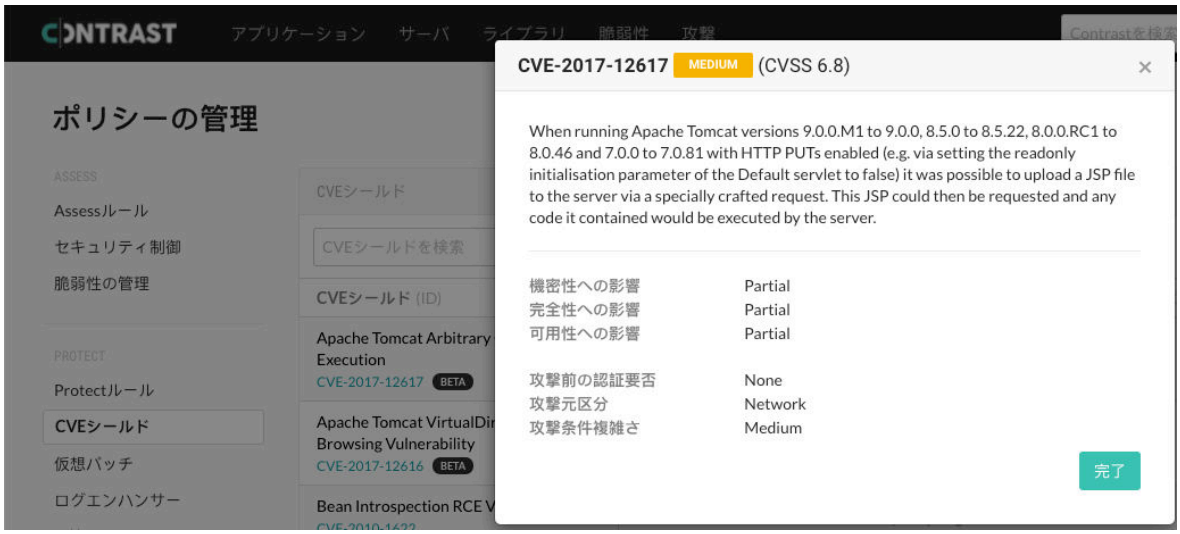


- Contrast が特定の CVE に対して提供する CVE シールド
- CVE の説明
- アプリケーションをホストするサーバが稼働している環境
- CVE シールドに設定されているモード：
  - **オフ**：このモードでは、CVE シールドが完全に無効になります。
  - **監視**：このモードでは、CVE シールドが攻撃を特定し報告します。
  - **ペリメータで監視**：このモードでは、アプリケーションが攻撃を処理する前に、CVE シールドが攻撃の可能性の特定と報告を試みます。このオプションは、全ての CVE シールドで利用できるわけではありません。
  - **ブロック**：このモードでは、CVE シールドが攻撃を特定して報告し、ブロックします。
- 特定の CVE を含むアプリケーション(存在する場合)
  CVE シールドはこの脆弱性を攻撃から守ります。

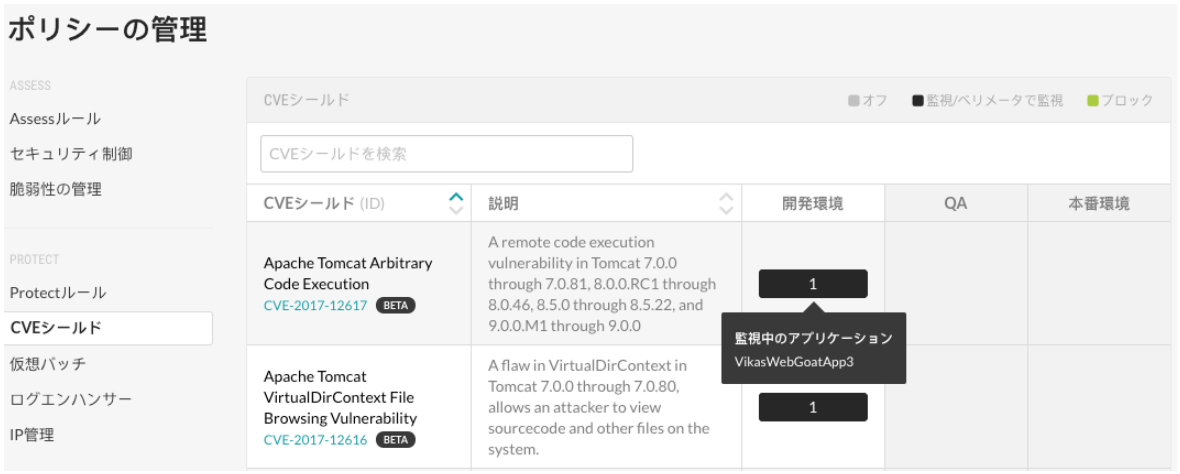
手順

CVE シールドを表示するには：

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. 左ペインの Protect より、**CVE シールド**を選択します。
3. 特定の CVE を検索するには、検索ボックスに CVE シールド名の全部または一部分を入力します。
4. 特定の CVE の詳細を参照するには、CVE 名の下にあるリンクをクリックします。



5. CVE があるアプリケーションを確認するには、サーバ環境の列の 1 つで、数字の上にカーソルを合わせます。ツールチップには、CVE シールドが防御しているアプリケーションの一覧が表示されます。
   数字は、その CVE を含むアプリケーションの数を示します。 モードは、CVE シールドがどのモードで設定されているかを示します。



CVE シールドのモードを設定

CVE シールドは、攻撃のカテゴリを検知するのではなく、アプリケーションに含まれる特定の CVE を攻撃から守るものです。

開発環境、QA 環境または本番環境で実行中のサーバでホストされているアプリケーションに対して、以下のいずれかのモードを設定します。

- **オフ**：このモードでは、CVE シールドが完全に無効になります。
- **監視**：このモードでは、CVE シールドが攻撃を特定し報告します。
- **ベリメータで監視**：このモードでは、アプリケーションが攻撃を処理する前に、CVE シールドが攻撃の可能性の特定と報告を試みます。このオプションは、全ての CVE シールドで利用できるわけではありません。
- **ブロック**：このモードでは、CVE シールドが攻撃を特定して報告し、ブロックします。

設定を行う前に

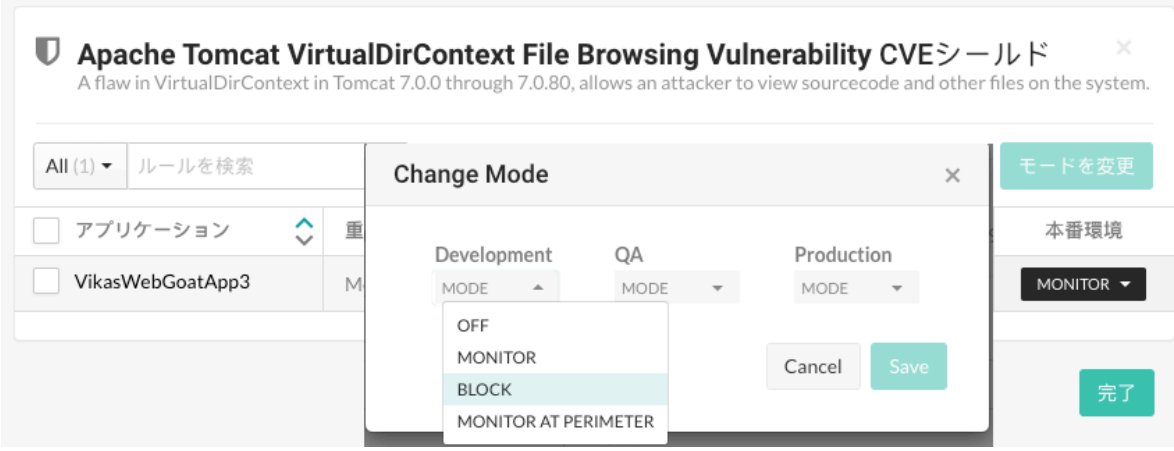
- **必須**：組織またはルールの管理者権限(Admin もしくは Rules Admin ロール)があることを確認してください。

- サーバの設定 (593ページ)を確認し、アプリケーションをホストするサーバの環境が正しく設定されていることを確認してください。

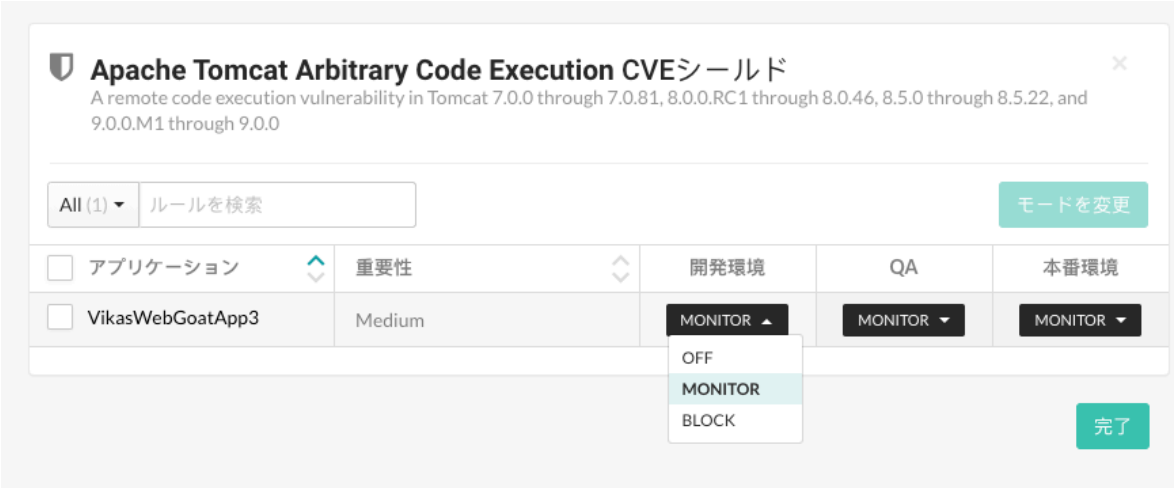
手順

CVE シールドのモードを設定するには：

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **CVE シールド**を選択したら、CVE シールド名をクリックします。
   特定の CVE を検索するには、検索ボックスに CVE シールド名の全文または一部分を入力します。
3. 全てまたは複数のアプリケーションのモードを設定するには：



   a. 全てのアプリケーションを選択する場合は、**アプリケーション**のチェックボックスを選択します。複数のアプリケーションを選択する場合は、各アプリケーションのチェックボックスを選択します。
   b. **モードを変更**をクリックします。
   c. 「モードの変更」画面で、選択したアプリケーションの CVE シールドのモードを 1 つまたは複数の環境で選択したら、保存します。
   d. **完了**をクリックします。
4. 1 つのアプリケーションのモードを設定するには：



   a. アプリケーションの行の最後にある、環境の列でメニューを選択します。
   アプリケーションをホストするサーバに環境が定義されていない場合、その環境にカーソルを合わせると、ツールチップが表示されます。その環境にサーバを設定するには、**設定する**をクリックして、サーバの行の最後にある設定アイコン( )を選択します。

b. 選択した環境で、アプリケーションの CVE シールドのモードを選択します。
c. **完了**をクリックします。

組織の CVE シールドのモードを設定

Contrast の組織内のアプリケーションにエージェントを追加して設定すると、デフォルトの CVE シールドセットが適用されます。

注記

2021 年 8 月から、新しい組織には最適化された CVE シールドセットが含まれます。この構成は、パフォーマンスの向上など、Contrast をご利用の皆様に最高の価値を提供するためのものです。

組織レベルで CVE シールドのデフォルト設定を変更するには、以下の手順を実行してください。ここでの設定は、Contrast の組織に新規に追加される全てのアプリケーションに適用され、組織内の既存のアプリケーションには影響しません。

設定を行う前に

- 組織管理者(Admin ロール)またはルール管理者(Rules Admin ロール)であることを確認してください。
- 正しい組織にログインするか、選択してください。

手順

CVE シールドのモードを変更するには：

1. ユーザメニューから、**ポリシーの管理**を選択します。
2. **CVE シールド**を選択します。
3. **デフォルトポリシーを設定**を選択します。

4. CVE シールドごとに、アプリケーションがホストされている環境(開発環境、QA、本番環境)のドロップダウンを選択します。
5. 次のいずれかのモードを選択します。
   - **オフ**：このモードでは、CVE シールドが完全に無効になります。
   - **監視**：このモードでは、CVE シールドが攻撃を特定し報告します。
   - **ペリメータで監視**：このモードでは、アプリケーションが攻撃を処理する前に、CVE シールドが攻撃の可能性の特定と報告を試みます。このオプションは、全ての CVE シールドで利用できるわけではありません。

- **ブロック**：このモードでは、CVE シールドが攻撃を特定して報告し、ブロックします。

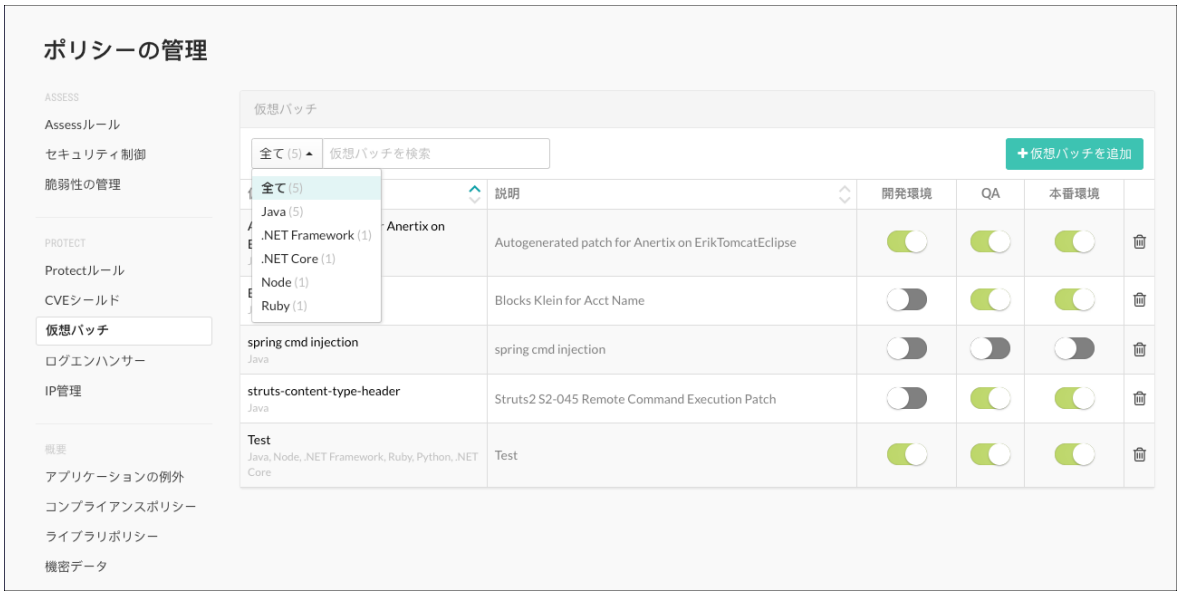
仮想パッチの管理

仮想パッチとは、暫定的なカスタムルールで、特定の条件(URL、パラメータのキーや値など)を満たす HTTP リクエストをアプリケーションが処理する前にブロックできます。

組織の管理者(Admin)とルール管理者(RulesAdmin)は、仮想パッチの表示と管理ができます。

仮想パッチを追加するには：

1. **ユーザメニュー**で、**ポリシーの管理**より**仮想パッチ**を選択します。
2. 言語フィルターや一覧表の上にある検索フィールドを使用して、仮想パッチを検索します。



3. ルールの設定を編集するには、仮想パッチ名をクリックします。または、新規に作成するには、**仮想パッチを追加**を選択します。
   削除アイコンを選択してルールを削除したり、一覧表にあるトグルボタンを使用して各環境を有効/無効にすることもできます。
4. 表示される画面で、**名前**と**説明**を入力します。

5. **適用対象**の欄では、ラジオボタンを使用して、特定の**アプリケーション**、**アプリケーション言語**、または**アプリケーションテクノロジー**のいずれにルールを適用するか選択します。該当のボタンをクリックしたら、複数選択が可能なフィールドが表示されるので、選択をさらに絞り込むことができます。
6. **条件**では、ドロップダウンメニューを使用して、パッチがアプリケーションに適用される条件を選択します。必要であれば、別の行に**条件を追加**します。
   仮想パッチの値の適用方法を指定する場合は、次のいずれかを選択します。
   - 「が次と等しい」
   - 「に次が含まれる」
   - 「が次と一致する」(Perl 互換正規表現 - PCRE を使用)
   - 「が次と等しくない」
   - 「に次が含まれない」
   - 「が次と一致しない」(Perl 互換正規表現 - PCRE を使用)

   「**が次と一致する**」オプションと「**が次と一致しない**」オプションはどちらも Perl 互換正規表現 (PCRE)の使用をサポートしています。「**が次と一致する**」オプションか「**が次と一致しない**」オプションを選択した場合、HTTP リクエストの選択したフィールドの値に対して一致する正規表現を定義できます。
   式が一致する場合、または定義通りに一致しない場合、仮想パッチが適用されて緩和策が実行されます。

   > **注記**
   > 正規表現はとても効果的ですが、複雑で正しく定義するのが難しい場合もあります。PCRE 形式がよくわからない場合は、セキュリティ専門家または Contrast Security にサポートを依頼して、仮想パッチが正しく効果的に設定されるように確認して下さい。
   >
   > 手始めに、正規表現リファレンス (806ページ)を参照することもできます。このリファレンスには、PCRE 形式で可能な表現の例がいくつかあります。

7. **追加**を選択して、設定を保存します。

ログエンハンサーの追加や編集

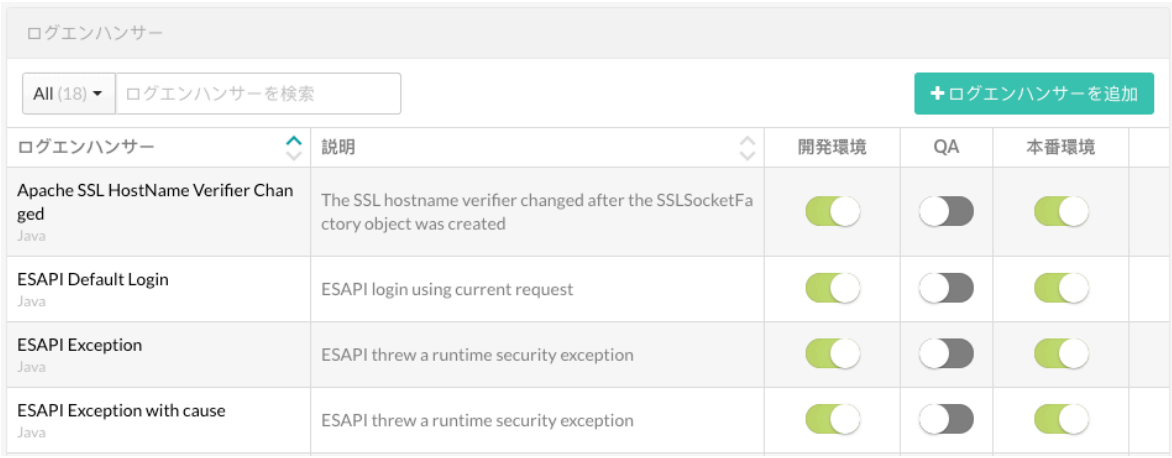
ログエンハンサーとは、ソースコードの変更を必要とせずに、Contrast エージェントがアプリケーションのパラメータやデータを追加でログに記録するためのエージェント型手法の指示となるものです。

この高度なエージェント型手法を使用することで、ユーザはログに記録する API やパラメータを指定でき、Contrast エージェントは RASP ログの一部として *security.log* ファイルにこの情報を追加します。

> **注記**
> 2021 年 8 月から、新しい組織には最適化されたログエンハンサーセットが含まれます。この構成は、パフォーマンスの向上など、Contrast をご利用の皆様に最高の価値を提供するためのものです。

ログエンハンサーを追加、編集、削除するには：

1. ポリシーの管理 (780ページ)で、**ログエンハンサー**を選択します。
2. 言語でフィルタをかけるか検索を使用して、編集する既存のログエンハンサーを検索して名前を選択します。もしくは、**ログエンハンサーの追加**を選択します。各環境でルールを有効または無効にするには、一覧表の行にあるトグルを使用します。



3. 表示される画面で、**名前**と**説明**を入力します。

4. **ログレベル (960ページ)**と**ログタイプ**を入力します。
5. **ログを記録する API** では、以下を入力します。
   - **言語**
   - **API**：`<class_name>.<method_name>(<argument_types>)`の構文を使用します。例：

     ```
     public boolean com.acme.Authenticator.authenticate(String user, \
     String password)
     ```

   - **フォーマット**：関数呼び出しからの関連データを含めて、ログの説明を入力します。次のプレースホルダーをメッセージに含めることができます。
     - `{{O}}`：この呼び出しが行われたオブジェクトの文字列表記を出力します。メソッドが static の場合、これは NULL か空となる場合があります。
     - `{{Pn}}`：インデックス n で指定されたパラメータを出力します。n は、1 から始まることに注意してください。
     - `{{P1}}`：最初のパラメータをメッセージに出力します。
     - `{{R}}`：関数の戻り値を出力します。
6. **追加**を選択して、ログエンハンサーを保存します。

アプリケーションの例外

何らかの理由により報告が不要なイベントを制御するために、例外を使用できます。Contrast エージェントの検査範囲外で外部のセキュリティ制御を使用している場合は、イベントの制御が必要になる場合があります。例：

- 管理者としては、クロスサイトスクリプティング(XSS)の脆弱性に該当するとしても、Web ページに表示される HTML の変更が必要な場合があります。この場合、例外を作成して、このような変更が報告されないようにすることができます。
- エッジデバイスを使用して、アウトバウンド HTTP のレスポンスに正しいヘッダを含めることで、クリックジャッキング攻撃を防いでいるとします。ただし、アプリケーションには必要な防御がされていないために、この問題が適切に報告される可能性があります。
- ベータ版のルールをテストする際などに、例外を使用することで誤検知を抑えることができます。

Java、Node.js、.NET、Python、Go、Ruby エージェントを使用している場合は、ポリシーの管理から、または攻撃イベントの一覧からアプリケーションの例外を追加 (804ページ)できます。

定義済みの例外の一覧を表示するには、**アプリケーション > アプリケーション名 > ポリシー > 例外**、または**ユーザメニュー > ポリシーの管理 > アプリケーションの例外**を選択します。

特定のアプリケーションに例外を追加するには、**アプリケーション > *アプリケーション名*> ポリシー > 例外**または**攻撃 > 攻撃イベント**にアクセスしてください。

アプリケーションの例外の追加

Java、.NET Framework、.NET Core、Node.js、Python、Go、Ruby エージェントを使用している場合、アプリケーションの例外 (804ページ)を使用して、特定のアプリケーションやアプリケーションの一部をセキュリティ検査の対象から外すことができます。

現在、PHP エージェントは、アプリケーションの例外をサポートしていません。

注記

メッセージキューの例外は、現在一般的に利用可能なわけではありません。メッセージキューの例外を利用するには、Contrast サポートまでご連絡ください。

開始する前に

- コードの例外は、Java と.NET エージェントでサポートされます。
- 入力の例外は、Java、.NET、Node.js、Python、Ruby、Go エージェントでサポートされます。
- URL の例外は、Java、.NET、Node.js、Python、Ruby エージェントでサポートされます。
- キュー/トピック(メッセージキュー)の例外は、Java と.NET Core エージェントでサポートされます。

手順

1. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択して、アプリケーションの一覧よりアプリケーション名をクリックします。
   例外を適用できるのは、その例外を作成したアプリケーションだけです。
2. **ポリシー**タブを選択したら、**例外**を選択します。
3. **例外を追加**を選択します。

ヒント
既存の攻撃イベントから例外を作成することもできます。攻撃イベントの一覧を表示する場合は、**攻撃 > 攻撃イベント**を選択します。一番右の列にある三角形を選択して、**例外を追加**を選択します。このボタンを選択すると、そのイベントの情報に基づいて例外の各フィールドにデータが事前に入力されます。

作成後、この例外は例外の一覧に表示されます。

4. 「例外を追加」の画面で、この例外の**名前**を入力します(覚えやすい名前にします)。
5. **例外の種類**を選択します。
入力、URL、キュー/トピックベースによる例外定義には、以下の値を含む Perl 互換正規表現(PCRE)のサブセットを使用できます。

```
.* for 0 or more of any character
.+ for 1 or more of any character
.? for 0 or 1 of any character
. for 1 of any character
\. for an escaped literal of . for usage Examle: somefile\.jsp
```

詳細は、こちらの正規表現の例 (806ページ)を参考にしてください。
以下のいずれかの種類を選択してください。

- **コード**：例外にするメソッドシグネチャを指定します。例えば、`com.Acme.OldSecurity` というクラス内に `doLegacySecurity()` と呼ばれるメソッドがあり、安全でない暗号化アルゴリズムの使用が報告されている場合、以下の行を入力することで、無視させることができます。

```
Com.Acme.OldSecurity.DoLegacySecurity
```

  完全なメソッドシグネチャを指定し、末尾のパラメータ定義が欠如したり他の余分な文字を含めないようにしてください。このメソッドシグネチャが、検出された脆弱性のスタックトレースに対して照合されます。一致するものが含まれるメソッドシグネチャは全て検査の対象外になります。
- **入力**： 入力タイプを選択し、入力名を入力します。この入力が使用される検出結果は、全て表示されなくなります。
  - **Parameter**、**Header**、**Cookie** の場合：検出結果を非表示にする特定の入力名を指定してください。ワイルドカード*を使用すると、選択した入力タイプの全ての検出結果が非表示となります。
  - **QueryString** および **Body** の場合：それぞれ QueryString と Body 全体の検出結果が表示されなくなります。QueryString と Body は、以下で定義されている URL の例外パターンとの組合せでのみ例外として有効になります。

  例外の種類が**入力**の場合、「適用する URL」の項目で、URL の適用範囲を選択します。
  - **全ての URL**：指定された入力タイプと入力名が使用されている検出結果は、どこから発生したかに関係なく表示されなくなります。
  - **以下の URL**：例外を適用するパスを指定します。正規表現 (806ページ)とワイルドカード表記 (807ページ)を使用できます。

重要
プロトコルスキーム(`http://`または `https://`)やホスト名は含めず、`/`で始まるパス名のみを使用してください。

スラッシュの後にワイルドカード(`/.*`)を指定すると、全ての URL を記述する代わりに使用できます。
特定のルールで無視する必要がある URL を指定します。

- **URL**：特定のルールで無視する必要がある URL を指定します。例外にする URL のパスを 1 行ずつ記述します。正規表現 (806ページ)とワイルドカード表記 (807ページ)を使用できます。
- **キュー/トピック**： 特定のルールで無視されるべきメッセージキューやトピックを指定します。メッセージキューには 1 つのコンシューマがあり、トピックには複数のコンシューマがあります。
  現在、このオプションは Java エージェントのみでサポートされています。
  キュー/トピックの例外タイプの場合、「適用するキュー」で、キューまたはトピック名の適用方法を選択します。
  - **全てのキュー/トピック**：全てのキュー/トピックからの結果が表示されなくなります。
  - **以下のキュー**：例外とするキュー名またはトピック名のリストを指定します。キュー名を指定するか、正規表現 (806ページ)とワイルドカード表記 (807ページ)を使用できます。

6. **対象ルール**には、例外の影響を受けるルールの範囲を指定します。デフォルトでは全てのルールが対象になります。チェックボックス内をクリックして複数のオプションを選択できます。
   - **全てのルール**を選択すると、Assess と Protect の両方で検出される全ての脆弱性に対して例外が適用されます。
   - 「ASSESS」下にある**全ての Assess ルール**を選択すると、Assess が有効になっている場合、検出された全ての脆弱性に適用されます。
   - 「PROTECT」下にある**全ての Protect ルール**を選択すると、Protect が有効になっている場合、全ての攻撃イベントに適用されます。
   - 「ASSESS」セクションおよび「PROTECT」セクション下で、 個々のルールを選択することで、適用対象をさらに絞り込むことができます。例外は、選択したルールで検出された脆弱性にのみ適用されます。
     例外の種類として**入力**を選択した場合は、ユーザ入力によってトリガーされないルールのみを選択します。
   - 「ASSESS」および「PROTECT」下では、Assess と Protect で検出される個々のルールを複数選択できます。
7. 既に報告されているイベントを非表示にする場合は、**この例外と一致する全てのイベントを消去**の横のチェックボックスをオンにします。
8. **追加**を選択します。
   例外が、例外の一覧に追加されます。定義した条件に一致する入力は、適用したルールによって処理が行われなくなります。
   例外の一覧は、**アプリケーション > アプリケーション名 > ポリシー > 例外**または、**ユーザメニュー > ポリシーの管理 > アプリケーションの例外**で確認できます。一覧にあるトグルボタンを使用して、Assess または Protect の例外を有効/無効にできます。

正規表現リファレンス

アプリケーションの例外を追加 (804ページ)する際には、以下の表と例を参考にしてください。

適用	パターン	パターン例	一致例
文字列の先頭	`^`	`^w+`	`Start of a string`
文字列の末尾	`$`	`w+$`	`End of a string`
後に続く文字列の大文字と小文字を区別しない一致	`(?i)`	`(?i)%0a`	`%0a or %0A`
a、b、または c の内の 1 文字	`[abc]`	`[abc]+`	`a bb ccc`
a、b、c 以外の文字	`[^abc]`	`[^abc]+`	`Anythingbutabc.`
a-z までの範囲内の 1 文字	`[a-z]`	`[a-z]+`	`Only a-z`
a-z までの範囲内にない 1 文字	`[^a-z]`	`[^a-z]+`	`Anythingbuta-z.`
a-z または A-Z の範囲内の 1 文字	`[a-zA-Z]`	`[a-zA-Z]+`	`abc123DEF`
任意の 1 文字	`.`	`.+`	`abc`
空白文字	`\s`	`\s`	`anywhitespacecharacter`
空白以外の文字	`\S`	`\S+`	`any non-whitespace`
10 進数字	`\d`	`\d`	`not 1 not 2`
10 進数字以外	`\D`	`\D+`	`not 1 not 2`
0 個または 1 個の a	`a?`	`ba?`	`ba b a`

適用	パターン	パターン例	一致例
0 個以上の a	a*	ba*	a ba baa aaa ba b
1 個以上の a	a+	a+	a aa aaa aaaa bab baab
ちょうど 3 個の a	a{3}	a{3}	a aa aaa aaaa
3 個以上の a	a{3,}	a{3,}	a aa aaa aaaa aaaaaa
3 個以上 6 個以下の a	a{3,6}	a{3,6}	a aa aaa aaaa aaaaaa aaaa
ピリオド(ドット)はリテラル文字	.	a.b	string.string

ワイルドカード表記

アプリケーションで、入力、URL、キュー/トピックの例外を指定する場合に、以下を使用してワイルドカード式を作成できます。

- .*は、任意の文字が 0 個以上あることを意味します。
- .+は、任意の文字が 1 個以上あることを意味します。
- .?は、任意の文字が 0 個か 1 個あることを意味します。
- .は、任意の 1 文字を意味します。
- \.は、.というリテラルをエスケープするために使用します。
  例：somefile\.jsp

ワイルドカード表記の例

期待される結果	正規表現	例
全てのサブパスを例外にする	/myapp/.+	/myapp/で始まる URL があるパスを全て例外にする
サブパスの 1 文字を外したものを例外とする	/.yapp	yapp で終わる 5 文字のサブパス (myapp など)を全て例外にする
1 つのサブパスを明示的に例外とする	/myapp/thispath	/myapp/thispath のみを例外とする
パスの末尾によって例外とする	/.*ignore	ignore で終わるパスを全て例外とする
指定値を含むパスを例外とする	/.*value.*	value を含むパスを全て例外とする
指定値を含むパスを例外とする	/.?value.*	value で始まるパス、または value の前に 1 文字あるパスを全て例外とする
ピリオド(ドット)がワイルドカードではなくリテラル文字として使用されているパスを例外とする	/myapp\.js	myapp.js のみを例外とする この表記は 3 インスタンスまで使用可能

コンプライアンスポリシーの設定

組織内のアプリケーションのコンプライアンスに対するポリシーを定義できます。指定されたアプリケーションのいずれかがこのポリシーに違反している場合は、Contrast によってマークが付けられるので、すぐに確認して修正することができます(管理者にも E メールで違反が通知されます)。

コンプライアンスポリシーを設定するには：

1. ポリシーの管理 (780ページ)で**コンプライアンスポリシー**を選択します。
2. 既存のコンプライアンスポリシーがある場合は、そのリストが表示されます。トグルボタンを使用してポリシーを有効/無効にできます。また、**ゴミ箱**アイコンを使用してポリシーを削除できます。
3. 編集するポリシーの名前を選択するか、グリッドの上部にある**ポリシーを追加**を選択して新しいコンプライアンスポリシーを作成します。
4. 表示されるパネルに以下の項目を入力します。
   - **名前**：ポリシーの名前を選択します。
   - **ポリシー基準**：デフォルトは**全てのルール**ですが、脆弱性名を入力して、深刻度レベル、セキュリティ基準または Assess ルールごとに脆弱性を選択できます。
   - **アプリケーション**：デフォルトは全てのアプリケーションですが、アプリケーション名を入力して、重要度レベルや個々の名前ごとにアプリケーションを選択できます。

5. **追加**または**保存**を選択します。

注記
デフォルトポリシーでは、**名前**と**ポリシー基準**のフィールドがロックされていて削除できません。ただし、デフォルトポリシーのアプリケーションの選択は変更できます。

ヒント
有効化されたポリシーを使用して、コンプライアンスポリシーごとにアプリケーションを絞り込むことができます。これを行うにはアプリケーションを選択します。アプリケーションページで詳細リンクをクリックして、コンプライアンスポリシーごとにアプリケーションを絞り込みます。

注記
Contrast では、対象の脆弱性が修復されていない場合や対象のセキュリティ標準や Assess ルールの違反がある場合に、アプリケーションページで該当するアプリケーションにフラグが立てられます。アプリケーショングリッドの警告アイコンにカーソルを合わせるか、アプリケーションの詳細ページに移動すると、違反しているポリシーへのリンクが表示されます。

IP の管理

拒否リスト、許可リスト(信頼できるホスト)、ソース名に関して、組織の IP ポリシーを管理できます。

注記
拒否リストおよび許可リストに対して、Contrast はや `X-Forwarded-For` リクエストヘッダをチェックして、IP アドレスがリストにあるエントリと一致するかどうかを確認します。

- **IP 拒否リスト**: Contrast Protect でリスト内の全ての IP アドレスをブロックするようにルールを設定します。
  拒否リストの使用は、より永続的な Protect ポリシーを導入できるまでの間か、調査を実施できるようになるまでの間の、即時的なトリアージに適しています。
- **IP 許可リスト**：脆弱性スキャンを実行する信頼できるホストを登録して、安全な IP アドレスとして設定します。このリストに登録された IP アドレスからのデータは Contrast では表示されなくなります。
  IP 許可リストのエントリ(登録された IP アドレス)は、IP 拒否リストのエントリを上書きしません。
  Contrast Assess 機能には影響せず、通常通り機能します。
  Contrast Protect は、このリストにあるエントリと一致する全ての IP アドレス(または IP 範囲)を無視します。リストに指定されている IP アドレスからの攻撃を監視したり、ブロックをしません。

- **ソース名：** 1 つ以上の IP アドレスまたはサブネットマスクで、既知のソース(ペネトレーションテスト担当者など)から発生する攻撃イベントにラベルを付けることができます。
  攻撃 > 監視および**攻撃の詳細**ページで攻撃を表示すると、攻撃者の IP 情報の代わりにソース名が表示されます。この値を表示することによって、想定しているイベントと注意が必要な攻撃イベントをすばやく特定して区別することができます。

関連項目

- IP の拒否/許可 (809ページ)
- ソース名の管理 (810ページ)

IP アドレスの拒否と許可

IP 拒否リストと IP 許可リストを使用して、 組織内の IP アドレスを管理 (808ページ)します。

開始する前に

- 組織のロールとして、管理者(Admin)またはルール管理者(Rules Admin)が必要です。
- CIDR(Classless Inter Domain Routing)表記を使用して、サブネットマスクを指定します。

手順

1. ユーザメニューから、**ポリシーの管理 > IP 管理**を選択します。
2. 拒否リストの管理：
   a. **IP 拒否リスト**タブを選択します。
   b. 既存の拒否リストを編集する場合は、拒否リスト名を選択して情報を変更したら、**保存**を選択します。
   c. 拒否リストに IP アドレスを新規に登録する場合は、**拒否リストに IP を登録**を選択して情報を入力したら、**追加**を選択します。

3. 許可リストの管理：
   a. **IP 許可リスト**タブを選択します。
   b. 既存の許可リストを編集する場合は、許可リスト名を選択して情報を変更したら、**保存**を選択します。
   c. 許可リストに信頼できるホストを新規に登録する場合は、**信頼できるホストを登録**を選択して情報を入力したら、**追加**を選択します。

ソース名の管理

ソース名を使用すると、組織内の攻撃イベントを監視しながら、脅威ではない内部トラフィックやテストをすばやく特定できます。

1 つまたは複数の IP アドレスやサブネットマスクに、任意のソース名を付けることができます。ソース名を保存すると、**攻撃 > 監視**を選択する、または**攻撃の詳細**ページを表示すると、(ユーザの IP 情報ではなく)ソース名を確認できます。これにより、攻撃イベントを評価する際に、ソース名の付いた攻撃者を既知の攻撃元として、すばやく特定できます。

ソース名を作成するには：

1. **ユーザメニュー > ポリシーの管理 > IP 管理 > ソース名**にアクセスします。
2. **ソース名を登録**を選択します。
3. 1 つまたは複数の IP アドレスを識別するために使用する**名前**を入力します。
4. このソース名で識別する **IP アドレス/サブネットマスク**を追加します。必要に応じて、**IP アドレス/サブネットマスクを追加**のリンクをクリックして、IP アドレスまたはサブネットマスクをさらに追加します。
5. ドロップダウンメニューを使用して、ソース名を使用する**開始**日時と**終了**日時を選択します。過去の日付を開始日時とするカスタム期間を作成することもできます。この場合、ソース名は過去の攻撃イベントに遡って適用されます。
6. 入力が完了したら、**追加**をクリックしてソース名を保存します。
   組織にソース名が追加されると、**監視**ページや**攻撃**の詳細ページで、条件が一致する攻撃に対してソース名が表示されます。これは、攻撃を監視 (710ページ)するのに役立ちます。
   攻撃イベントについて報告されたデータが複数のソース名と一致する場合、Contrast では最後に更新された名前が適用されます。
7. ソース名を編集するには、ソース名を選択します。**ソース名を編集**画面が表示されますので、変更をし、**保存**を選択します。
8. ソースを削除するには、**ソース名一覧**で**削除**アイコンを選択するか、**ソース名を編集**の画面の下にある**削除**アイコンを選択します。名前を削除すると、その名前への全ての参照は、IP 情報に置き換わります。

ライブラリポリシーの設定

重要
ライセンスポリシーは、Contrast SCA をご利用のお客様のみが利用できます。
Contrast SCA を有効にするには、組織の管理者にご連絡ください。

Contrast では、組織の基準を満たさないライブラリにフラグが立てられるため、アプリケーションが安全であるかを確認できます。

ライブラリが制限されている、もしくは、特定のバージョンより古いものがアプリケーションで使用されている場合、Contrast ではポリシー違反としてマークされます。また、Contrast のインターフェイスで"F"という文字を付けてポリシー違反のライブラリを自動的に評価付けすることもできます(管理者には、Contrast 内と E メールの両方で違反が通知されます)。

注記
ライブラリのバージョンには、メジャーバージョン、マイナーバージョン、パッチバージョンが含まれます。

ライブラリポリシーを設定するには：

1. ユーザメニューで**ポリシーの管理 > ライブラリポリシー**を選択します。
2. **ライブラリを制限**するチェックボックスをオンにして、ポートフォリオから除外するライブラリを選択します。ライブラリは複数選択できます。
3. **バージョン要件を有効化**するチェックボックスをオンにして、指定したバージョン数の範囲内とする 1 つ以上のライブラリを選択します。
4. **他の要件を追加**のリンクをクリックすると、追加のライブラリのバージョン要件を作成できます。
5. **ライセンスを制限**のチェックボックスをオンにして、制限するオープンソースライセンスにポリシーを設定します。オープンソースライセンスが制限されている場合、その制限されたライセンスを使用するライブラリはポリシー違反としてマークされます。
   ライセンスポリシーでは、オープンソースライブラリが SPDX 形式で、識別子にフルネームが続く形式でリストされます。制限するライセンスの種類を選択する必要があります。ポートフォリオ内で認識される「以降」のライセンスも含まれます。例えば、GPL-3.0-only でライセンスを制限する場合、GPL-3.0 以降のライセンスは全てその制限に含まれます。
6. **ポリシー違反のライブラリ**のチェックボックスをオンにすると、設定されたポリシーに違反するライブラリに対して、低いスコアが自動的に割り当てられます。
   設定されたポリシーに準拠していないライブラリは、**ライブラリ**ページで、名前と警告アイコンおよびライブラリスコアが赤色で強調表示されます。違反の詳細については、アイコンの上にカーソルを合わせるか、ライブラリの**概要**ページにアクセスしてください。
   ライブラリを自動的に低いスコアにするように選択した場合、スコア設定を調整 (834ページ)すると、組織管理者に通知されます。

機密データのマスキング

機密データのマスキング機能は、組織のリスクを低減し、コンプライアンスの要件を満たすのに役立ちます。

データマスキングは、Contrast Web インターフェイス、Syslog またはセキュリティログに送信される脆弱性や攻撃レポートの情報にマスクをかけることで、アプリケーション内の機密データを保護します。

Contrast では、機密データやデータタイプをいくつかのカテゴリに分類しています。これらは特定のキーワードで構成され、エージェントが自動的に識別しレポート内でマスクされます。RulesAdmin 以上の権限を持つユーザは、機密データを管理 (812ページ)できます。

Contrast エージェントにより、クエリパラメータ、リクエストヘッダ、Cookie およびボディの機密データがマスクされます。エージェントによって、入力名に使用されている特定のキーワードが検索されて機密データが認識されます。一致するものが見つかった場合、その入力値にはマスクがかけられ、`contrast-redacted-{datatype}`という形式のプレースホルダに置き換えられます。`datatype` の部分は、キーワードが含まれる機密データの分類になります。

Contrast エージェントは、`application/x-www-form-urlencoded` 以外のコンテンツタイプがあるリクエストボディの個々のフィールドはマスク**しません**。ただし、リクエストボディ全体をマスクするようにエージェントを設定することができます。また、Assess を使用している場合の脆弱性レポートのデータフローに表示されるデータや、Protect を使用している場合の攻撃イベントのベクトルで表示されるデータもマスクされません。

注記

Contrast ログに記録されるステートメントには、Contrast エージェントによって「可能な限り」機密データが出力されないようにしていますが、ログレベルが DEBUG 以下に設定されていると、Contrast ログに機密データが記録される可能性があります。できる限り、本番環境のシステムでは、DEBUG 以下でログを記録するよう設定しないでください。機密データを扱うシステムが、DEBUG 以下でログを記録するように設定されている場合は、機密データの漏洩を防ぐために、これらのログが外部システムに送信されていないことを確認する手順を実行する必要があります。

例えば、以下は脆弱性レポートの一部としてエージェントによって送信された HTTP リクエストで、エージェントによって機密データと判断された 2 つの入力があります。これらは、Contrast Web インターフェイス、syslog サーバやセキュリティログに送信される前に入力値をマスクするために使用されたプレースホルダを示しています。

```
PUT /employee/5 HTTP/1.1
Host: yourdomain.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
apikey: contrast-redacted-authentication-info

ssn=contrast-redacted-government-id&department=sales
```

この場合、ヘッダの値の「apikey」が「Authentication Info」データタイプにあるキーワードと一致しているためマスクされます。また、Contrast の組織にある「Government ID」データタイプのキーワードと「ssn」が一致しているため、フォームパラメータもマスクされます(キーワードの一致では、大文字と小文字は区別されません)。

機密データタイプの管理

機密データのマスキング (811ページ)は、組織のリスクを低減し、コンプライアンス要件を満たすのに役立ちます。

1. ポリシーの管理で**機密データ**を選択します。
2. ここでは機密データタイプのリストがアルファベット順に表示されます。検索フィールドを使用して、名前またはキーワードで特定のタイプを検索できます。
3. **本文全体をマスクする**の横にあるチェックボックスをオンにして、HTTP リクエストボディ全体のリダクションを有効にします。これは、組織内の全てのアプリケーションに適用されます。
4. Contrast で重大と判断されているデータタイプとキーワードは、デフォルトで組織内の全てのアプリケーションに適用され、編集したり無効にしたりすることはできません。Contrast で重大と判断されていないデータタイプに関しては、グリッドのトグルボタンを使用して、組織で有効/無効にできます。
5. グリッドでデータタイプの名前をクリックして、カスタムキーワードを追加します。**機密データタイプを編集**画面で、**カスタムキーワードを追加**を選択するとキーワードを追加でき、これらが適用されるアプリケーションを指定できます。 **デフォルトキーワード**は編集できず、全てのアプリケーションに適用されます。
6. **保存**を選択します。

ルール管理者として通知を追加/編集

通知のデフォルト設定は組織管理者(Admin)によって行われますが、ルール管理者(Rules Admin)も既存の通知を有効/無効にしたり、新しい通知を作成することができます。

1. 組織の設定 (818ページ)で、**通知**を選択します。
2. トグルボタンを使用して、既存の通知を有効または無効にします。
3. 新しい通知を作成するには、**通知を作成**を選択します。
4. 表示される画面で、以下の項目を入力します。
   - 名前
   - 頻度
   - 説明
   - アプリケーション
   - アプリケーションのタグ
   - ユーザ
5. **保存**を選択します。

組織

組織の管理者権限(Admin)がある場合は、ルール管理(Rules Admin) (780ページ)や編集(Edit) (519ページ)権限で実行できる全ての操作に加えて、組織レベルで次の操作を実行できます。

- 組織の設定を行う (818ページ)
- Assess を有効にする (813ページ)
- Protect を有効にする (815ページ)

注記

組織管理者(Admin)は、組織レベルで最も高い権限を持ちます。その他の組織ロール (955ページ)にも、組織全体におよぶ機能があります。

システムレベルで設定 (913ページ)すると、ユーザは複数の組織にわたって組織管理(Admin)ロールを持つことになります。

Assess を有効にする

Assess の利用を設定するには、設定ファイル、変数、または Contrast Web インターフェイスを使用することができます。この手順では、Contrast Web インターフェイスで Assess を有効にする方法について説明します。YAML 設定ファイル (63ページ)と環境変数 (66ページ)を使用する場合は、Contrast Web インターフェイス以外の方法を使用してエージェントの設定を行うことになります。

開始する前に

- Assess ライセンスなしでも Contrast で検出された脆弱性の種類が表示されますが、アプリケーションにライセンスを適用しないと詳細情報は取得できません。

手順

1. Contrast Web インターフェイスにログインします。
2. Contrast Web インターフェイスのナビゲーションバーで、**サーバ**を選択します。
3. スクロールするか、ページ上部の検索を使用して、Assess をで検査を行うアプリケーションに関連付けられているサーバを見つけます。
4. Contrast Web インターフェイスで特定のサーバの Assess 利用の設定を行うには、次のいずれかの方法を使用します。

- サーバの一覧で、Assess 列のトグルボタンを選択します。
- サーバの一覧でサーバ名を選択して、サーバの詳細画面を表示し、そこで Assess のトグルボタンを使用します。

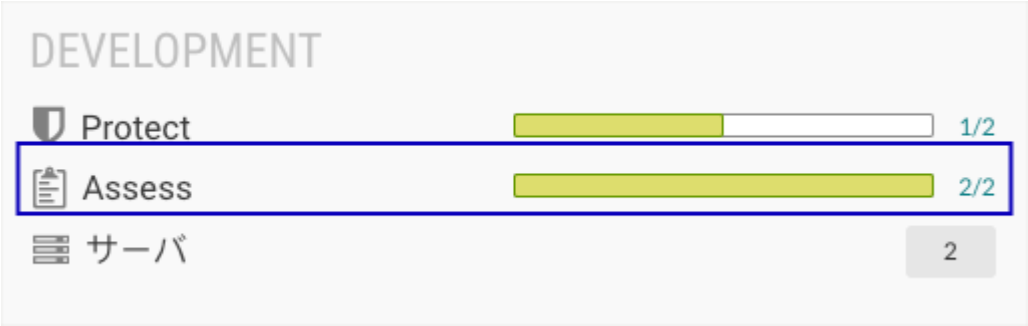
注記

- Assess のオン/オフの切り替えに Contrast Web インターフェイスのみを使用する場合、Assess の設定は、オンなら緑色、オフなら灰色になります。この設定は、Contrast Web インターフェイスで変更できます。

- Contrast Web インターフェイス以外の方法で Assess の設定を指定した場合(エージェント設定ファイルなど)、オンなら緑色で使用不可になり、オフなら灰色で使用不可になります。この場合は、Contrast Web インターフェイスから変更できません。

- Contrast Web インターフェイスからの設定が使用不可になっている場合は、その設定にカーソルを合わせて、設定されている箇所を確認できます。Contrast でどの設定を有効な設定として使用するかは、優先順位 (62ページ)によって決まります。

5. アプリケーションが、関連付けられている各サーバで、Assess を使用しているかどうかを確認するには、アプリケーションページへアクセスします。
   a. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   b. アプリケーションの一覧で、アプリケーション名をクリックします。
   c. 少なくとも 1 つのサーバで Assesst がオンになっている場合、概要タブの各環境に、Assess ラベルの横にバーが表示されます。これは、アプリケーションに関連付けられている全てのサーバの Assess のステータスを表します。バーの緑色は、Assess が有効になっているサーバの数を表します。バーの白色は、Assess が有効になっていないサーバの数を表します。



   Assess がオンになっているサーバがない場合は、オフのアイコン(オフ)が表示されます。
   d. アプリケーションが、関連付けられているサーバに対して Assess を使用するように設定されているかを確認するには、 Assess のラベル横のバーを選択すれば、フィルタされたサーバの一覧が表示されます。

6. 新しいサーバに対するデフォルトの Assess の設定を指定するには、ユーザメニューから**組織の設定 > サーバ**を設定し、そこで Assess のトグルを使用します。
7. Assess で検査を行いたいアプリケーションにライセンスが適用されていない場合、 アプリケーションにライセンスを追加します。
   a. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   b. アプリケーション名の横にある**ライセンスなし**を選択します。
   c. 「ライセンスを適用」画面で、**ライセンスを適用**を選択します。
   d. Contrast エージェントがアプリケーションで Assess 機能を使用して解析できるようにするために、アプリケーションサーバを再起動してください。再起動が完了すると、Contrast で脆弱性の解析情報の受信が始まります。アプリケーションの横に**ライセンスなし**が表示されなくなります。これは、Assess ライセンスが割り当てられていることを意味します。

> **注記**
> 組織管理者(Admin)は、全てのアプリケーションに対して手動でライセンスを適用するのではなく、新しいアプリケーションに自動的に Assess ライセンスを適用するよう、デフォルトの設定を指定できます。
>
> 1. 組織の設定 (818ページ)で、**組織**を選択します。
> 2. 「ライセンス」セクションの Assess の下にある、**新しいアプリケーションにライセンスを自動で適用**を選択します。

Protect を有効にする

ユーザに対して Protect を有効にすると、そのユーザが Protect データにアクセスして表示できるようになります。サーバに対して Protect を有効にすると、アプリケーションが Protect を使用して攻撃の監視やブロックができるようになります。

> **注記**
> 既存のアプリケーションがあるサーバで Protect を有効にした場合は、Protect を有効にするためにアプリケーションを再起動してください。

開始する前に

- **組織の設定 > ユーザ**に移動し、 Protect データと Protect の設定にアクセスする権限があることを確認します。

- SaaS 版をご利用のお客様の場合、Contrast Security が各組織および組織内のユーザロールに Protect 権限を付与します。
- オンプレミス版をご利用のお客様の場合、 組織に Protect 権限を付与するには (915ページ)、SuperAdmin か ServerAdmin、または SystemAdmin ロールが必要です。
  これらのロールは、どのユーザロールが Protect データにアクセスできるかを設定することもできます。

- サーバに適用できるライセンス (820ページ)があることを確認してください。
- ユーザに対して Protect を有効にするには、組織管理者(Admin)ロールが必要です。

ユーザが Protect データにアクセスできるようにする

1. Contrast Web インターフェイスにログインします。
2. ユーザが Protect データを表示・使用できるようにするには：
   a. ユーザメニューで、**組織の設定**を選択します。
   b. **ユーザ**を選択します。
   c. Protect データへのアクセスを必要とするユーザ毎に、Protect のトグルボタン( )をオンにします。
   d. 新しい設定を有効にするために、ユーザは Contrast Web インターフェイスからログアウトし、再度ログインするよう指示してください。

サーバに対して Protect を有効にする

Protect を設定するには、設定ファイル、変数、または Contrast Web インターフェイスを使用することができます。この手順では、Contrast Web インターフェイスで Protect を有効にする方法について説明します。YAML 設定ファイル (63ページ)と環境変数 (66ページ)を使用する場合は、Contrast Web インターフェイス以外の方法を使用してエージェントの設定を行うことになります。

1. 各環境で新規サーバ対するデフォルトの Protect 設定を指定する場合：
   Contrast Web インターフェイス以外の方法を使用して特定のサーバに Protect の設定を指定している場合、その設定によりこのデフォルトの設定は上書きされます。
   a. ユーザメニューから、**組織の設定**を選択します。
   b. **サーバ** (828ページ)を選択します。
   c. 環境を選択します。
   d. Protect のセクションで、Protect のトグルボタンをオンにします。

> **注記**
> 自動的にライセンスを適用する (820ページ)には、組織管理者(Admin)またはルール管理者(Rules Admin)ロールが必要です。このオプションは、サーバの Protect を 1 台づつ手動で有効にしたくない場合に便利です。

2. 特定のサーバに対して Protect を有効にする場合：
   a. Contrast Web インターフェイスのナビゲーションバーで、**サーバ**を選択します。
   b. Contrast Web インターフェイスで Protect を有効にするには、Protect のトグルボタン( )をオンにします。次のいずれかの方法を使用します。
      - サーバの一覧で、**Protect** 列の設定をオンします。
      - サーバの一覧で、サーバ名を選択し、「概要」タブで Protect のトグルボタンをオンにします。

注記

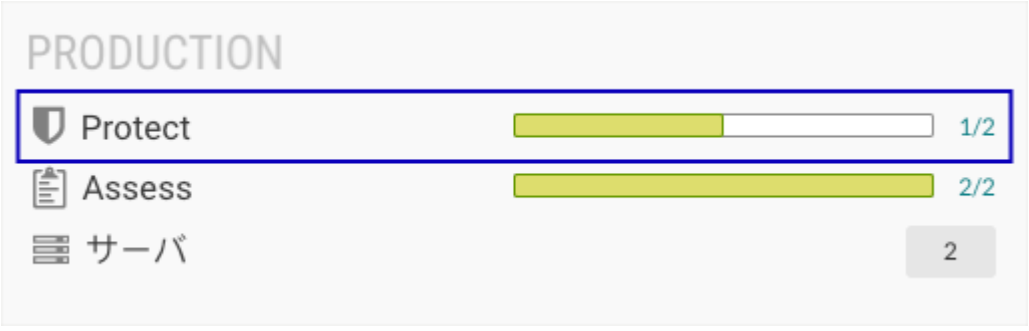
- Protect のオン/オフの切り替えに Contrast Web インターフェイスのみを使用する場合、特定のサーバに対する Protect の設定は、オンなら緑色、オフなら灰色になります。この設定は、Contrast Web インターフェイスで変更できます。

- Contrast Web インターフェイス以外の方法で Protect の設定を指定した場合(エージェント設定ファイルなど)、オンなら緑色で使用不可になり、オフなら灰色で使用不可になります。この場合は、Contrast Web インターフェイスから変更できません。

- Contrast Web インターフェイスからの設定が使用不可になっている場合は、その設定にカーソルを合わせて、設定されている箇所を確認できます。Contrast でどの設定を有効な設定として使用するかは、優先順位(62ページ)によって決まります。

3. 特定のサーバで Protect がオンになっていることを確認するには、サーバタブで対象のサーバを選択し、**概要**を選択して、Protect のトグルボタンが緑色であることを確認します。
   - サーバに関連付けられている 1 つ以上のアプリケーションで Protect を使用するよう設定されていない場合は、Protect のトグルボタンの横に警告のアイコンが表示されます。

4. アプリケーションが、関連付けられている各サーバで、Protect を使用しているかどうかを確認するには、アプリケーションページへアクセスします。
   a. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   b. アプリケーションの一覧で、アプリケーション名をクリックします。
   c. 少なくとも 1 つのサーバで Protect がオンになっている場合、概要タブの各環境に、Protect ラベルの横にバーが表示されます。これは、アプリケーションに関連付けられている全てのサーバの Protect のステータスを表します。バーの緑色は、Protect によって保護されているサーバの数を表します。バーの白色は、Protect によって保護されていないサーバの数を表します。



   Protect がオンになっているサーバがない場合は、オフのアイコン(オフ)が表示されます。
   d. アプリケーションが、関連付けられているサーバに対して Protect を使用するように設定されているかを確認するには、 Protect のラベル横のバーを選択すれば、フィルタされたサーバの一覧が表示されます。

ライセンス適用の動き

こららの条件が揃う場合、サーバに自動的に Protect ライセンスが適用されます。

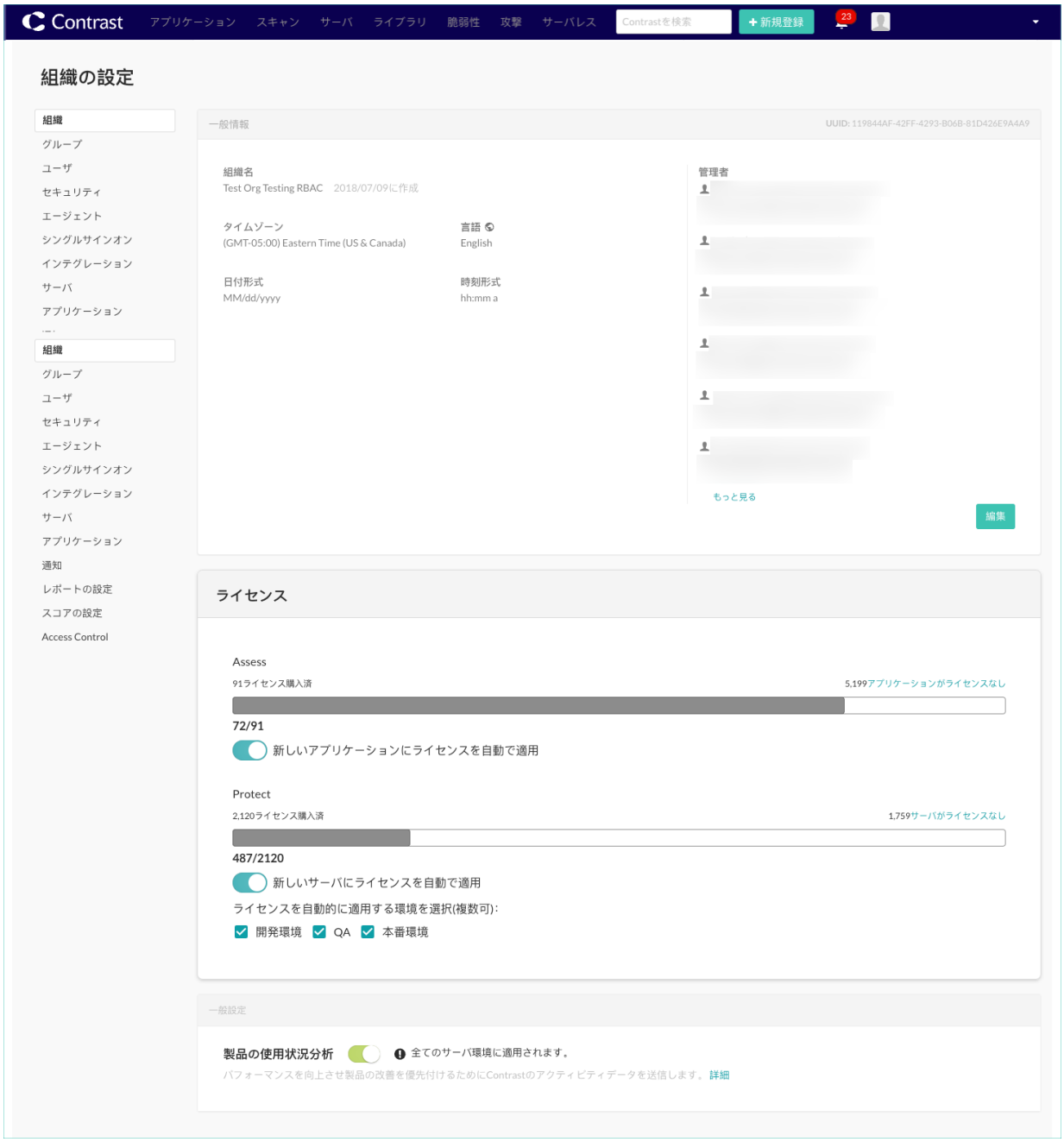
- 組織で Protect が有効である。
- 組織で Protect ライセンスの自動適用が有効である。
  [en] If Protect is enabled with a method external to the Contrast web interface, that effective configuration overrides the automatic licensing option in the Contrast web interface.
- サーバは、ライセンスの自動適用が有効になっている 1 つ以上の環境に存在する。

また、エージェントの設定ファイルで Protect の設定を使用する場合、ライセンス適用の動きは次のようになります。

- **エージェントの設定ファイルで Protect が有効**
  - アプリケーションの起動時に Protect ライセンスが利用可能な場合は、非常に短時間でサーバにライセンスが適用されます。エージェントがアプリケーションを登録すると同時に、自動的にライセンスが消費されます。
  - アプリケーションの起動時に利用可能な Protect ライセンスが**無い**場合は、エージェントが Contrast と通信するたびに、Contrast はサーバへのライセンスの適用を試みます。
- **エージェントの設定ファイルで Protect が無効**
  - アプリケーションの起動時に Protect ライセンスが利用可能な場合、サーバにライセンスが適用されます。アプリケーションが Contrast に登録されると、サーバに適用されたライセンスが消費されます。
  - アプリケーションの起動時に利用可能な Protect ライセンスが**無い**場合、サーバにライセンスは適用されません。

組織の設定を行う

組織管理者の権限(組織の Admin ロール)があるユーザは、組織の設定を行うことができます。



[en] Steps

1. 複数の組織がある場合は、**ユーザメニュー**より設定する組織の名前を選択します。
2. **ユーザメニュー**で、**組織の設定**を選択します。以下を設定できます。

   [en] The organization settings are:

   - 組織(一般情報 (820ページ)、ライセンス管理 (820ページ)、使用状況分析 (825ページ))
   - ユーザ、グループ、権限 (822ページ)
   - セキュリティ(パスワード (826ページ)、2 段階認証 (827ページ)、IP 範囲 (827ページ)、E メールドメインの制限 (826ページ))
   - API(組織キーとエージェントキーの表示)
   - シングルサインオン(SSO) (827ページ)
   - インテグレーション (726ページ)
   - サーバ (828ページ)
   - アプリケーション (830ページ)

- 通知 (831ページ)
- スコアの設定 (834ページ)

組織の一般情報の設定

1. 組織の設定 (818ページ)より、左側のナビゲーションメニューから**組織**を選択します。
2. 右上にこの組織の UUID が表示されます(これは、ユーザの一括登録 (911ページ)の際に便利です)。
3. このパネルには、以下のように組織の一般的な情報も表示されます。
   - 組織の名前
   - 組織のデフォルトの日付形式と時刻形式
   - 組織のデフォルトの言語設定：日本語(スーパー管理者によって有効にされている場合は選択可)または英語

   > **注記**
   > 個々のユーザ設定は、言語設定を除き、組織のほとんどの一般情報(時間や日付のフォーマットなど)よりも優先されます。 ユーザの言語設定は、ユーザへの通知や電子メールなど、ユーザ固有の項目については、組織の言語設定よりも優先されます。ただし、組織レベルの通知など、組織に固有のものについては、ユーザの言語設定は組織の言語設定を上書きしません。

4. このパネルにある情報を変更するには、**編集**を選択します。
5. 変更を加えたら、**保存**を選択します。

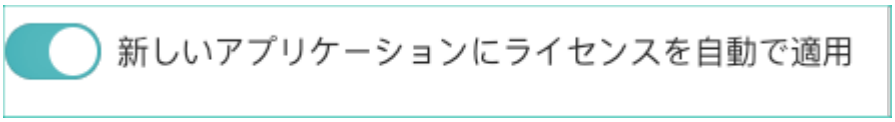
組織、アプリケーション、サーバへのライセンス割当て

開始する前に

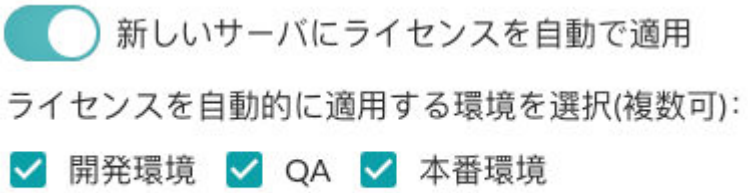
- 組織管理者のロール(Admin)が必要です。
- 割り当てられた Assess ライセンスが全て組織で使用された場合、SaaS 版をご利用のお客様は弊社サポートにお問い合わせください。
- オンプレミス版のお客様は、スーパー管理者(SuperAdmin ロールのユーザ)がシステムレベルで利用可能なライセンスを増やす (939ページ)ことができます。

手順

1. 組織でのライセンス使用状況の概要を確認します。
   a. ユーザメニューから、**組織の設定**を選択します。
   b. **組織**を選択します。
   c. **ライセンス管理**の下で、Assess ライセンスと Protect ライセンスについての情報を確認できます。
      - 使用可能および使用中の Assess(アプリケーション)ライセンス数、ライセンスなしのアプリケーションの数。
      - 使用可能および使用中の Protect(サーバ)ライセンス数、ライセンスなしのサーバの数。
        購入したライセンスより多くのライセンスを利用している場合は、ライセンスバーの使用可能なライセンス欄が、追加で使用中のライセンス数の表示に置き換わります。
2. (任意)**ライセンス管理**で、新しいアプリケーションやサーバにライセンスを自動的に適用するよう設定できます。
   a. Assess ライセンスの場合、**新しいアプリケーションにライセンスを自動で適用**をオンにします。



b. Protect ライセンスの場合、**新しいサーバにライセンスを自動で適用**をオンにします。
サーバの環境を選択します。



Protect の設定に Contrast Web インターフェイス以外の方法(例えば、エージェント設定ファイル)を使用した場合、その設定が Contrast で有効な設定と見なされます。有効と見なされた設定は、Contrast Web インターフェイスでの自動ライセンスの設定よりも優先されます。

- 有効なエージェントの設定で Protect がオフになっており、自動ライセンスの設定がオンになっている場合は、Contrast Web インターフェイスでの設定は無視されます。ライセンスは、自動的にサーバに適用されません。
- 有効なエージェントの設定で Protect がオンになっており、自動ライセンスの設定がオフになっている場合は、Contrast Web インターフェイスでの設定は無視されます。ライセンスは、サーバに自動的に適用されます。
- 有効なエージェントの設定が報告されていない場合は、Contrast Web インターフェイスの自動ライセンスの設定が全ての新しいサーバに適用されます。

3. 必要に応じて、個々のアプリケーションに対して Assess ライセンスを適用できます。
   a. Contrast Web インターフェイスのナビゲーションバーで**アプリケーション**を選択します。
   アプリケーションの一覧で、ライセンスが付与されていないアプリケーションは、アプリケーション名の横に**ライセンスなし**と表示されます。
   アプリケーションにライセンスが付与されていない場合、アプリケーションの脆弱性情報は表示されません。
   b. ライセンスが付与されていないアプリケーションにライセンスを適用するには、**ライセンスなし**を選択します。
   c. 「ライセンスを適用」画面で、**ライセンスを適用**を選択します。
4. 必要に応じて、アプリケーションを削除する (535ページ)ことによって、アプリケーションから Assess ライセンスを削除できます。
   アプリケーションに適用されたライセンスは、最大許容アプリケーション数に対して永続的にカウントされます。ライセンスが適用されたアプリケーションを削除しても、アプリケーションに許容されるライセンス数には影響はありません。
5. 必要に応じて、個々のサーバから Protect ライセンスを追加または削除できます。
   a. Contrast Web インターフェイスのナビゲーションバーで、**サーバ**を選択します。
   b. そのサーバと関連付いているアプリケーションを再起動します。
   - Protect のオン/オフの切り替えに Contrast Web インターフェイスのみを使用する場合、特定のサーバに対する Protect の設定は、オンなら緑色、オフなら灰色になります。この設定は、Contrast Web インターフェイスで変更できます。

   - Contrast Web インターフェイス以外の方法で Protect の設定を指定した場合(エージェント設定ファイルなど)、オンなら緑色で使用不可になり、オフなら灰色で使用不可になります。この場合は、Contrast Web インターフェイスから変更できません。

- Contrast Web インターフェイスからの設定が使用不可になっている場合は、その設定にカーソルを合わせて、設定されている箇所を確認できます。Contrast でどの設定を有効な設定として使用するかは、優先順位 (62ページ)によって決まります。

c. そのサーバと関連付いているアプリケーションを再起動します。

組織レベルでのユーザ、グループ、権限の管理

システムレベルで許可 (910ページ)されている場合、組織管理者(Admin ロールのあるユーザ)は以下の方法で組織のユーザ権限を管理できます。

- ユーザを追加またはユーザ設定を編集する (822ページ)
- アクセスグループ (822ページ)で組織の権限を管理する

組織レベルでのユーザの追加/編集

組織内でユーザを追加すると、その組織内のアプリケーションのアクセスグループにユーザを割り当てることができます。ほとんどの Contrast インストールでは、デフォルト (910ページ)ではロールと権限を組織レベルで設定しますが、複数の組織にアクセスする必要のあるユーザがいる場合は、システムレベルにそれらのユーザを追加 (910ページ)します。

ユーザを追加するには：

1. 組織の管理者権限(Admin ロール)のあるユーザとして、Contrast にログインします。
2. **ユーザメニュー**で、**組織の設定**を選択します。
3. 左側のナビゲーションで**ユーザ**を選択します。
4. 一覧からユーザ名を選択してエントリを編集するか、**ユーザを追加**を選択して新規ユーザを追加します。
5. ユーザについて、以下の項目を入力します。
   - **組織ロール**：この組織内の全てのアプリケーションに適用されるデフォルトの組織ロール (955ページ)の 1 つを選択するか、カスタムアクセスグループを作成 (822ページ)します。
   - **アプリケーションアクセスグループ**：この組織の全てのアプリケーションに適用されるデフォルトのアプリケーションロール (953ページ)の 1 つを選択するか、カスタムアクセスグループを作成して特定のアプリケーションへの特定の権限を付与します。
   - **アクセス権限**：ユーザに、API、Contrast の Web インターフェイス、Protect データへのアクセス権を付与できます(Protect 権限は、システムレベルで付与 (915ページ)することもできます)。

   ヒント
   ユーザに管理者ロールを割り当てる場合は、API と Contrast の Web インターフェイスの両方についてアクセス権を付与するようにしてください。
6. **追加**または**保存**を選択します。

組織のアクセスグループの追加/編集/削除

組織のアクセスグループを使用して、ロール (953ページ)ごとにユーザに権限や機能を割り当てることができます。

Contrast ではデフォルトのアクセスグループを提供しており、自分で作成する代わりに使用することができます。

- **View**：このグループのメンバーは、Contrast インターフェイスに読み取り専用でアクセスし、スコア、ライブラリ、脆弱性、コメントなどを参照できます。
- **Edit**：このグループのメンバーは、検出結果の修復、タグの追加、脆弱性の管理、属性の編集、アプリケーションのマージ、アプリケーションの追加・削除、サーバの作成などが可能です。
- **Rules Admin**：このグループのメンバーは、アプリケーションのルールとポリシーの編集、Protect の有効化、通知とスコアの管理を行うことができます。
- **Admin**：このグループのメンバーは、組織の設定を構成し、管理することができます。

開始する前に

組織管理者のロール(Admin)が必要です。

手順

1. **組織の設定 (818ページ)より、グループを選択します。**
2. 既存のグループを選択して編集するか、**グループを追加**を選択して新しいグループを作成します。

ヒント

グループを検索するには、クイックフィルターのドロップダウンまたは左上の検索フィールドを使用します。または、各列の上部にある上下方向の矢印を使用して並び替えを行います。

Contrast が提供するデフォルトグループはロックアイコン付きで表示されますが、アプリケーションとロールは固定されており、削除はできません。これらのデフォルトグループには、ユーザの追加か削除のみが可能です。

3. 画面で以下の項目に入力します。
   - **グループ名**：このグループに割り当てる権限、機能、目的を反映したものを選んでください。
   - **アプリケーションアクセス**：ここでアプリケーション名を選択すると、このグループとアプリケーションが関連付けられます。新しいアプリケーションを設定する際に、グループ名を指定することもできます。
   - **ロール**：このグループのメンバに対してアプリケーション内で持たせるアプリケーションロールを選択します。
   - さらにアプリケーションとロールを追加するには、**アクセスを追加**を選択します。
4. **メンバー**では、フィールドに文字を打鍵するとユーザが表示されるので、グループに割り当てる 1 人以上のユーザを選択します。
5. 入力が完了して**追加**を選択すると、新しいグループが作成されます。

注記

ユーザが、全てのアプリケーションや組織のロールに対して競合する 2 つのグループに割り当てられた場合は、アクセスの制限が最も厳しいロールが適用されます。

6. グループを削除するには、**ユーザメニュー > 組織の設定 > グループ**を選択します。削除するグループを探し、その行の削除アイコンを選択します。
   この操作を確定すると、グループが削除され、そのグループによって提供されたアクセス権が、そのグループに割り当てられた全てのユーザから取り消されます。

ヒント
組織内の全てのアプリケーションに対するロールをユーザに割り当てるには、デフォルトロールグループの組織ロールとアプリケーションロールの両方を割り当てます(例、組織とアプリケーションの両方のロールに管理者権限である「Admin」を設定すると、組織内の全てのアプケーションの管理者権限を持つことになります)。

特定のアプリケーションへのアクセス権をユーザに付与するには、そのアプリケーションのアクセスグループを作成し、そのグループにユーザを追加する必要があります。どのアプリケーションアクセスグループにも割り当てられていないユーザは、アクセスがありません。ユーザは、1 つの組織内の各アプリケーションで様々なロールを持つことができます。

Contrast のほとんどのお客様は、単一の組織で展開しています。組織レベルで作成されたグループは、その組織全体のロールと権限に影響を与えます。組織のアクセスグループをシステムレベル (913ページ)で作成し、ユーザを複数の組織にアクセスさせることも可能です。

API のみのユーザの作成

全てのプラグインやインテグレーションで使用できる、API のみのユーザを作成します。

参考：ベストプラクティスとして、プラグインやインテグレーションを使用するためだけのユーザアカウントを登録することを推奨します。この専用ユーザにより、あるユーザが会社や組織を離れた際に、使用しているプラグインやインテグレーションが機能するよう継続できます。そのユーザアカウントが削除されて問題が発生するような状況を避けることができます。

API のみのアカウントは、通知設定が有効になっている場合でも、電子メールの通知を受信しません。

インストールを行う前に

- 組織の Admin ロールが必要です。
- API のみのユーザは、Contrast の REST API にアクセスできますが、Contrast Web インターフェイスには口グインできません。
- SAML 認証の SSO(シングルサインオン)を使用するように組織が構成されている場合でも、API のみのユーザを作成できます。

手順

API のみのユーザを作成するには：

1. ユーザメニューから、**組織の設定**を選択します。
2. **ユーザ**を選択します。
3. **ユーザを追加**を選択します。

4. ユーザの名前と名字、メールアドレスを入力し、タイムゾーンを選択します。
5. **組織ロール**を選択します。
   参考：ベストプラクティスとして、組織ロールには **Edit** (955ページ)を選択して、最小の許容ロールを付与します。
   API のみのユーザに Admin 権限を付与することは推奨されません。
6. アプリケーションアクセスグループを選択します。
   参考：ベストプラクティスとして、アプリケーションアクセスグループには **View (953ページ)**または **Edit (953ページ)**を選択します。呼び出したい API エンドポイントや、GET(読み取り)や

POST(書き込み)を実行する場合によっては、API のみのユーザには View ではなく、より高い Edit 権限が必要になることがあります。

7. **API のみ**のチェックボックスを選択します。

> **注記**
> API のみのチェックボックスをオンにすると、アクセスのオプションがオンになっている場合は無効になります。API のみのユーザは、Contrast Web インターフェイスにアクセスできません。

8. **組織の設定 > ユーザ**で、新規に作成したユーザ名の横に **API のみ**のラベルがあることを確認します。

9. 特定のアプリケーションに対してアクセスを制限するためにアクセスグループを使用している場合は、API ユーザにアクセスさせたいアプリケーションのグループ (822ページ)に、API のみのユーザを追加します。ユーザの設定の**権限**でアクセスを確認してください。
10. API のみのユーザを使用するには、接続文字列を取得します。
    a. ユーザメニューから、**組織の設定**を選択します。
    b. **ユーザ**を選択します。
    c. ユーザ名の横にある API のみのラベルにカーソルを合わせ、表示されたサービスキーをコピーします。
    d. 以下の例のように、コマンドを使用して認証ヘッダを作成します。

```
echo -n '[email address of the API only account:Service Key]' | \
base64
```

使用状況分析の管理

Contrast では、より良い製品を開発するために使用状況のデータを収集しています。お客様のデータは追跡されず、全てののユーザデータは匿名化されて、集約された情報に対して分析が行われます。

組織管理者は、組織の使用状況データの収集を有効/無効にできます。

1. 組織の設定 (818ページ)で、**組織**を選択します。
2. **一般設定**で、トグルボタンを使用して**製品の使用状況分析**を有効/無効にします。デフォルトでは有効になっています。

診断サービスもシステムレベルで管理 (938ページ)できます。

制限付きの Edit ロール

脆弱性の削除やアプリケーションのアーカイブに関してユーザに制限を加えたい場合、制限付きの Edit ロールの設定を使用することができます。選択すると、この設定は組織の Edit ロールまたはアプリケーションの Edit ロールを持つ全てのユーザに適用されます。

注記
この設定は、オンプレミス版のお客様のみが使用できます。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **組織**を選択します。
3. 一般設定にて、**制限付きの Edit ロール**を選択します。
   選択すると、組織の Edit ロールまたはアプリケーションの Edit ロールを持つ全てのユーザは、脆弱性の削除とアプリケーションのアーカイブができなくなります。

組織レベルでのパスワードポリシーの設定

システムレベルで許可 (934ページ)されている場合は、組織管理者はパスワードポリシーを作成し、組織内のパスワードを規制することができます。

1. 組織の設定 (818ページ)で、**セキュリティ**を選択します。
2. ポリシーについて、以下の設定を入力します。
   - パスワード強度：**弱**、**中**、**強**、**複雑**または**カスタム**を選択できます。
     カスタムを選択した場合は、**最小限必要な大文字**、**小文字**、**数字**、**記号**の文字数を入力します。
   - 必要な文字数を**最小桁数**フィールドに入力します。
   - ドロップダウンを使用して、**パスワード期限**が切れるまでの時間を選択します。
   - **ログインのロックアウト**までのログイン試行回数を入力します。
   - **非アクティブアカウントの有効期限**が切れるまでの時間を選択します。
   - **パスワードの再利用を制限**のチェックボックスをオンにし、ドロップダウンを使用して、各パスワードを再利用できる回数を選択します。
   - **パスワードのリセットを制限**のチェックボックスをオンにし、ドロップダウンを使用して、リセットのリクエストの送信後にユーザがパスワードをリセットできる日数を選択します。
   - ドロップダウンを使用して、**アイドルタイムアウト**および**セッションタイムアウト**までの経過時間を選択します。
3. **保存**を選択します。

E メールドメインの制限

Contrast の情報を受信できる E メールアドレスのドメインを制限します。

1. 組織の設定で、**セキュリティ**を選択します。
2. **E メールドメイン**に、Contrast からの情報の受信を許可する E メールドメイン名をカンマ区切りで入力します。例えば、`yourbusiness.com` のように入力します。

3. **保存**を選択します。

IP 範囲の設定

Contrast アカウントにアクセスできる IP アドレスを制限します。これは、ブラウザと API の両方のアクセスに影響します。

1. 組織の設定 (818ページ)で、**セキュリティ**を選択します。
2. アクセスを許可する IP アドレスを入力します。追加アドレスを入力するには、**IP アドレスを登録**を選択します。CIDR 表記(10.0.0.0/24)を使用して、IP アドレスの範囲を指定することもできます。
3. **保存**を選択します。

組織レベルでの 2 段階認証の有効化

注記

Contrast で SSO を設定し、二要素認証も使用する場合は、Contrast ではなく、ID プロバイダ(idP)を使用して二要素認証を設定してください。Contrast で SSO を設定すると、ユーザを認証する責任は idP に渡されます。

2 段階認証を有効/無効にするには：

1. 組織の設定 (818ページ)で、左側のナビゲーションにある**セキュリティ**を選択します。
2. トグルボタンをオン(緑色)にして 2 段階認証を有効にします。
3. 2 段階認証が有効になり、ユーザは 2 段階認証の通知を受け取る方法を選択 (521ページ)できます。

注記

オンプレミス版のお客様の場合、これはシステムレベルで許可 (921ページ)されている必要があります。

ユーザが複数の組織に属している場合は、デフォルトの組織によって 2 段階認証の設定が決まります。

組織レベルでのシングルサインオン(SSO)の設定

オンプレミス版のお客様の場合は、シングルサインオンをシステムレベルで設定 (931ページ)できます。SaaS 版のお客様の場合は、Contrast Security が認証を設定します。ただし、組織管理者に組織の SSO を設定する権限が付与される場合があります。

注記

ユーザがメールアドレスではなくユーザ ID で識別されている場合、それらのアカウントは自動的に SSO 設定に移行されず、再作成する必要があります。

Contrast で SSO を設定し、二要素認証も使用する場合は、Contrast ではなく、ID プロバイダ(idP)を使用して二要素認証を設定してください。Contrast で SSO を設定すると、ユーザを認証する責任は idP に渡されます。

1. 組織の設定で**シングルサインオン**を選択して、**開始する**のリンクをクリックします。
2. 認証方法の変更による影響に関して、警告メッセージが表示される場合があります。警告をよく読んでから作業を進めてください。
3. 提供された情報を使用して、Contrast と IdP との設定を行います。
4. IdP の名前と、Contrast と接続するための関連メタデータを入力します。
5. Contrast にログインするための SAML リクエストがあった時に、新しいユーザアカウントを自動的に作成する場合は、**ユーザプロビジョニングを有効にする**のボックスにチェックします。
   - ドロップダウンを使用して、新規ユーザの**デフォルトの組織ロール**と**デフォルトのアプリケーションアクセスグループ**を選択します。
   - ユーザプロビジョニングをトリガーするのに必要な**承認されているドメイン**を追加します(例：contrastsecurity.com)。
6. **接続をテスト**ボタンをクリックして、設定をテストします。エラーが発生した場合は、トラブルシューティングのためのデバッグログが提供されます(このテストでは、メタデータの検証と、Contrast が IdP と接続できるかのみを検証します)。
7. テストが成功したら、**保存**を選択します。
8. **新しい**ブラウザ画面、プライベートブラウザのセッションまたはシークレットウィンドウを開いて、自分のアカウントで SSO ログインを試してください。ログインに失敗した場合は、ログイン状中のブラウザに戻り、組織の SSO を無効にしてから、Contrast 管理認証に戻すを選択します。

関連項目

Okta を使用してユーザとグループのプロビジョニングを設定する

グループにユーザを自動的に追加するように ADFS を構成する

組織レベルでのサーバのデフォルト設定

サーバの設定では、Contrast に追加する新しいサーバ(およびそのエージェント)に対するデフォルトの設定を定義します。これらの設定をカスタマイズして、各環境ごとに特定のデフォルトを設定することができます。

手順

1. 組織の設定 (818ページ)で、**サーバ**を選択します。
2. ドロップダウンを使用して、デフォルトにする環境(開発、QA、本番環境のいずれか)を選択します。サーバのデフォルトの環境に指定する場合、**デフォルト環境として設定する**の横にあるチェックボックスをオンにします。
3. ドロップダウンを使用して、 **ログレベル**を選択します。デフォルトのログレベル (899ページ)は、**ERROR** が選択されます。
4. **サーバクリーンアップを自動化**オプションでは、サーバがオフラインになってから自動的にクリーンアップされるまでの時間を指定します。デフォルトの値は、30 日です。
   バックグラウンドタスクが 5 分ごとに実行され、自動サーバクリーンアップが有効になっている組織があるかどうかチェックが行われます。
   設定された時間内にアクティビティが受信されなかったサーバが 1 つ以上ある場合、サーバは自動的に無効になります。無効になったサーバは、Contrast Web インターフェイスの**サーバ**画面に表示されなくなります。
   サーバが無効化された後も、そのサーバに関連する脆弱性や攻撃の情報は Contrast に保持されます。無効化されたサーバの Protect ライセンスは、ライセンスプールに戻されます。
5. **Assess** セクションでは、以下を設定します。
   a. キャプチャするスタックトレースを選択します。ALL(全て)、SOME(一部)、NONE(なし)から選択します。
   b. **パフォーマンス向上のためサンプリングを有効にする**を選択すると、解析のパフォーマンスが最適化されます。
      - Contrast で同じ URL が複数回呼び出されているのが確認された場合、「基準」に指定された回数に基づいて URL が解析されます。
      - その後、Contrast で同じ URL が引き続き確認された場合、「頻度」の設定に基づいてのみチェックが行われます。

- サンプルは、「サンプル保持」の設定で指定した秒数だけ保持されます。「サンプル保持」の設定で指定した時間が経過すると、「基準」の設定に従って再度 URL が解析されます。

以下を設定します。

- **基準**：サンプリングが完了するまでに、Contrast で URL を解析する回数。デフォルトの設定は、**5** です。
- **頻度**：基準のサンプル回数を取得後、毎回 N 番目のリクエストのみを解析します。頻度には、N の値を指定します。デフォルトの設定は、**10** です。
- **サンプル保持画面**：基準に戻る前に、Contrast でサンプルを保持する秒数。指定した秒数が経過すると、サンプリングはリセットされ基準サンプルが再度行われます。デフォルトの設定は、**180** です。

6. **Protect** セクションでは、以下を設定します。
   a. Protect を有効にするには、Protect トグルをオンにします。

> **重要**
> Protect を有効にすると、新しいサーバに Protect ライセンスを自動的に適用する設定が選択されます。
>
> 管理者は、サーバにライセンスが適用されるたびに E メールを受け取ります。サーバが頻繁にアップ/ダウンしたりする場合に、不要なトラフィックに対してメールのフィルタを設定するとよいです。

このセクションのライセンスバーには、購入済みの Protect ライセンスで使用中のライセンス数が表示されます。購入したライセンス数よりも多くのライセンスを使用している場合は、ライセンスバーには追加で使用中のライセンス数も表示されます。

   b. **ボットのブロックを有効にする**を選択すると、ボットのブロックがオンになります。
ボットをブロックすることで、スクレーパー、攻撃ツール、その他の自動化からの不要なトラフィックをブロックできます。
ブロックされたポッドのアクティビティを表示するには、**攻撃 >攻撃イベント**で、**自動化**フィルターを使用します。

> **注記**
> ボットのブロックは、各言語(Java、.NET Framework、.NET Core、Ruby、Python)のエージェントの YAML 設定ファイル (63ページ)で指定できます。

   c. **Syslog へ Protect イベントの出力を有効にする**を選択すると、Protect イベントが syslog に送信されます。
以下を設定します。
   - 所定のフィールドに **IP アドレス**と**ポート**を入力します。ドロップダウンを使用して**機能**を選択します。
   - 攻撃イベントの深刻度バッジをクリックして、ドロップダウンを使用し、Syslog の各メッセージの**重要度**を選択します。デフォルトは次の通りです。
     - 「攻撃検出済」は、**1 - Alert** になります。
     - 「ブロック済」は、**4 - Warning** になります。
     - 「探査検出」は、**5 - Notice** になります。

7. **ライブラリデータを保持**のトグルをオンにすると、ライブラリのデータを保持する機能が有効になります。この機能を有効にすると、サーバのクリーンアップ時に、Contrast から最後に削除されたサーバのライブラリデータが保持されます。
8. **エージェント診断**のトグルをオンにすると、エージェントのデータが Contrast に送信されます。Contrast では、このデータをルールやパフォーマンスの改善、製品改良の優先順位付けのために使用します。

組織レベルでのアプリケーションのデフォルト設定

組織レベルでアプリケーションのデフォルト設定を選択するには、以下の手順を使用します。

開始する前に

- 組織の Admin ロールが必要です。

手順

1. 組織の設定 (818ページ)で、**アプリケーション**を選択します。
2. ドロップダウンを使用して、アプリケーションの**重要性**レベルを選択します。**重大**、**高**、**中**、**低**、**重要でない**から選択できます。デフォルトでは、「中」が選択されます。
3. ポリシーのドロップダウンでは、自動的にアプリケーションに適用する修復ポリシーやコンプライアンスポリシー (807ページ)を選択します。
   デフォルトの設定に含まれないアプリケーションは、後からポリシーに追加することもできます。
4. 「アプリケーションのオンボード」では、次のいずれかまたは両方の設定を選択します。
   - アプリケーションメタデータがないアプリケーションはオンボードしない
   - セッションメタデータがないアプリケーションはオンボードしない

   選択した設定に応じて、アプリケーションに必要なメタデータが不足している場合、Contrast でそのアプリケーションの登録は失敗することになります。これらの設定により、エージェントの設定ファイルでメタデータの設定を強制することができます。
5. 「セッションメタデータ」では、デフォルトのセッションメタデータのフィルタリング (535ページ)を最新のセッションに設定する場合、**最新のセッションでアプリケーションの詳細をフィルタリング**を選択します。
   このフィルタの選択は、アプリケーションの「脆弱性」タブと「ルートカバレッジ」タブに影響があります。
6. ライセンスを自動で適用する場合、「動作」の欄で**新しいアプリケーションにライセンスを自動で適用**を選択します。
   ステータスバーには、使用可能なライセンス数のうち、使用されているライセンス数が表示されます。ライセンスのリンクを選択すると、組織のライセンスの内訳を参照することができます。
7. アプリケーションメタデータセクションを使用して、組織の各アプリケーションで利用するのに必要なアプリケーションメタデータを設定 (830ページ)します。
8. **保存**を選択します。

アプリケーションメタデータのフィールド作成

Contrast に新しいアプリケーションを登録するたびに、アプリケーションメタデータが収集されるように設定できます。

エージェントをインストールして設定する (46ページ)際に、作成したフィールドのメタデータを入力して、エージェントの設定ファイルに情報を追加するよう要求できます。メタデータは、**アプリケーション**ページの一覧に表示され、メタデータを使用してアプリケーションをフィルタすることができます。また、Contrast Web インターフェイスのアプリケーションの**詳細**ページにもメタデータは表示されます。

注記
アプリケーションメタデータのフィールドは、以下のエージェントバージョンでサポートされます。

- Java 3.5.6.591 以降
- .NET 18.10.35 以降
- Node 1.35.0
- Python 1.2.0
- Ruby 2.0.8

重要
アプリケーションメタデータに提供するデータは、エージェントの設定および YAML ファイルのダウンロードに必要です。このデータが指定されないと、ダウンロードは有効になりません。

手順

1. 組織の設定で、**アプリケーション**を選択します。
2. アプリケーションメタデータで、各フィールドに以下の項目を入力します。
   - **フィールドタイプ**：フリーフォーマット、数値、連絡先のいずれかを選択します。フィールドタイプにより、適切な検証が決定されます。
   - **名前**：このフィールドの名前を入力します。
     Contrast API と互換性を保つために、フィールド名には先頭を小文字にしたキャメルケース (camel-case)の表記を使用します。例えば、`BusinessID` や `Business ID` ではなく、`businessID` を使用します。
   - **値の条件**：チェックボックスを使用して、提供されるメタデータの値を**必須**か**ユニーク**にするかを指定します。
3. **フィールドを追加**を選択して、必要な行を追加します。
4. 各フィールドに情報を入力すると、フォーマットされたプロパティが表示されるので、エージェントの設定ファイルにコピー＆ペーストします。エージェントの設定ファイルには、key=value(キー=値)の形式でメタデータ情報を追加します。
5. 必須フィールドが含まれていないアプリケーションのデータを報告しないようにする場合は、**必須フィールドが足りないアプリケーションを制限する**のチェックボックスを選択します。これは、組織の新規のアプリケーション、もしくは新規と既存のアプリケーションに適用できます。
   このオプションを選択すると、アプリケーションで必須フィールドが指定されていない場合、Contrast Web インターフェイスで警告メッセージが表示されます。Contrast Web インターフェイスにはアプリケーションは表示されますが、エージェントからは、疎通済ルートや脆弱性などのデータは報告されません。
   アプリケーションを制限しない場合は、必須フィールドが指定されていないアプリケーションでも正常に登録され、エージェントからデータが報告されます。Contrast Web インターフェイスには、1 つ以上のフィールドが足りないという警告メッセージが表示されます。

組織レベルでの通知の管理

通知により、ユーザは脆弱性の検出やアプリケーションへの攻撃など、特定の状況に関するアラートを受信することができます。組織管理者は、組織内の全てのユーザに対して Contrast の通知のデフォルトを設定 (832ページ)できます。個々のユーザは、各種の通知を受け取る方法を選択 (523ページ)できます。

通知には主要なチャネルが 2 つあります。

- **Contrast 内**：直接 Contrast Web インターフェイス内に通知されます。通知を確認するには、トップメニューバーにある**通知**アイコンを選択します。
- **E メール**： 適切な SMTP システムと通信して通知を E メールで受信するよう、システム管理者が Contrast を設定 (943ページ)します。

注記
Contrast では、ユーザ通知が必要な一部の機能について Contrast 管理者がその機能を有効にすると、影響を受けるユーザに自動的に通知が送信されます(この通知は**通知**ページでは制御できません)。Contrast では、脆弱性ステータスの承認 (868ページ)やその他の**ポリシーの管理**の設定などの機能について、ユーザと管理者に通知する必要があります。

管理者用通知の設定

管理者は、Contrast アプリケーション内と E メールで、組織全体のイベントに関する以下の通知を自動的に受信します。

- **アプリケーションライセンスの付与**： Contrast で新しいアプリケーションにライセンスが付与 (820ページ)された。
- **アプリケーションライセンス期限切れ間近**： 有効なアプリケーションのライセンスの期限が切れる(ライセンス有効期限の 2 か月前、1 か月前、1 週間前にこの通知が送信)。
- **ライセンス期限切れ間近**: 関連付けられたアプリケーションの無い既存のライセンスの期限が切れる(ライセンス有効期限の 2 か月前、1 か月前、1 週間前にこの通知が送信)。
- **修復ポリシーの違反**： 脆弱性が既存の修復ポリシー (786ページ)に違反している。
- **ライブラリポリシーの違反**： ライブラリが既存のライブラリポリシー (810ページ)に違反している。
- **コンプライアンスポリシーの通知**： アプリケーションが既存のコンプライアンスポリシー (807ページ)に違反している。

アプリケーションおよび組織レベルの Admin および RulesAdmin ユーザは、ポリシー違反の通知を受信する必要があります。通知を管理 (523ページ)することで、通知件数を最小限にしたり、1 つのメールに通知をまとめたりすることができます。

また、ユーザが特定の脆弱性をクローズするようクエストした時に、承認を得るための通知を受信することができます。これは、組織管理者が設定 (868ページ)する必要があります。

ユーザのデフォルトの通知設定

組織管理者は、組織内の全てのユーザを対象に、インテグレーションおよび Contrast 内/E メールでの通知に関するデフォルトの通知設定を定義できます。個々のユーザは、各種の通知を受け取る方法を選択 (523ページ)できます。

組織でのデフォルトの通知設定を定義するには：

1. 組織の設定 (818ページ)で、**通知**を選択します。
2. トグルボタンを使用して、左側に表示されている通知登録を有効/無効にします。
   - **積極的な攻撃**：Protect が有効なアプリケーションで積極的な攻撃が発生。
   - **新しい脆弱性**： Contrast で新たに脆弱性を検出。フィールドをクリックして、特定の深刻度レベルや「ライブラリ(Library)」を指定します。デフォルトでは、「全て」が選択されます。
   - **サーバのオフライン**：サーバへのアクセス不可。
   - **新しいコメント**： チームメンバーが検出結果についてコメントを投稿。

- **新しいアセット:** アクセス権限のある新しいアセットが追加。フィールドをクリックすると、「アプリケーション」か「サーバ」のいずれかに通知を設定できます。デフォルトでは「全て」が選択されます。
- **E メールダイジェスト：** Contrast の毎日のアクティビティをまとめたものです(E メールのみ)。

> **注記**
> 特定のインテグレーションの通知登録を有効にするには、**インテグレーションを追加**を選択してインテグレーションを追加します。または、インテグレーション列の上部にあるドロップダウンから既存のインテグレーションを選択します。

カスタム通知の作成

組織の管理者は、通知のデフォルト設定を指定したり、カスタム通知を作成したりすることができます。

カスタム通知を作成するには：

組織の設定 > 通知 > カスタム通知にアクセスし、**通知を作成**を選択します。表示される画面で、次のフィールドに入力します。

1. ラジオボタンを使用して、**脆弱性**か**攻撃**を選択します。
2. 通知の**名前**を入力します。
3. ドロップダウンを使用して、通知の**頻度**を、またはに設定します。
4. 通知の目的などを**説明**に入力します。
5. フィールドをクリックして(複数選択可)、この通知が適用される**アプリケーション**を選択します。
6. この通知に適用する**アプリケーションのタグ**を選択します。
7. 通知を受信する組織の**ユーザ**を選択します。
8. ドロップダウンを使用して、**条件**を選択します。
   条件を追加するには、**条件を追加**をクリックします。
   Contrast のカスタム通知には、以下の条件を指定できます。

通知タイプ	条件	説明
カテゴリ (Category)	等しい(=)、等しくない(≠)	カテゴリとは、認証やインジェクション、暗号化などのルールタイプに関する上位レベルのグループです。Contrast のルールタイプには、11 のカテゴリがあります。
影響度(Impact)	等しい(=)、より低い、より高い	影響度とは、ルールタイプが特定の組織にどのように影響するかに基づいて、高、中、低の評価で測定されます。全てのルールタイプには、デフォルトの影響度が設定されていますが、カスタマイズできます。
可能性 (Likelihood)	等しい(=)、より低い、より高い	可能性とは、ルールタイプが発生する頻度に基づいて、高、中、低の評価で測定されます。すべてのルールタイプには、デフォルトの可能性が設定されていますが、カスタマイズできます。
URL	等しい(=)、含む、〜で始まる	アプリケーションの特定の URL です。
クラス (Class)	等しい(=)、含む、〜で始まる	Java または.NET の特定のクラスです。
メソッド (Method)	等しい(=)、含む、〜で始まる	Java または.NET の特定のメソッドです。

> **重要**
> 複数の条件を選択した場合、Contrast は AND ロジックで通知を行います。選択した条件が全て該当した場合に通知が生成されます。

組織レベルでのスコア設定のカスタマイズ

Contrast ではアプリケーションのスコア (958ページ)が算出されますが、必要に応じてライブラリのスコア (610ページ)に依存させることができます。 組織レベルでスコア設定をカスタマイズするには：

1. 組織の設定 (818ページ)で、**スコアの設定**を選択します。
2. **総合スコア**では、この組織内のアプリケーションをどのようにスコア付けするかを選択します。
   - **デフォルトの評価方法**は、アプリケーションのライブラリスコアとカスタムコードのスコアの平均です。
   - **カスタムコードのみを評価対象とする**場合は、アプリケーション全体のスコア計算にライブラリのスコアが含まれなくなります。このオプションを選択した場合、特定の言語を選択するか、全ての言語に適用するかを、クリックして選択します。
3. **ライブラリのスコア**で、アプリケーション内のライブラリをどのようにスコア付けするかを選択します。
   - **デフォルトの評価方法**では、ライブラリの脆弱性だけでなく、ライブラリの古さやバージョン管理を含むアルゴリズムが使用されます。
   - **脆弱性のみを評価対象とする**方法では、ライブラリに存在する脆弱性のみに基づいてスコア付けされます。
4. **保存**を選択します。

ヒント

ルール管理者(Rules Admin)は、**ポリシーの管理**でポリシーの設定を行い、ポリシーに違反するライブラリを自動的に低いスコア(F)にすることができます。この設定が選択されると、スコアの設定に警告メッセージが表示されます。警告にあるポリシーのリンクをクリックすると、ライブラリポリシーページに移動します。ここで、管理者はそれらの設定を確認して更新ができます。

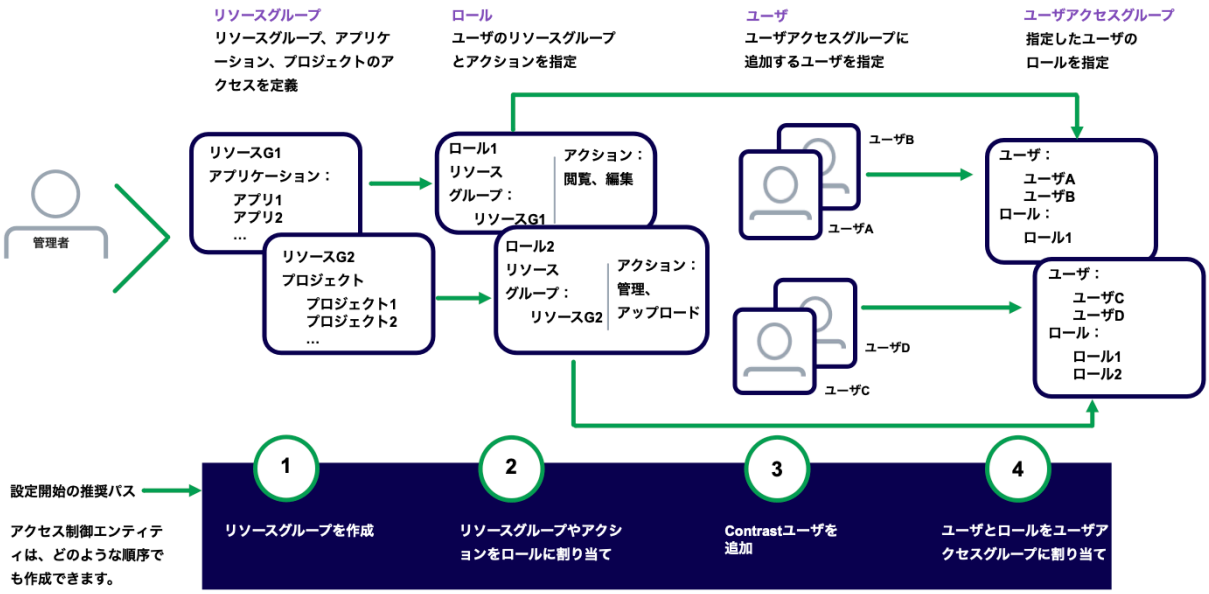
ロールベースのアクセス制御

Contrast でアプリケーション、プロジェクトおよび組織の設定へのユーザアクセスを管理するには、リソースグループ、ロール、およびユーザアクセスグループを設定します。

注記

この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。



アクセス制御の設定

アクセス制御には、以下の設定が必要です。

- **ユーザ： (841ページ)** Contrast ユーザのリストで、ユーザアクセスグループに追加できます。
- **リソースグループ： (848ページ)**リソースグループとは、割り当てるロールに従って、ユーザがアクセスできるアプリケーション、データ、設定、スキャンプロジェクト、および組織の設定を決定するものです。
  組込みのリソースグループを使用することも、カスタムグループを作成することもできます。
- **ロール： (853ページ)**ロールとは、ユーザに実行権限がある操作を定義するものです。
  組込みのロールを使用することも、カスタムロールを作成することもできます。
  組込みのロールである、組織 Admin を使用すると、全てのユーザアクセスグループ、ロールおよびリソースグループを管理できます。
- **ユーザアクセスグループ： (862ページ)**ユーザアクセスグループとは、ユーザにロールを割り当てるものです。
  組込みのユーザアクセスグループを使用することも、カスタムのユーザアクセスグループを作成することもできます。

アクセス制御の管理方法

アクセス制御を管理するには、以下のいずれかの方法を使用します。

- **Contrast Web インターフェイス**
  Contrast Web インターフェイスを使用すると、アクセス制御の設定と管理を視覚的に行えます。これを利用するには、組織の管理者であるか組織 Admin ロールが必要です。
- **アクセス制御 API**
  アクセス制御の管理を自動化したい場合や、既存のシステムに統合したい場合に、アクセス制御 API を使用すると便利です。この方法には、PLATFORM_ORG_MANAGE 権限が必要です。
  詳しくは、アクセス制御 API をご覧ください。

命名規則と条件

リソースグループ、ロール、およびユーザアクセスグループの名称は、各組織で一意にする必要があります。例えば、**MyAdmins** という名前のロールを 2 つ作成することはできません。

カスタムのロールやグループを作成する前に、すべてのカスタム設定の命名規則を定義することを推奨します。命名規則を定義しておくことで、多数のカスタム設定の管理が容易になります。1 つの方法と

して、リソースグループの名前を、そのグループに関連付けるロールとマップさせる方法があります。例えば、**E コマース**というリソースグループを作成する場合、**E コマースの開発**というロール名などは適切でしょう。この場合、このロールのあるユーザは、E コマースのアプリケーションに携わる開発者と定義することができます。

すべてのカスタム設定の命名条件

- 名前には、最大 50 文字までの英数字を使用できます。
- 説明文には、最大 1,024 文字まで使用できます。
- 名前と説明文には、特殊文字やスペースを使用できます。
- 名前の大文字と小文字は区別されません。
  例えば、「ecommerce developers」と「Ecommerce Developers」は同じ名前とみなされます。

アクションと権限

ロールに割り当てるアクションには、特定のタスクを実行し、データにアクセスするための権限が与えられます。

> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

組織のアクションと権限

アクション：	含まれる権限：	属する組込みロール：
組織の閲覧	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 **レポート** • アプリケーションのレポート作成	組織 View アプリとプロジェクト Admin および組織 View アプリとプロジェクト Admin
組織の編集	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • スコアの設定の表示 • スコアの設定の表示 • 組織の設定の表示 • 新規登録ボタンへのアクセス **セキュリティ** • エージェントキーの取得 **レポート** • レポートの設定の表示 • アプリケーションのレポート作成	組織 Edit

アクション：	含まれる権限：	属する組込みロール：
組織のルールの管理	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • スコアの設定の表示 • 組織の設定の表示 • Protect の有効化/無効化 • 新規登録ボタンへのアクセス **ポリシー** • スコア設定のポリシーの管理 **レポート** • レポートの設定の表示 • アプリケーションのレポート作成	組織 RulesAdmin
組織の管理	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • 組織の設定の表示 • 新規登録ボタンへのアクセス **インテグレーション** • インテグレーション情報の表示 • インテグレーション設定の編集 **サーバ** • サーバライセンスの管理 **ユーザ** • ユーザの表示 • ユーザの作成 • ユーザの削除 • ユーザの設定の編集 **ポリシー** • スコア設定のポリシーの管理 **レポート** • レポートの設定の表示 • レポートの設定の変更 • アプリケーションのレポート作成 **セキュリティ** • API キーのローテーション • 監査ログの表示 • IP の制限の管理 • パスワードポリシーの管理 • セッションタイムアウトの管理 • 2 段階認証· SAML の管理 **Protect** • Protect の有効化/無効化	組織 Admin アプリとプロジェクト Admin
プラットフォーム組織の管理	**アクセス制御** • ユーザアクセスグループの管理 • ロールの管理 • リソースグループの管理 **レポート** • アプリケーションのレポート作成	アプリとプロジェクト Admin 組織 Admin

アプリケーションのアクションと権限

アクション:	含まれる権限：	属する組込みロール：
アプリケーションの閲覧	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • スコアの設定のカスタマイズ • レポートの設定の変更 • 通知の設定の変更 **脆弱性** • 脆弱性の詳細の表示 • エクスポート(脆弱性、トレース、ルート) • HTTP リクエストの再実行 **アプリケーション** • アプリケーションの詳細の表示 • タグの表示 **ポリシー** • アプリケーションポリシーの表示 • アプリケーションの例外の表示 **サーバ** • サーバの詳細の表示 **レポート** • レポートの生成 **ライブラリ** • ライブラリの詳細の表示 • タグの表示 • マニフェスト情報の表示 • ライブラリのエクスポート	組織 View アプリとプロジェクト Admin アプリとプロジェクト Admin および組織 View アプリ View アプリ Admin アプリ Edit アプリ RulesAdmin
アプリケーションの編集	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • 組織の設定の表示 • スコアの設定のカスタマイズ • 通知の設定の変更 • 新規登録ボタンへのアクセス **脆弱性** • 脆弱性の詳細の表示 • バグ管理システムへの脆弱性の送信 • 脆弱性のマージ • 脆弱性の設定の編集 • 履歴の管理 • 脆弱性の削除 • エクスポート(脆弱性、トレース、ルート) • HTTP リクエストの再実行 **アプリケーション** • アプリケーションの詳細の表示 • タグの表示 • アプリケーションのマージ • アプリケーションのアーカイブ • アプリケーションの復元 • タグの編集と削除 • フィルタの管理 • トレースの管理 • バグ管理システムの設定の管理 **ポリシー** • アプリケーションポリシーの表示 • アプリケーションの例外の表示 • スコア設定のポリシーの管理 **サーバ** • サーバの詳細の表示 • タグの編集と削除 • サーバの設定の編集 • バグ管理システムの設定の管理 **レポート** • レポートの生成 • レポートの設定の変更 **ライブラリ** • ライブラリの詳細の表示 • タグの表示 • タグの編集と削除 • マニフェスト情報の表示 • ライブラリのエクスポート **Protect** • Protect の有効化/無効化	アプリ Admin アプリ Edit アプリとプロジェクト Admin および組織 View
アプリケーションのルールの管理	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 **アーキテクチャ** • アーキテクチャ情報の表示 **ポリシー** • Assess ルールの設定の管理 • ライブラリポリシーの管理 • 修復ポリシーの管理 **Protect** • Protect の有効化/無効化	アプリ RulesAdmin アプリとプロジェクト Admin および組織 View アプリ Admin

アクション:	含まれる権限：	属する組込みロール：
アプリケーションの管理	**全般** • Contrast Web インターフェイスへのログイン • ユーザプロファイルの表示 • 通知の表示 • 組織の設定の表示 • スコアの設定の表示 • スコアの設定のカスタマイズ • レポートの設定の変更 • 通知の設定の変更 • 新規登録ボタンへのアクセス **ポリシー** • アプリケーションポリシーの表示 • アプリケーションの例外の表示 • ジョブ結果ポリシーの管理 • Assess ルールの設定の管理 • ライブラリポリシーの管理 • 修復ポリシーの管理 • スコア設定のポリシーの管理 **アプリケーション** • アプリケーションの詳細の表示 • タグの表示 • タグの編集と削除 • アプリケーションのマージ • アプリケーションのアーカイブ • アプリケーションの復元 • フィルタの管理 • トレースの管理 • バグ管理システムの設定の管理 • アプリケーションへのライセンス付与 • アプリケーションの設定の編集 • アプリケーションのリセット • アプリケーションの削除 **アーキテクチャ** • アーキテクチャ情報の表示 **サーバ** • サーバの詳細の表示 • タグの編集と削除 • サーバの設定の編集 • サーバの削除 • バグ管理システムの設定の管理 • サーバライセンスの管理 **Protect** • Protect の有効化/無効化 **インテグレーション** • インテグレーション情報の表示 • インテグレーション設定の編集 **レポート** • レポートの生成 • レポートの設定の表示 **ライブラリ** • ライブラリの詳細の表示 • タグの表示 • タグの編集と削除 • マニフェスト情報の表示 • ライブラリのエクスポート **ユーザ** • ユーザアクセスグループの管理 • ロールの管理 • リソースグループの管理 **脆弱性** • 脆弱性の詳細の表示 • バグ管理システムへの脆弱性の送信 • 脆弱性のマージ • 脆弱性の設定の編集 • 履歴の管理 • 脆弱性の削除 • エクスポート(脆弱性、トレース、ルート) • HTTP リクエストの再実行	アプリ Admin

プロジェクトのアクションと権限

アクション：	含まれる権限：	属する組込みロール：
プロジェクトの閲覧	**スキャン** • スキャンプロジェクトとスキャンに関する詳細の表示 • スキャン結果のダウンロード(SARIF か CSV ファイル)	アプリとプロジェクト Admin アプリとプロジェクト Admin および組織 View プロジェクト View プロジェクト Admin
スキャンのアップロード	**スキャン** • スキャンするファイルのアップロード • スキャンの開始 • スキャンプロジェクトとスキャンに関する詳細の表示 • スキャン結果のダウンロード(SARIF か CSV ファイル)	スキャン Upload

アクション：	含まれる権限：	属する組込みロール：
プロジェクトの閲覧、編集、削除	**スキャン** • スキャンプロジェクトのアーカイブ • スキャンプロジェクトの削除 • プロジェクトの設定の編集 • スキャンするファイルのアップロード • スキャンの開始 • スキャンプロジェクトとスキャンに関する詳細の表示 • スキャン結果のダウンロード(SARIF か CSV ファイル)	アプリとプロジェクト Admin および組織 View プロジェクト Admin
プロジェクトの作成	**スキャン** • スキャンプロジェクトの作成	プロジェクト Admin
プロジェクトの編集	**スキャン** • スキャンプロジェクトの設定の編集 • スキャンするファイルのアップロード • スキャンプロジェクトとスキャンに関する詳細の表示 • スキャン結果のダウンロード(SARIF か CSV ファイル)	プロジェクト Admin
プロジェクトの削除	**スキャン** • スキャンプロジェクトとスキャンに関する詳細の表示 • スキャン結果のダウンロード(SARIF か CSV ファイル) • スキャンプロジェクトの削除	プロジェクト Admin

Protect のアクションと権限

アクション：	含まれる権限：	属する組込みのリソースグループ：
Protect データのアクセス	**Protect** • Protect データの表示	Protect データ View
Protect の例外管理	**Protect** • Protect ルールに対するアプリケーションの例外の表示 • Protect ルールに対するアプリケーションの例外の作成と編集	Protect 例外 Admin
Protect のポリシー管理	**Protect** • Protect ポリシーの表示と編集 • ログエンハンサーの追加と編集 • 攻撃アラートの管理 • IP 拒否リストの管理 • Protect ライセンスの管理 • Protect ルールの設定の管理	Protect ポリシー Admin
Protect の機密データ管理	**Protect** • 機密データポリシーの詳細の表示 • 機密データポリシーの編集	Protect 機密データ Admin

サーバレスのアクションと権限

アクション：	含まれる権限：	属する組込みロール：
サーバレスデータの閲覧	**サーバレス** • サーバレスデータの表示 • API エンドポイントとのやり取り	サーバレス View

組織のユーザとグループからの移行

2023 年 8 月 14 日より、Contrast ではユーザアクセスをより詳細なレベルで管理できる新しいアクセス制御設定を追加しました。

既存のユーザとアプリケーショングループは、新しいアクセス制御に自動的に移行されます。

注記
この機能はホストされているお客様のみにサポートされており、プレビューモードです。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

移行後の流れ
ユーザは以下のように移行されます。

- ユーザの組織のロールに基づいて、同等のロールが付与されている組込みのユーザアクセスグループに移行されます。
  移行完了後、ユーザのユーザアクセスグループを変更できます。
- アプリケーションの権限は新しいアクセス制御に移行されません。
  ユーザ (845ページ)が、必要なアプリケーションへのアクセスが付与されているユーザアクセスグループに属していることを確認してください。
  ユーザアクセスグループには、ユーザのロールが指定されます。ユーザのロールのリソースグループには、特定のアプリケーションへの権限が含まれます。

ユーザ
ユーザリストにユーザを追加すると、そのユーザを特定のユーザアクセスグループに割り当てることができま す。

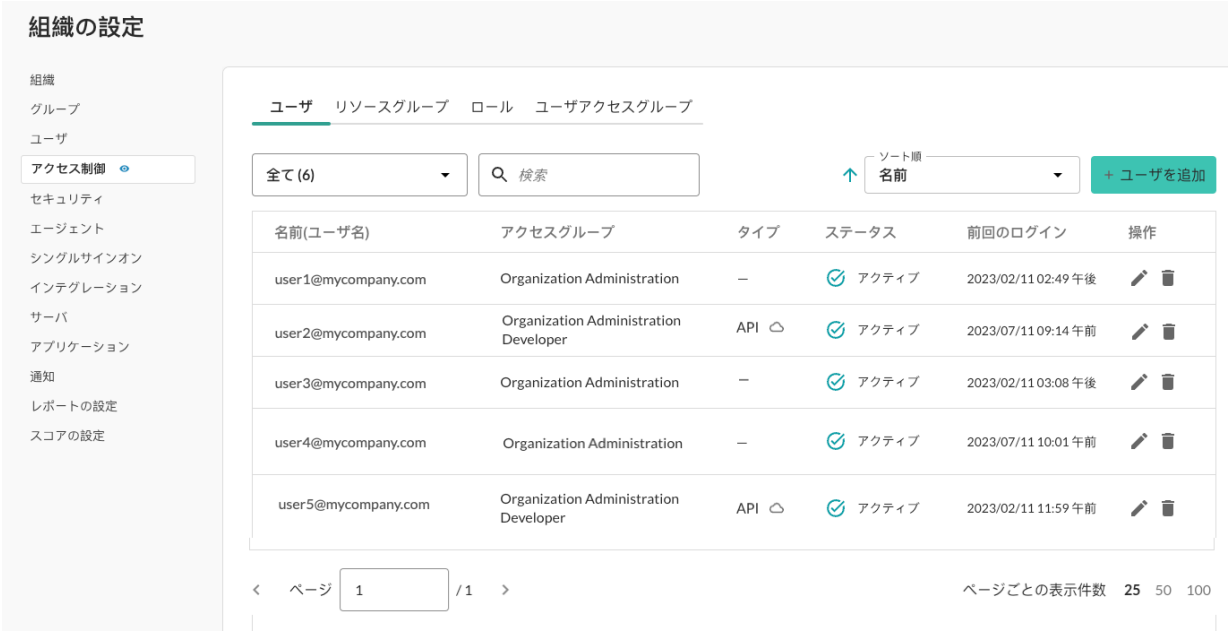
注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとグループ (822ページ)を設定して、Contrast へのアクセスを管理します。

ユーザタブ
ユーザタブには、Contrast ユーザの詳細が表示されます。

- **名前**： ユーザの名字と名前。
- **アクセスグループ**： ユーザに割り当てられたアクセスグループ。
  ユーザアクセスグループにより、ユーザに割り当てられるロールが決まります。ロールにより、ユーザの操作やリソースが決まります。
- **タイプ**： アカウントの種類(ユーザ、API、ゲスト)
  API ユーザの認証情報を取得するには、タイプの横にある雲のアイコン()を選択します。そして、認証情報の横にあるコピーアイコン()を選択します。
  ゲストアカウントを標準アカウントに変更するには、タイプ列で**ゲスト**を選択して、**ユーザを追加**を選択します。
- **ステータス**： ユーザアカウントの現在のステータス。
  - **アクティブ**： ユーザは、Contrast のデータにアクセスできます。
  - **アクティブ化待ち**： ユーザは、Contrast のデータにアクセスする前に、Contrast からのアクティベーションメールに返信する必要があります。

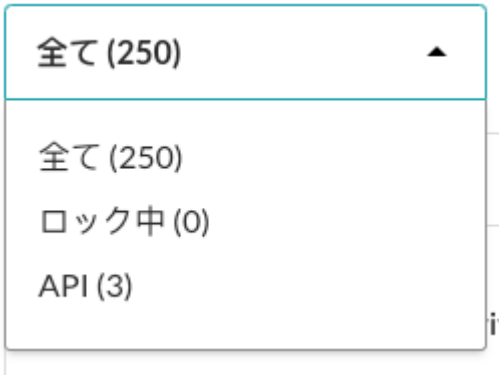
- **非アクティブ**： アカウントは、現在使用されていません。
  アカウントを再度アクティブにするには、アクティブ化アイコン(⃠)を選択し、 **アクティブ化**を選択します。
- **ロック中**： セキュリティポリシーに基づき、ユーザのアカウントがロックされています。
  アカウントのロックを解除するには、その横にあるチェックマークを選択し**ロック解除**を選択します。
- **前回のログイン**： ユーザが Contrast に最後にログインした日付と時刻。



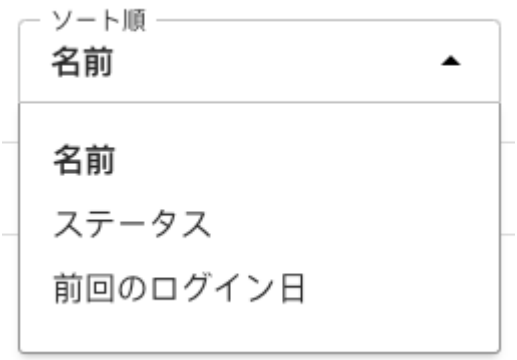
フィルタとソート

以下のフィルタを選択することで、表示を絞り込むことができます。

- **全て**： 全てのユーザを表示します。
- **ロック中**： アカウントがロックされているユーザを表示します。
- **API**： API アクセスのみのユーザを表示します。



名前、ステータス、前回のログイン日でもソートできます。



操作

このタブでは、次のことを行えます。

- ユーザを作成 (843ページ)してユーザアクセスグループを割り当てる
- ユーザの情報を編集 (845ページ)する
- ユーザを削除 (846ページ)する

ユーザの追加

以下の手順でユーザを追加します。

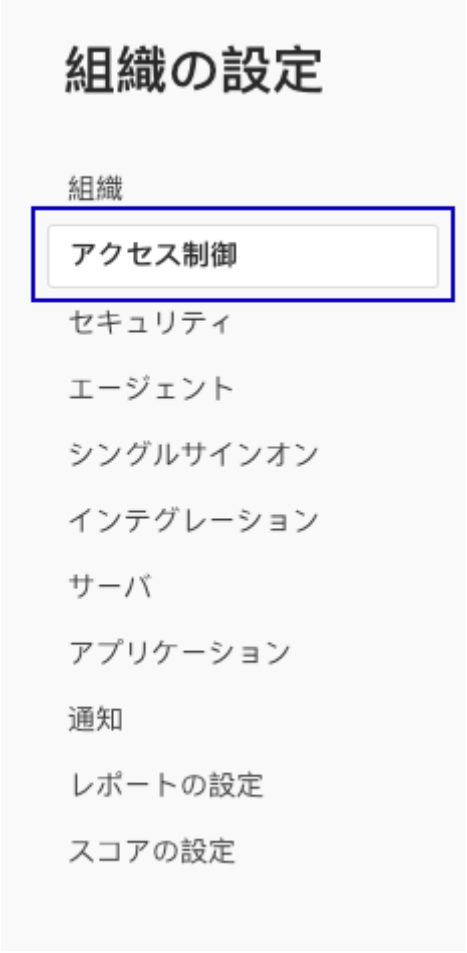
> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

開始する前に

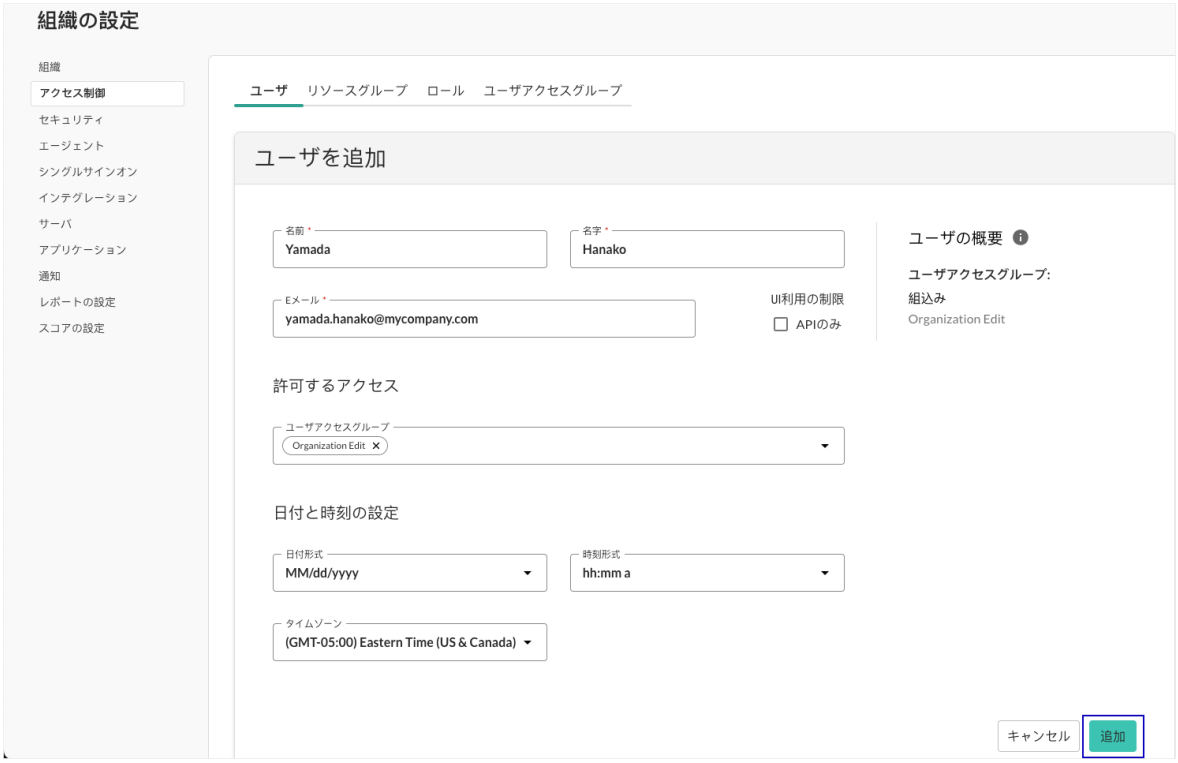
- **オンプレミス版のユーザ：** 組織レベルでのユーザの追加/編集 (822ページ)の手順を使用してください。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ユーザ**タブを選択します。
4. **ユーザを追加**を選択します。
5. ユーザの詳細を入力します。



- **名前**：ユーザの名前を入力します。
- **名字**： ユーザの名字を入力します。
- **E メール**： ユーザの E メールアドレスを入力します。
- **UI 利用の制限**： ユーザに Contrast API へのアクセスを許可し、Web インターフェイスにはアクセスできないようにする場合は、「API のみ」のオプションを選択します。
  このオプションを選択すると、ユーザはパスワードを使用して Contrast Web インターフェイスにログインできなくなります。この設定を解除すれば、ユーザはパスワードでログインできるようになり、API も引き続き使用できるようになります。
- **許可するアクセス**：組込みまたはカスタムのユーザアクセスグループを選択します。
  アクセスグループは、ユーザが利用できる操作やリソースを決定します。
- **日付と時刻の設定**： ユーザの日付形式、時刻形式、タイムゾーンを選択します。

6. **追加**を選択します。

ユーザの編集

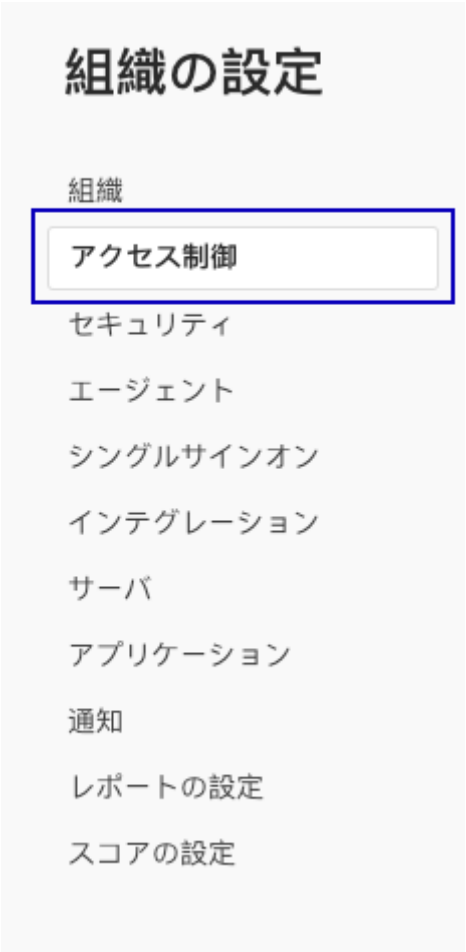
以下の手順でユーザを編集します。

[en]

> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ユーザ**タブを選択します。
4. 情報を変更したいユーザの行の末尾にある**編集**アイコン(✎)を選択します。
5. 必要に応じて設定を変更したら、**保存**を選択します。
   ユーザの E メールアドレスを変更するには、新しいユーザを追加 (843ページ)し、新しいアドレスを入力してください。

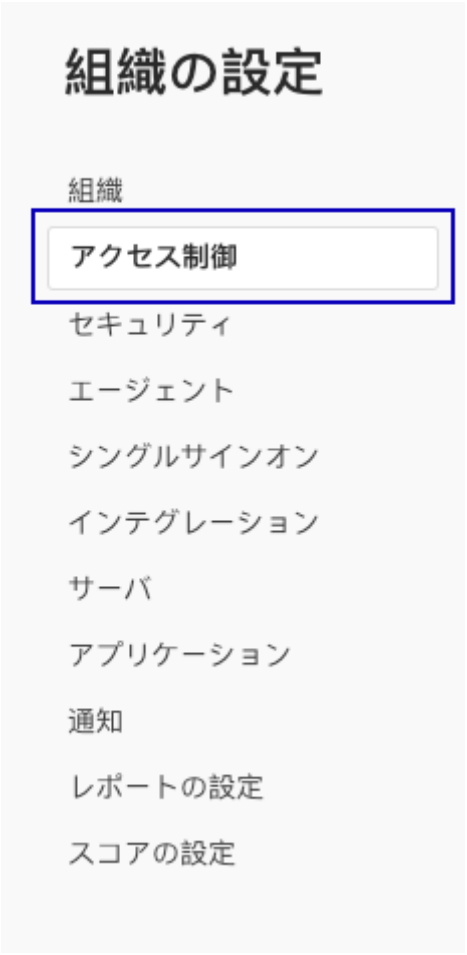
ユーザの削除

以下の手順でユーザを削除します。

> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ユーザ**タブを選択します。
4. 削除したいユーザの行の末尾にある**削除**アイコン(🗑)を選択します。
5. 「ユーザを削除」画面で、**削除**を選択します。

API のみのユーザの作成

全てのプラグインやインテグレーションで使用できる、API ユーザアカウントを作成します。

> **注記**
> この手順は、SaaS 版のお客様用です。
>
> オンプレミス版をご利用のお客様は、こちらの API ユーザの作成 (824ページ)手順を使用して下さい。

参考： ベストプラクティスとして、プラグインやインテグレーションを使用するためだけのユーザアカウントを登録することを推奨します。この専用ユーザにより、あるユーザが会社や組織を離れた際に、使用しているプラグインやインテグレーションが機能するよう継続できます。そのユーザアカウントが削除されて問題が発生するような状況を避けることができます。

API のみのアカウントは、通知設定が有効になっている場合でも、電子メールの通知を受信しません。

開始する前に

- API ユーザは、Contrast の REST API にアクセスできますが、Contrast Web インターフェイスには口グインできません。
- SAML 認証の SSO(シングルサインオン)を使用するように組織が構成されている場合でも、API ユーザを作成できます。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。
3. **ユーザ**タブを選択します。
4. **ユーザを追加**を選択します。

5. ユーザの名前と名字、E メールを入力します。
6. **API のみ**オプションを選択します。

7. **追加**を選択します。
8. ユーザの一覧で、タイプ列に API ラベルがある新しいユーザが表示されていることを確認します。

9. API ユーザを使用するには、接続文字列を取得します。
   a. ユーザの一覧で、タイプ列にある API アイコンを選択したら、サービスキーと認証ヘッダーをコピーします。
   b. Contrast API を使用する場合に、これらの認証情報を使用して下さい。

リソースグループ

リソースグループによって、割り当てたロールに基づいて、ユーザがアクセスできるアプリケーション、プロジェクト、および組織の設定を指定することができます。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

リソースグループタブ

「リソースグループ」タブには、既存のグループの一覧が表示されます。このタブから、次のことを行えます。

- リソースグループの一覧を表示
  検索を使用して、特定のグループを見つけることができます。
- リソースグループの管理
  - リソースグループを追加 (849ページ)

- リソースグループを編集 (851ページ)
- リソースグループを削除 (852ページ)

組込みのリソースグループ

以下の組込みのリソースグループを、アクセス制御のロールに選択できます。

- **全てのアプリケーション**： 組織内の全てのアプリケーションにアクセスできます。
- **全ての Protect 例外**： Protect の全ての例外設定にアクセスできます。
- **全ての Protect 機密データ**： 全ての機密データポリシーにアクセスできます。
- **全ての関数**： 全てのサーバレス関数とデータエンドポイントにアクセスできます。
- **全てのプロジェクト**： 組織内の全てのスキャンプロジェクトにアクセスできます。
- **全てのロール**： 組織内の全てのロールのみにアクセスできます。
- **全てのアクセス制御**： ユーザ、ロール、ユーザアクセスグループ、リソースグループの全ての設定にアクセスできます。
- **全ての組織の設定**： 組織の設定の全てにアクセスできます。これには、全てのユーザアクセスグループ、リソースグループ、ロールの管理が含まれます。
- **全てのユーザアクセスグループ**： 全てのユーザアクセスグループにアクセスできます。
- **全てのリソースグループ**： 全てのリソースグループにアクセスできます。

注記

組込みのリソースグループの設定を変更することはできません。**表示**アイコン(👁)を選択すると、組込みのリソースグループの設定を参照することができます。

リソースグループの追加

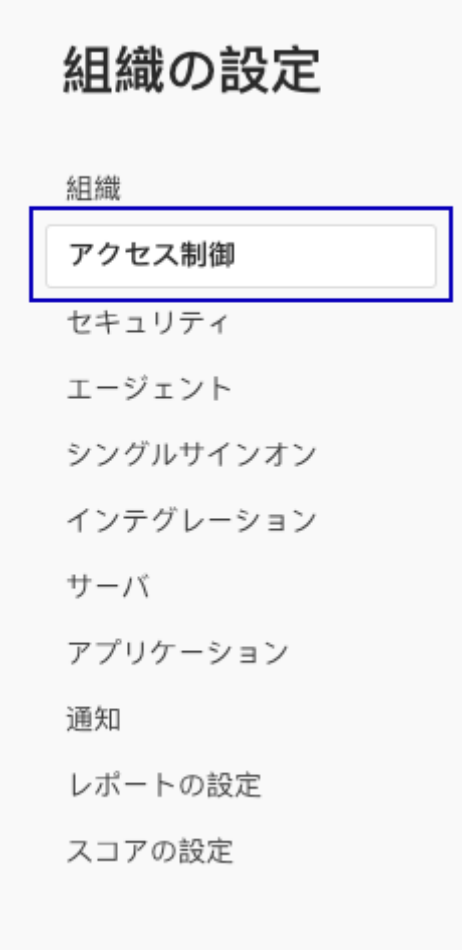
カスタムリソースグループによって、ユーザがアクセスできるアプリケーション、プロジェクト、および組織の設定を指定することができます。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **リソースグループ**タブを選択します。
4. **グループを追加**を選択します。

5. 「リソースグループを追加」の画面で、以下の設定を行います。

- **リソースグループ名**：リソースグループの名前を入力します。
  名前は、組織内で一意である必要があります。スペースや特殊文字を含め、最大 255 文字まで使用できます。
- **リソースグループの説明**：わかりやすい説明を明記します。
  このリソースグループの目的が分かるような説明を入力してください。スペースや特殊文字を含め、最大 1,024 文字まで使用できます。
- **リソースグループ**：このグループに含めるリソースグループを選択します。
  利用可能なリソースグループのリストには、組織内に存在する全てのリソースグループが表示されない場合があります。ある一つのリソースグループが別のリソースグループの一部である場合は、この単独のリソースグループを選択することはできません。ただし、その単独のリソースグループを含むリソースグループを選択することはできます。例：
  - １を作成して `MyResource` グループを含めます。
  - ２を作る際に、リストには `MyResource` グループは表示されません。１は表示され、これを２に追加することはできます。

  リソースグループを選択した場合に、個々のプロジェクトやアプリケーションを選択することはできません。
- **プロジェクト**：このグループに含めるスキャンプロジェクトを個々に選択します。
  個々のスキャンプロジェクトを選択した場合、リソースグループを選択することはできません。
- **アプリケーション**：このグループに含めるアプリケーションを個々に選択します。
  個々のアプリケーションを選択した場合、リソースグループを選択することはできません。

6. **追加**を選択します。
   これで、このリソースグループをロールに割り当てることができます。

リソースグループの編集

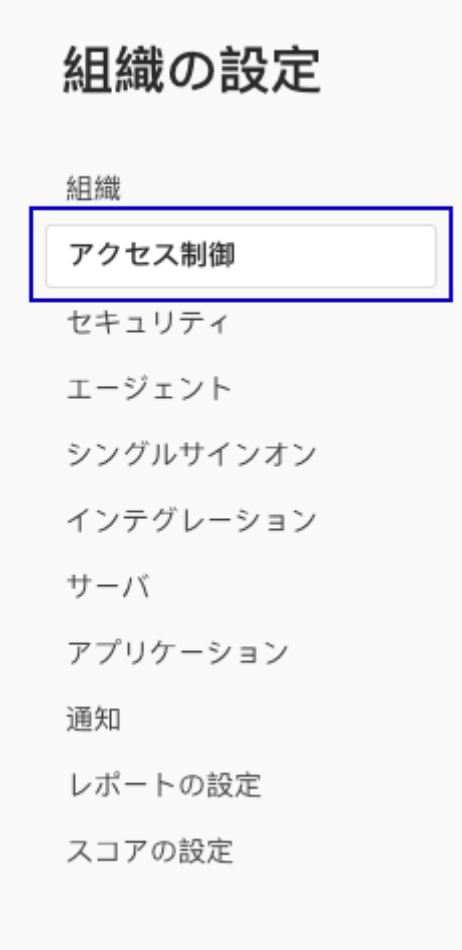
カスタムリソースグループの設定を変更する場合は、以下の手順を使用します。組込みのリソースグループの設定を変更することはできません。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **リソースグループ**タブを選択します。
4. 変更したいリソースグループの行の末尾にある**編集**アイコン(✎)を選択します。
5. リソースグループの設定を変更したら、**保存**を選択します。

リソースグループの削除

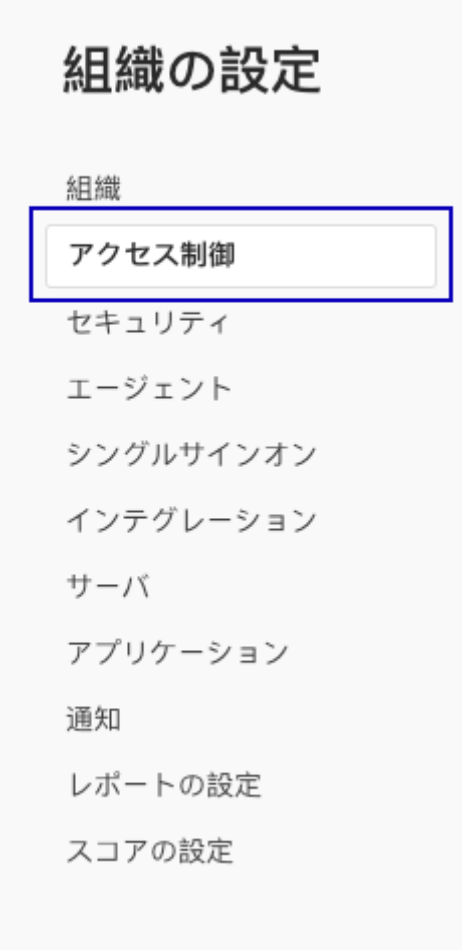
カスタムリソースグループを削除するには、以下の手順を使用します。 組込みのリソースグループは削除できません。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **リソースグループ**タブを選択します。
4. 削除したいリソースグループの行の末尾にある**削除**アイコン(🗑)を選択します。
5. 「リソースグループを削除」の画面で、**削除**を選択します。

ロール

ロールを使用することで、特定のロールを持つユーザがアクセスできるアプリケーション、プロジェクト、組織の設定を定義できます。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

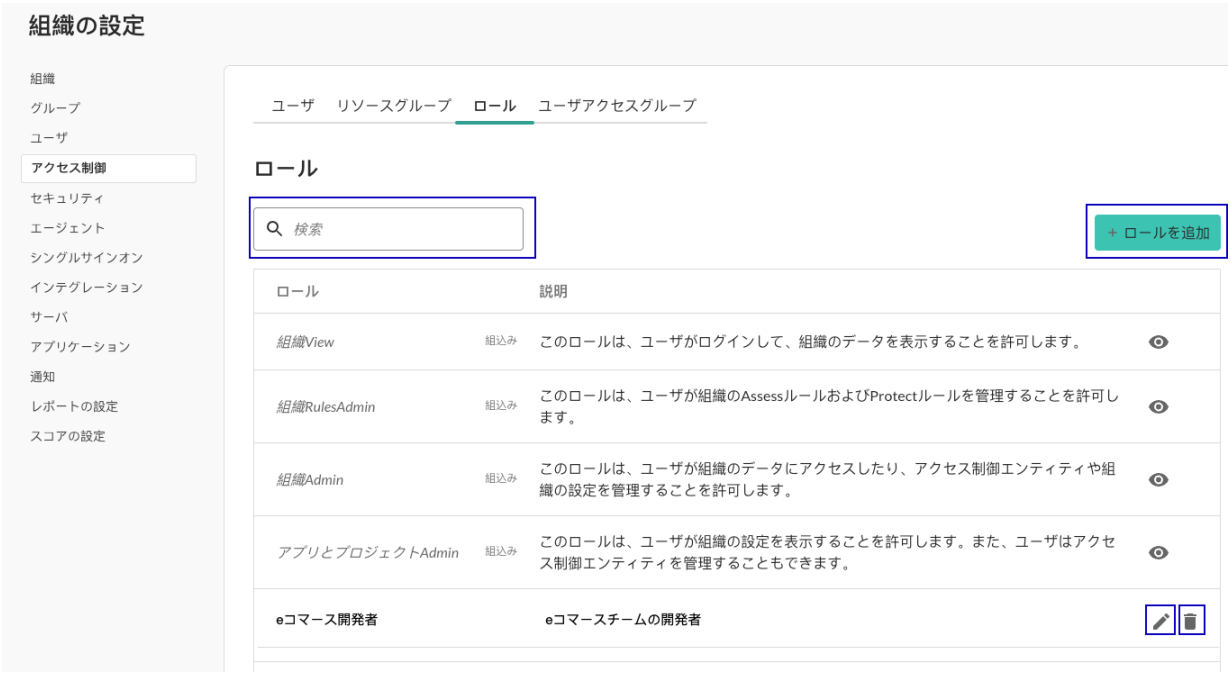
オンプレミス版をご利用のお客様は、組織のユーザとグループ (822ページ)を設定して、Contrast へのアクセスを管理します。

Contrast にはあらかじめ定義されている組込みのロールがありますが、カスタムロールを追加することもできます。

ロールタブ

「ロール」タブには、既存のロールの一覧が表示されます。このタブから、次のことを行えます。

- ロールの一覧を表示
  検索を使用して、特定のロールを見つけることができます。
- カスタムロールを追加 (856ページ)
- カスタムロールを編集 (858ページ)
- カスタムロールを削除 (859ページ)



組込みのロールと操作

ロールに関連付けられた各操作には、特定のタスクの実行とデータアクセスに対する権限 (836ページ)が与えられます。

注記
組込みのロールの設定を変更することはできません。**表示**アイコン(👁)を選択して、組込みロールの設定を参照することができます。

組織のロール

ロール名	含まれる組込みのリソースグループ	含まれるアクション
組織 View	全ての組織の設定	組織の閲覧
組織 Edit	全ての組織の設定	組織の設定の編集
組織 Admin	全てのアクセス制御	組織の管理
	全ての組織の設定	プラットフォーム組織の管理
組織 RulesAdmin	全ての組織の設定	組織のルールの管理

AppSec のロール

含まれる組込みのリソースグループ	含まれるアクション	含まれるアクション
アプリとプロジェクト Admin および組織 View	全てのアプリケーション	アプリケーションのルール管理
	全ての組織の設定	プロジェクトの閲覧、編集、削除
	全てのプロジェクト	組織の閲覧

DevOps のロール

ロール名	含まれる組込みのリソースグループ	含まれるアクション
アプリとプロジェクト Admin	全てのアプリケーション	組織の管理
	全ての組織の設定	アプリケーションの閲覧
	全てのプロジェクト	プロジェクトの閲覧
	全てのリソースグループ	組織の閲覧
	全てのロール	
	全てのユーザアクセスグループ	

アプリケーションのロール

ロール名	含まれる組込みのリソースグループ	含まれるアクション
アプリ View	全てのアプリケーション	アプリケーションの閲覧
アプリ Edit	全てのアプリケーション	アプリケーションの編集
アプリ Admin	全てのアプリケーション	アプリケーションの管理
アプリ RulesAdmin	全てのアプリケーション	アプリケーションのルール管理

スキャンプロジェクトのロール

含まれる組込みのリソースグループ	含まれるアクション	含まれるアクション
プロジェクト View	全てのプロジェクト	プロジェクトの閲覧
スキャン Upload	全てのプロジェクト	スキャンのアップロード
プロジェクト Admin	全てのプロジェクト	プロジェクトの閲覧、編集、削除
		プロジェクトの作成

Protect のロール

ロール名	含まれる組込みのリソースグループ	含まれるアクション
Protect データ View	全てのアプリケーション	Protect データのアクセス
Protect ポリシー Admin	全てのアプリケーション	Protect のポリシー管理
Protect 例外 Admin	全ての Protect 例外	Protect の例外管理
Protect 機密データ Admin	全ての Protect 機密データ	Protect の機密データ管理

サーバレスのロール

ロール名	含まれる組込みのリソースグループ	含まれるアクション
サーバレス View	全ての関数	サーバレスデータの閲覧

ロールの追加

ロールを作成して、ユーザがアクセスできる Contrast のリソースや操作をカスタマイズして指定します。

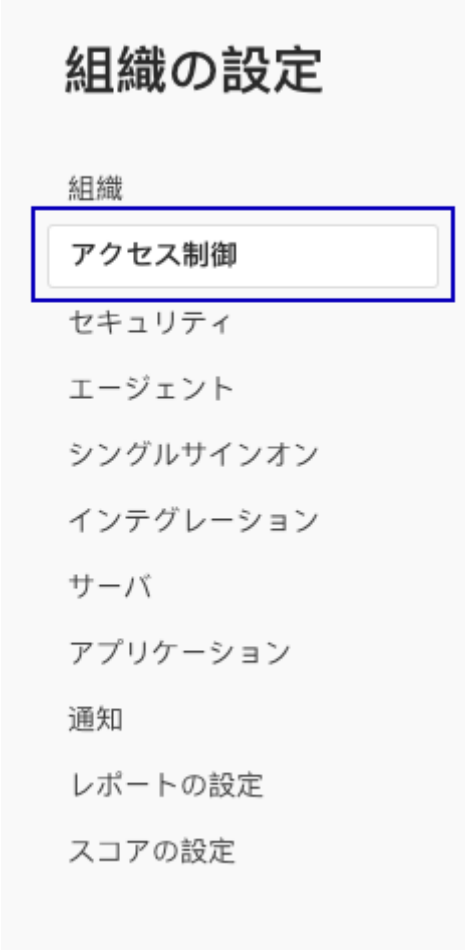
> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

開始する前に

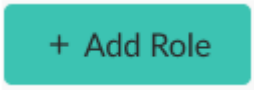
- **SaaS 版をご利用のお客様：**「Manage user access」操作がある口ールが必要です。
- **オンプレミス版をご利用のお客様：**組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。
- ロールにカスタムのリソースグループを追加 (849ページ)する必要があるかどうかを決めます。

手順

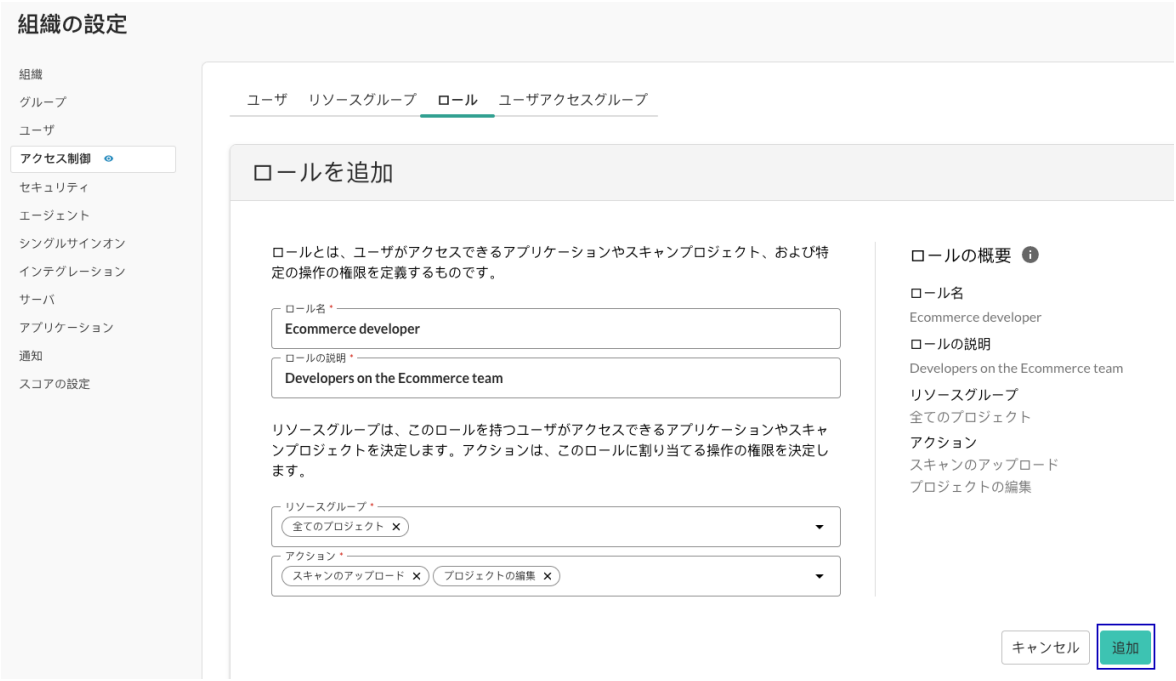
1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **Roles** タブを選択します。
4. **Add Role** を選択します。



5. ロールの設定を指定します。



a. **Role name**：ロールの名前を入力します。
名前は、組織内で一意である必要があります。スペースや特殊文字を含め、最大 255 文字まで使用できます。
b. **Role description**：ロールの説明を入力します。
このロールの目的が分かるような説明を入力してください。スペースや特殊文字を含め、最大 1,024 文字まで使用できます。
c. **Resource groups**：リソースグループを 1 つ以上選択します。
組込みのリソースグループ (848ページ)かカスタムグループ (849ページ)を選択します。
d. **Actions**：グループの操作 (836ページ)を選択します。

ロールの編集

カスタムロールを編集するには、以下の手順を使用します。組込みのロールは変更できません。

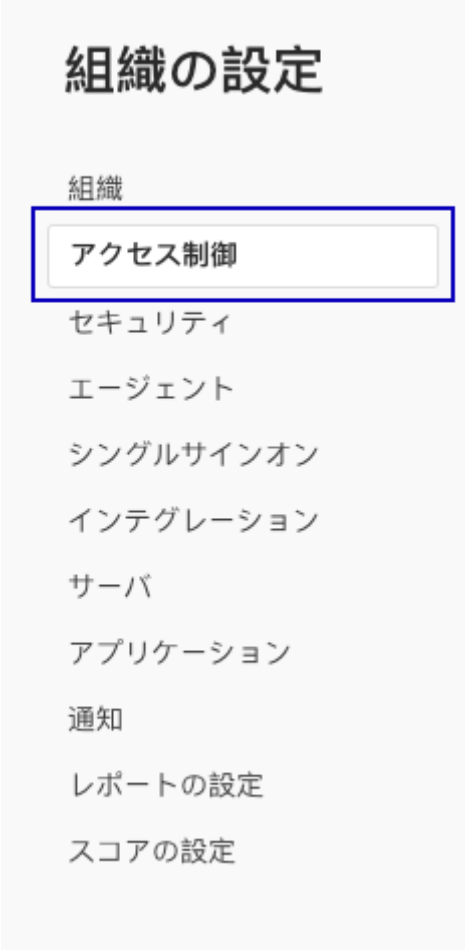
注記

この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ロール**タブを選択します。
4. 既存のロールを編集するには、変更したいロールの行の末尾にある**編集**アイコン(✎)を選択します。
5. 必要に応じて設定を変更したら、**保存**を選択します。

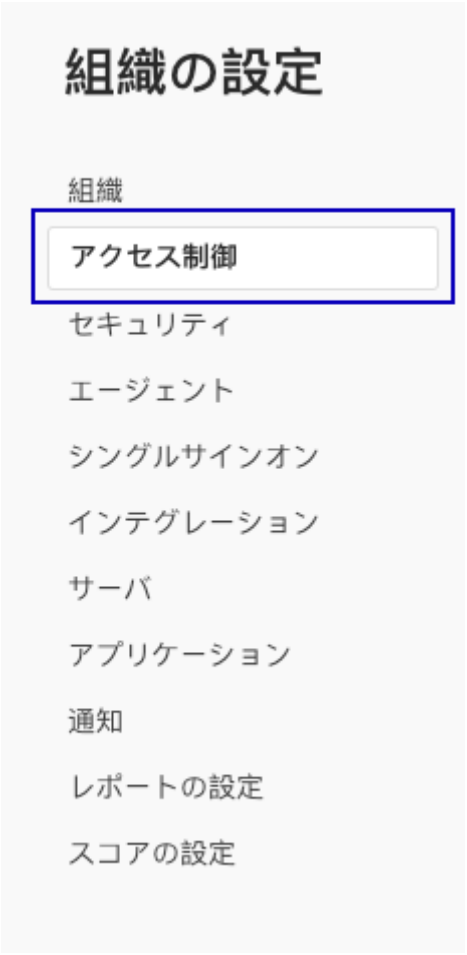
ロールの削除

カスタムロールを削除するには、以下の手順を使用します。組込みのロールは削除できません。

> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ロール**タブを選択します。
4. 削除したいロールの行の末尾にある**削除**アイコン(🗑)を選択します。
5. 「ロールを削除」の画面で、**削除**を選択します。

カスタムロールのベストプラクティス

複数の部門・担当が 1 つ以上のプロジェクトで共同作業している場合は、カスタムロールの作成を検討してください。カスタムロールを使用すると、部門・担当が必要なリソースにアクセスし、それらのリソースに対して適切な権限を持つことができるようになります。

カスタムロールに追加するアクションやリソースグループを決めるときの参考として、組込みロールの設定を表示して確認することもできます。

カスタムロールの計画

最初のステップは、カスタムロールの計画を立てることです。

- 現在、各人が担っている役割(ロール)について考えてみましょう。
  例えば、他の開発者とは異なるアクセスが必要なマネージャや主任開発者がいますか？
- 部門・担当が作業する必要のあるリソースについて考えみましょう。
  例えば、静的スキャンを使用して脆弱性を確認する部門・担当はありますか？その場合、スキャンプロジェクトへのアクセスが必要になるでしょう。
  部門・担当がどのアプリケーションで作業するかを考え、確実にアクセスできるようにしましょう。
- アクセスする必要があるリソースに対して、各人が実行する必要がある処理や操作について考えましょう。
  例えば、各担当者が組織の設定を変更する必要がありますか？スキャンを行うためにアーティファクトをアップロードできる必要がありますか？アプリケーションデータにアクセスする必要はありますか？など。

リソースグループ

次のベストプラクティスを検討してください。

- 管理を容易にするために、1 種類のリソースのみを含むリソースグループを作成します。例えば、プロジェクトのリソースを 1 つのグループにし、アプリケーションを別のグループにします。
  組込みのリソースグループが要件に合っていれば、カスタムグループを作成する代わりにそれを使用することもできます。
- ロールを使用して、異なるリソースグループに対してユーザが実行できるアクション(操作)を指定します。

例

カスタムロールを作成する 1 つの方法として、以下に例を示します。

この例では、e コマース製品を共同開発する開発部門のために、管理者がカスタムロールを作成していきます。

ステップ 1：全てのカスタムロールの計画を立てる

- **現在のロール**：このプロジェクトには、開発マネージャ、開発主任、ユーザインターフェイスに取り組むフロントエンド開発者、サービスおよび API を作成するバックエンド開発者がいます。これらの担当が全て共同で作業をしています。
- **リソース**：e コマース製品には、Contrast エージェントが組み込まれた共有アプリケーションとコンポーネント、および開発サイクルの初期段階でコードがスキャンされるスキャンプロジェクトがあります。
- **アクション**：開発マネージャと主任は、アプリケーションとスキャンプロジェクトの全ての設定を管理できる必要があります。フロントエンド開発者は、共有されたユーザインターフェイスモジュールにアクセスできる必要があります。バックエンド開発者は、API とバックエンド サービスにアクセスできる必要があります。

ステップ 2：カスタムリソースグループを作成する

ユーザごとに割り当てる必要があるさまざまなアクセス許可に対応するために、次のリソースグループを作成します。

- **共有 UI アプリケーション**：このリソースグループには、フロントエンド開発者が実行時にテストする必要がある UI に特化したアプリケーションを含めます。
- **共有 UI プロジェクト**：このリソースグループには、フロントエンド開発者が開発の初期段階で脆弱性を見つけるために実行する、UI に特化したスキャンプロジェクトを含めます。
- **共有 API アプリケーション**：このリソースグループには、バックエンド開発者が実行時にテストする必要がある共有 API アプリケーションを含めます。
- **共有バックエンドサービス**：このリソースグループには、バックエンド開発者が実行時にテストする必要がある共有サービスを含めます。

全てのリソースグループを **e コマース開発**という、親リソースグループにグループ化します。

全てのカスタムロールに親グループを含めることにします。アクションを使用して、特定のリソースにアクセスできるロールと、これらのリソースに対してユーザが持つ権限を決定します。

ステップ 3：カスタムロールを作成する

親リソースグループである「e コマース開発」を特定のアクションで使用するように、以下のロールを設定します。

- **e コマース管理者**：このロールは、e コマース開発マネージャと主任のためのものです。このロールには、以下のアクションを含めます。
  - **アプリケーションのルール管理**：Assess ルールや Protect ルールの変更、修復ポリシーの設定、ライブラリポリシーの設定などのタスクを実行できます。

  - **アプリケーションの管理**：アプリケーションの設定を変更できます。
  - **プロジェクトの閲覧、編集、削除**：スキャンプロジェクトの表示、編集、作成ができます。
  - **プロジェクトの削除**：スキャンプロジェクトの削除ができます。
  - **組織の閲覧**：スコアの設定や API キーの取得などのタスクを実行できます。
- **e コマースフロントエンド開発者**：このロールは、e コマースのフロントエンド開発者のためのものです。このロールには、以下のアクションを含めます。
  - **アプリケーションの閲覧**：アプリケーションのデータ(脆弱性、アプリケーションの情報、ライブラリの情報など)を表示できます。
  - **アプリケーションの編集**：アプリケーションのマージ、バグ管理システムへのデータ送信、脆弱性の設定の編集などのタスクを実行できます。
  - **組織の閲覧**：スコアの設定や API キーの取得などのタスクを実行できます。
  - **スキャンのアップロード**：コードの脆弱性をスキャンできます。
- **e コマースバックエンド開発者**：このロールは、e コマースのバックエンド開発者のためのものです。このロールには、以下のアクションを含めます。
  - **アプリケーションの閲覧**：アプリケーションのデータ(脆弱性、アプリケーションの情報、ライブラリの情報など)を表示できます。
  - **アプリケーションの編集**：アプリケーションのマージ、バグ管理システムへのデータ送信、脆弱性の設定の編集などのタスクを実行できます。
  - **組織の閲覧**：スコアの設定や API キーの取得などのタスクを実行できます。

ステップ 4：カスタムユーザグループを作成する

ロールごとにユーザアクセスグループを作成し、アクセスが必要なリソースに基づいて、個々のユーザを割り当てます。

- **e コマース管理グループ**：「e コマース管理者」ロールのある全てのユーザを入れます。
- **e コマースフロントエンド開発グループ**：「e コマースフロントエンド開発者」ロールのある全てのユーザを入れます。
- **e コマースバックエンド開発グループ**：「e コマースバックエンド開発者」ロールのある全てのユーザを入れます。

ユーザアクセスグループ

ユーザアクセスグループを使用して、1 人以上のユーザに対して、特定のロールとリソースグループを指定します。ロールとリソースグループは、ユーザの権限を定義します。

組込みのユーザアクセスグループにユーザを追加することも、カスタムグループ (864ページ)を作成することもできます。

注記

この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

ユーザアクセスグループタブ

「ユーザアクセスグループ」タブには、既存のグループの一覧が表示されます。このタブから、次のことを行えます。

- ユーザアクセスグループの一覧を表示
  検索を使用して、特定のグループを見つけることができます。

- ユーザアクセスグループを追加 (864ページ)
- ユーザアクセスグループを編集 (866ページ)
- ユーザアクセスグループを削除 (867ページ)

組込みのユーザアクセスグループ

Contrast にあらかじめ定義された組込みのユーザアクセスグループがあります。以下の組込みのユーザアクセスグループに、ユーザを追加することができます。

組込みのグループ	含まれる組込みのロール
組織閲覧グループ	組織 View プロジェクト View アプリ View
組織編集グループ	組織 Edit
組織ルール管理グループ	組織 RulesAdmin
組織管理グループ	アプリ Admin プロジェクト Admin 組織 Admin Protect データ View サーバレス View
AppSec 管理グループ	アプリとプロジェクト Admin および組織 View
DevOps 管理グループ	アプリとプロジェクト Admin
開発グループ	組織 View Protect データ View サーバレス View スキャン Upload アプリ View プロジェクト View
スキャン運用グループ	プロジェクト View
Protect 閲覧グループ	Protect データ View

組込みのグループ	含まれる組込みのロール
サーバレス閲覧グループ	サーバレス View

注記
ユーザを追加するか削除する以外は、組込みのユーザアクセスグループの設定を変更することはできません。

関連項目
組込みのロール (853ページ)

ユーザアクセスグループの追加

ユーザアクセスグループを作成することで、プロジェクト、アプリケーションおよび組織の設定にアクセスできるユーザの集合体を定義できます。グループ内のユーザに割当てるロールによって、ユーザがアクセスできる機能を決定できます。

注記
この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

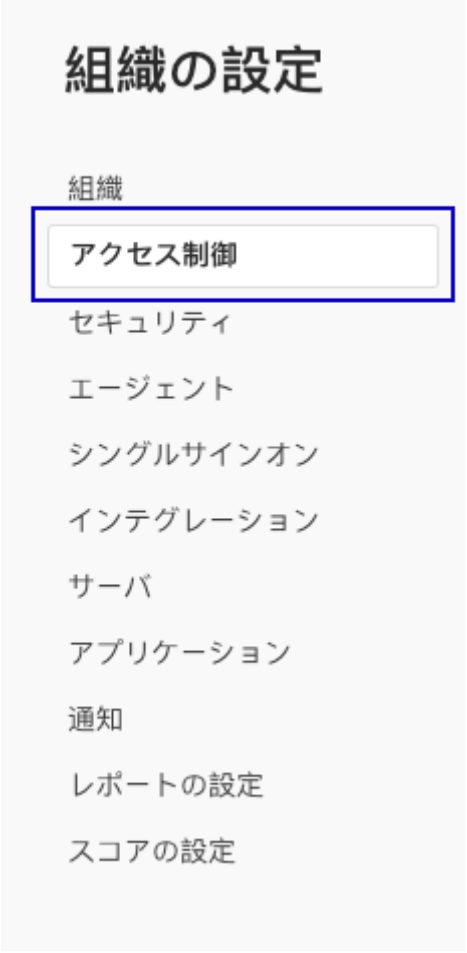
オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

開始する前に

- 必要な場合は、ユーザを追加 (843ページ)します。
- カスタムロールを追加 (856ページ)したり、カスタムのリソースグループ (849ページ)が必要であるかどうか判断してください。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. 「ユーザアクセスグループ」タブで、**グループを追加**を選択します。

4. 「ユーザアクセスグループを追加」の画面で、以下の設定を行います。

- **ユーザアクセスグループ名**：ユーザグループの名前を入力します。

名前は、組織内で一意である必要があります。スペースや特殊文字を含め、最大 255 文字まで使用できます。
- **ユーザアクセスグループの説明**：グループの説明を入力します。
  このグループの目的が明確に分かるような説明を入力してください。スペースや特殊文字を含め、最大 1024 文字まで使用できます。
- **ユーザ**：グループに所属するユーザを指定します。
  ユーザの一覧を表示するには、ボックスの端にある三角形 ▾ を選択するか、名前を入力してみてください。
  特定のユーザが表示されない場合は、**ユーザメニュー > 組織の設定**より**ユーザ**にアクセスして、そのユーザが Contrast の組織のユーザであることを確認してください。
- **ロール**：ドロップダウンから、グループ内の全てのユーザに適用されるロールを 1 つ以上選択します。
  組込みのロール (853ページ)やカスタムロール (856ページ)を選択できます。

5. **追加**を選択します。

ユーザアクセスグループの編集

既存のユーザアクセスグループを編集するには、以下の手順を使用します。

組込みのユーザアクセスグループでは、グループに割り当てたユーザのみ変更できます。

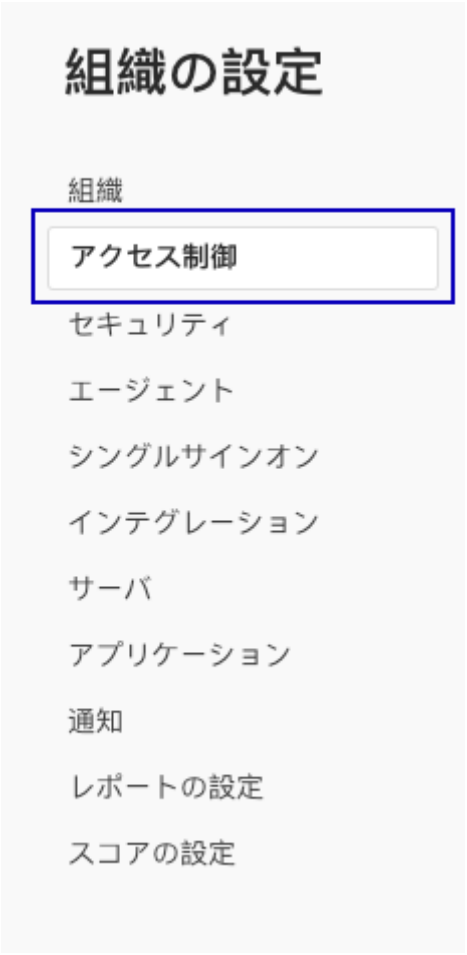
注記

この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。

オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ユーザアクセスグループ**タブを選択します。
4. 変更したいグループの行の末尾にある**編集**アイコン(✎)を選択します。
5. 必要に応じて設定を変更したら、**保存**を選択します。

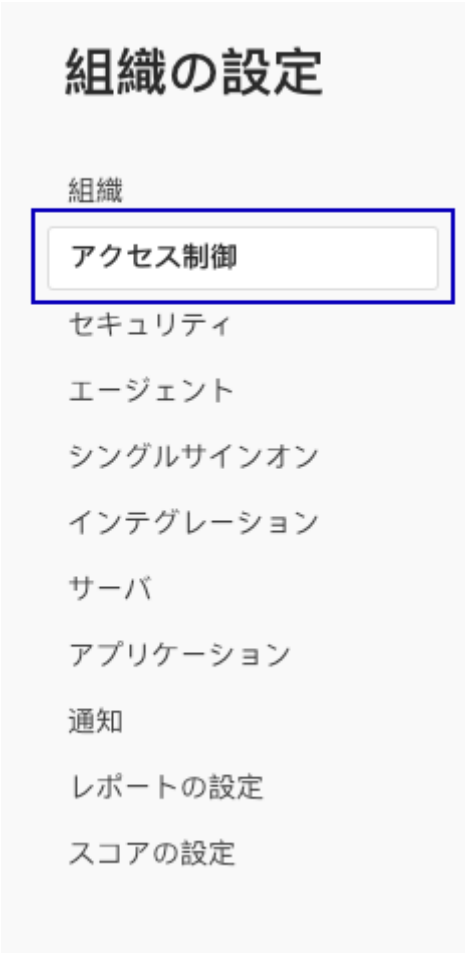
ユーザアクセスグループの削除

カスタムのユーザアクセスグループを削除するには、以下の手順を使用します。組込みのユーザアクセスグループは削除できません。

> **注記**
> この機能は、SaaS 版ご利用のお客様のみでサポートされ、プレビューモードとなります。この機能を使用するには、Contrast サポートまでご連絡ください。
>
> オンプレミス版をご利用のお客様は、組織のユーザとアクセスグループ (822ページ)を設定することで、Contrast へのアクセスを管理できます。

手順

1. ユーザメニューから**組織の設定**を選択します。
2. **アクセス制御**を選択します。



3. **ユーザアクセスグループ**タブを選択します。
4. 削除したいグループの行の末尾にある**削除**アイコン(🗑)を選択します。
5. 「ユーザアクセスグループを削除」の画面で、**削除**を選択します。

脆弱性クローズの承認

組織の管理者は、組織内の脆弱性のクローズ時に管理者の承認を必要 (786ページ)とすることができます。脆弱性のクローズを承認または却下 (706ページ)するには、組織ロールに Rules Admin があり、対象アプリケーションに対しても Rules Admin ロールが必要です。

この要件を設定するには：

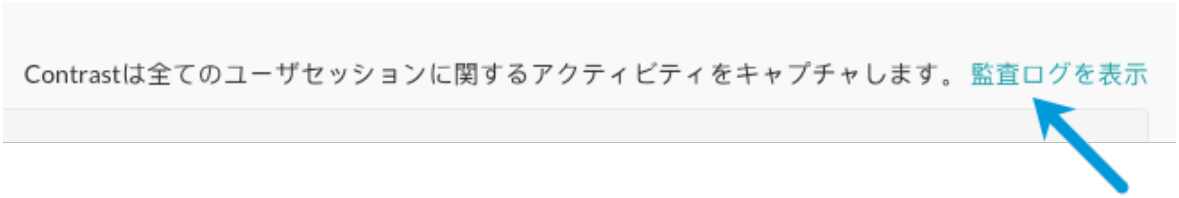
1. **ユーザメニュー**で、**ポリシーの管理 > 脆弱性の管理 > 脆弱性の動作**を選択します。
2. **脆弱性のクローズ時に管理者の承認が必要**の横にあるチェックボックスをオンにします。
3. ユーザが脆弱性をクローズするときに自動的に**保留中**になる脆弱性のステータスと深刻度を選択します。
4. 対象となる脆弱性のクローズがユーザから要求されると、全ての組織の管理者にレビューが必要であることが Contrast 内で通知されます。
   各脆弱性のステータスは、組織の管理者がクローズのレビューを送信するまで**保留中**のままになります。管理者の承認を得るには、ステータスと深刻度の*両方*を選択する必要があります。
   レビュー担当者が脆弱性のクローズを却下する場合は、却下する理由を提供する必要があります。
   レビューが完了すると、レビュー担当者からのフィードバックが脆弱性の**アクティビティ**タブに表示されます。
   この機能を無効にすると、保留中のクローズ処理は自動的に承認されます。

注記
保留中の状態でも、それまでの脆弱性のステータスが、組織のレポートや統計情報に反映されます。

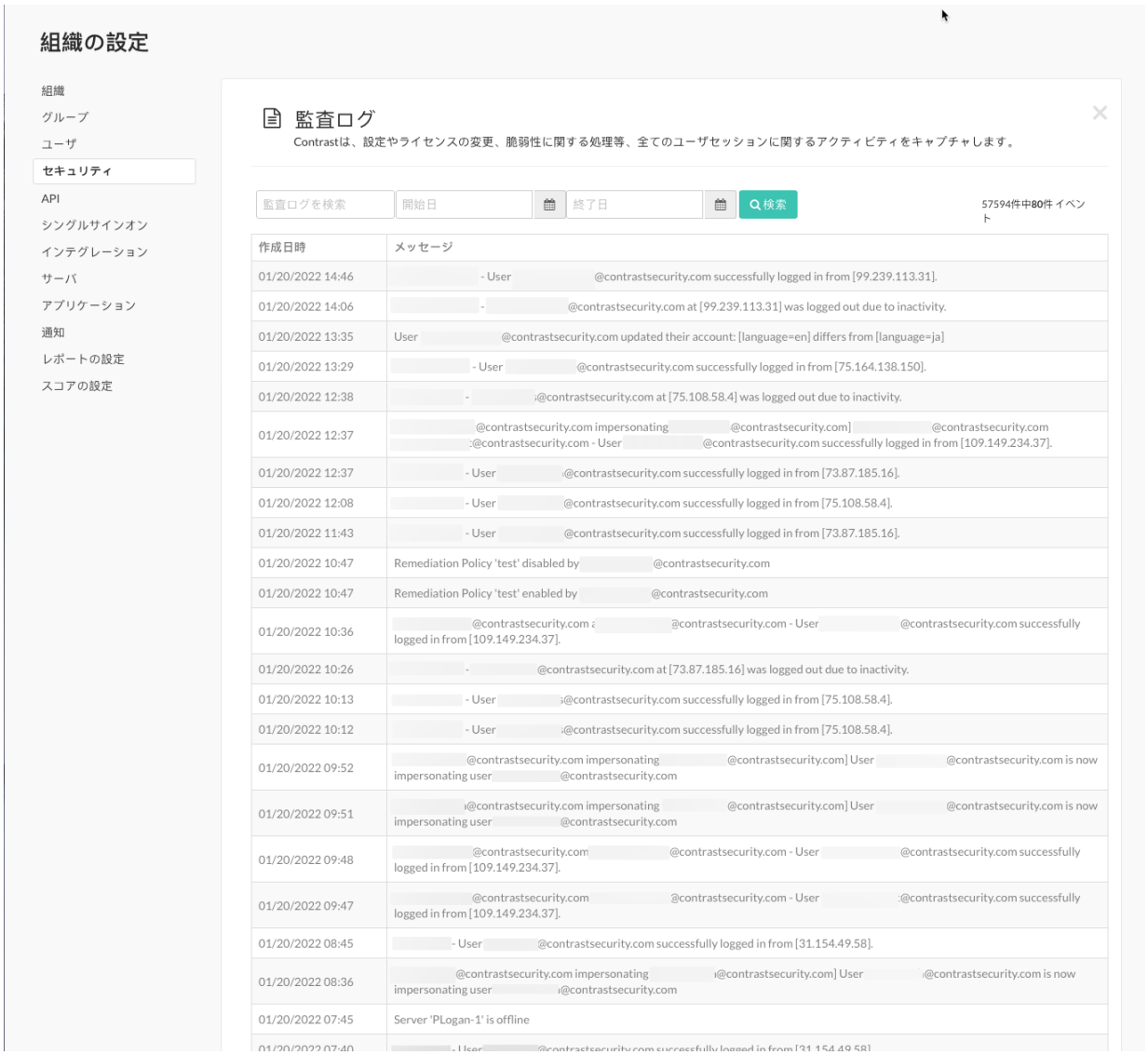
監査ログの表示

Contrast では、設定やライセンスの変更、脆弱性に対するアクションなど、全てのユーザセッションに関するアクティビティがキャプチャされます。

1. **ユーザメニュー**で**組織の設定** > **セキュリティ**を選択します。
2. 画面右上の**監査ログを表示**をクリックします。



監査ログのページが表示されます。



以下の様な情報が表示されます。

エージェント	エージェントのダウンロード/起動/Contrast サービス
警告	生成 削除 更新
アプリケーション	ライセンスの適用 アーカイブ Assess ルールの設定変更 組織の設定変更 ルールの有効/無効化 マージ/マージ解除 復元 リセット

攻撃	イベントに備考を入力 備考の作成/編集/削除 消去
バグ管理システム	生成 更新
E メール	ドメインの追加 ライブラリの共有
グループ	組織のグループへのメンバの作成/更新/修正
IP アドレスの範囲	追加 削除
キー	API およびエージェントのサービスキーのローテーション
通知の設定	組織やユーザの設定の更新
パッチ	仮想パッチの生成/更新/オンやオフ/削除
ポリシー	クリーンアップの設定の変更 ライブラリの制限の変更 パスワードの変更 スコア付けの変更 タイムアウトの変更 コンプライアンスポリシーの作成/削除/無効化/有効化/更新 組織の修復ポリシーの作成/削除/更新/有効化 セキュリティ制御の作成/削除/更新/有効化 例外ルールの作成/削除/更新/有効化
レポート	レポートの作成 脆弱性の傾向レポート
サーバ	デフォルト値の変更 通知の作成 ライセンスの削除/無効化/更新/削除 Protect の有効/無効化
システム管理者(SuperAdmin)	作成 ユーザのなりすまし 更新
トレース	備考の追加/編集/削除 自動修復 トレースの共有/削除/マージ/ステータス変更 深刻度の更新
ユーザ	追加 アクティベーション アクセスの変更 プロビジョニング/バルクによる作成/作成 削除 Protect の付与/取消 組織へのインポート 非特権ユーザによる試行 更新

Webhook	作成

ヒント
検索フィールドを使用して、日付や名前で特定のログを検索できます。

なりすまし

なりすまし機能により、システムロール (957ページ)のある他のユーザが既存のユーザとして組織にアクセスし、問題のトラブルシューティングを行うことが可能になります。

組織のなりすましは、24 時間後に自動的に無効になります。

SaaS 版をご利用のお客様で、組織になりすましの設定が表示されていない場合は、Contrast サポートにご連絡ください。オンプレミス版のお客様は、必要に応じてなりすましを管理・使用できます。

なりすましアクセス

デフォルトでは、全ての組織でなりすましの設定が有効です。組織内で別のユーザになりすますかどうかは、組織管理者が管理します。

なりすましを行うためのアクセスを変更するには：

- SuperAdmin ユーザは組織を編集 (908ページ)して、 特定の組織の**なりすましを有効にできる**オプションをオンまたはオフに設定します。
  この設定は、組織管理者が組織のなりすまし設定を表示できるかどうかに影響します。
- 組織管理者は、組織のセキュリティの設定を編集 (872ページ)して、なりすましの使用を管理します。

なりすましをオンにすると：

- SuperAdmin ロールがある場合は、組織のページより該当の組織の行で**なりすます**を選択すれば、なりすまし (919ページ)を使用できます。
  その組織内の最初の組織管理者に対して、なりすますことができます。
- SuperAdmin ロールがある場合は、ユーザのページより、なりすますユーザを選択することができます。
- ServerAdmin または SystemAdmin ロールがあるユーザの場合は、組織のページより該当の組織の行で**なりすます**を選択することで、なりすまし (919ページ)を使用できます。
  その組織内の最初の組織管理者に対して、なりすますことができます。組織へのアクセス権が必要です。

監査ログ

Contrast の監査ログには、以下のような、なりすましのアクティビティが記録されます。

- なりすましの有効化/無効化
- なりすましが発生した組織
- なりすましのステータス変更に関連付けられたサーバキー
- なりすまし試行の拒否

なりすましを有効にする

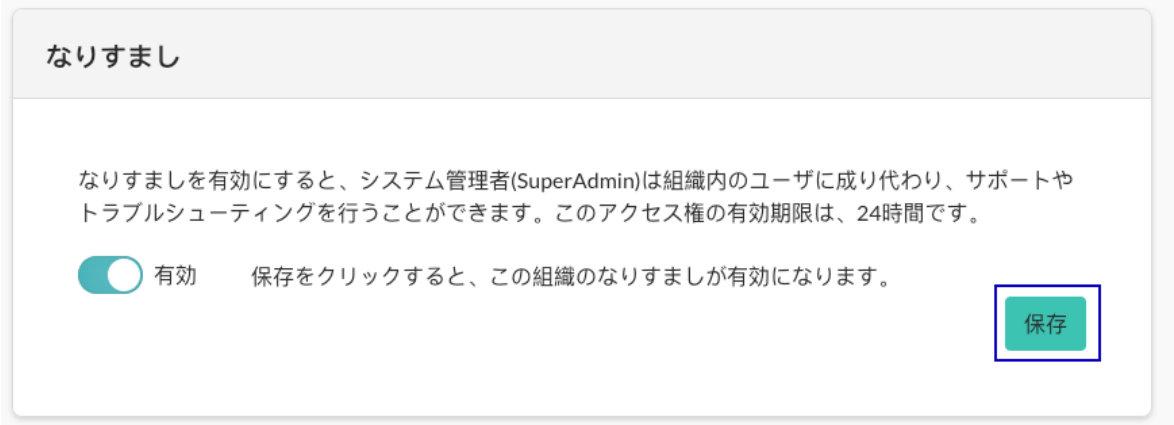
組織でなりすまし (872ページ)を有効にするには、次の手順を実行します。

開始する前に

- **組織の設定 > セキュリティ**にアクセスし、なりすましを有効にするための設定が表示されていることを確認してください。
  設定が表示されていない場合：
  - **SaaS 版のお客様：** Contrast サポートにご連絡ください。
  - **オンプレミス版のお客様：** SuperAdmin ユーザに組織を編集 (908ページ)して、**なりすましを有効にできる**の設定をオンにするよう依頼してください。

手順

1. ユーザメニューから、**組織の設定**を選択します。
2. **セキュリティ**を選択します。
3. なりすましのセクションで、有効にする設定を選択し、**保存**を選択します。



システム管理(EOP)

オンプレミス版のお客様のみ、システム管理が必要です。SaaS 版のお客様のシステム管理は、Contrast Security が行います。

オンプレミス版の利用を開始する (874ページ)方法を参照したり、システム運用の管理方法 (907ページ)方法などを検討してください。

オンプレミス版をはじめる

オンプレミス版のお客様は、インターネットに接続せずに独自の Contrast インスタンスを管理でき、セキュリティポリシー、データベース、認証などを設定できます。この設定は、1 組織につき 1 回だけ必要です。

重要

Contrast を SaaS 版として使用できる場合は、オンプレミス版 Contrast の追加のインストールやメンテナンスを行う必要はありません。

SaaS 版 Contrast は SOC-2 Type II に準拠しており、継続的に機能の更新が行われます。SaaS 版 Contrast に接続するには、管理者から提供された認証情報を使用してログインします。そして、続けてエージェントをインストール (46ページ)してください。

オンプレミス版のお客様は、Assess や Protect、もしくは両方を使用するために、少なくとも以下の 2 つのインストールを完了する必要があります。

- Contrast のインストール (879ページ)
- 各アプリケーションサーバのエージェントのインストール (46ページ)

Contrast のインストールには、Tomcat サーブレットコンテナ、MySQL データベースインスタンス、AdoptOpenJDK Hotspot Java 仮想マシンなど、システムを構成する全ての組込みコンポーネントが含まれています。これらのコンポーネントは全て、インストールバイナリ内に組み込まれており、Contrast アーキテクチャの一部として単一サーバにデプロイされます。

Contrast をインストールする前に、ご使用の環境が以下の要件を満たしていることを確認してください。

- システム要件 (875ページ)
- サイジングの推奨 (876ページ)

以下の項目は、インストール後さらに設定できます。

- Tomcat (891ページ)
- JRE (892ページ)
- HTTPS (892ページ)
- Contrast の設定 (936ページ)

長期的には、以下についての計画が必要となります。

- システム管理 (907ページ)
- 設定の構成 (936ページ)
- Contrast システムの保守 (943ページ)

Contrast のインストール

オンプレミス環境で Contrast をインストールし、デプロイするためのオプションには、次のものがあります。

- Contrast インストーラでインストールする (879ページ)
- 分散 MySQL データベース (883ページ)を使用する
- WAR ファイルを使用してデプロイする (882ページ)
- 分散環境 (886ページ)で導入する

次の手順

- Contrast のシステム要件 (875ページ)を確認する
- Contrast のサイジング推奨事項 (876ページ)を確認する
- Contrast Hub または curl コマンド (877ページ)で Contrast インストーラをダウンロードする

Contrast のシステム要件

次の表に、Contrast アプリケーションをインストールするためのシステム要件を示します。

Contrast をインストールする前に：

- Contrast Hub または curl コマンド (877ページ)を使用して、Contrast インストーラをダウンロード (878ページ)してください。
- サイジングの推奨 (876ページ)を確認してください。

要件	推奨	最小	備考
OS アーキテクチャ	64 ビット	64 ビット	メモリの要件のため、Contrast アプリケーションは 64 ビットアーキテクチャでのみ実行できます。
オペレーティングシステム	• RHEL 8 • RHEL/CentOS 7 • Microsoft Windows 2019 • Ubuntu 18.04 LTS 以降 (20.04 LTS まで)	• RHEL/CentOS 7 • Microsoft Windows 2012 R2 以降 • Ubuntu 16.04 LTS	CentOS 6 のサポートは、2020 年 12 月 1 日に終了しました。
Java	Java はインストーラにバンドルされています。		
MySQL	• Contrast のバージョン 3.8.11 以前のオンプレミス版には、MySQL 5.7.23 がインストーラにバンドルされています。 • Contrast 3.9.0 以降のオンプレミス版では、MySQL 8 が Linux インストーラにバンドルされています。 • Contrast 3.9.3 以降のオンプレミス版では、MySQL 8 が Linux インストーラと Windows インストーラにバンドルされています。 何か問題がありましたら、弊社サポートにご連絡ください。		

重要

- MySQL 8(デフォルトでバイナリログが有効)を使用しているオンプレミス版のお客様は、Contrast でストアドプロシージャが作成できるように、システム変数 `log_bin_trust_function_creators` を **ON** に設定する必要があります。詳細については、MySQL のドキュメントを参照してください。
- MySQL 8 を使用しているオンプレミス版のお客様は、Contrast で CSV ファイルを受け取り SCA データのインポートができるように、システム変数 `local_infile` を **ON** に設定する必要があります。詳細については、LOAD DATA LOCAL のセキュリティ上の考慮事項を参照してください。

分散インストールの場合の MySQL と Java の要件

Contrast を分散アプリケーションとしてデプロイする場合、または独自の MySQL データベースを使用する場合は、以下の要件を使用してください。それ以外の全てのオンプレミスインストールでは、Contrast インストーラに含まれる MySQL および Java ソフトウェアを使用してください。

何か問題がありましたら、弊社サポートにご連絡ください。

要件	推奨	最小
Java	17	11
MySQL	• 8.0.30(Contrast 3.9.1 以降) • 5.7.23(Contrast 3.9.0 以前)	• 8.0.27(Contrast 3.9.1 以降) • 5.7.23(Contrast 3.9.0 以前)

SuperAdmin アカウント

インテグレーションでの接続が正しく動作するように、Contrast Security で作成したデフォルトのアカウントとは異なる SuperAdmin アカウントを作成してください。デフォルトの SuperAdmin アカウントを使用し続けると、接続エラーが発生する可能性があります。

エージェントが多数の場合の分散構成

接続されているエージェントの数が 100 を超えて利用する予定がある場合は、分散環境の構成(886ページ)を検討してください。このような場合に分散構成を使用しないと、パフォーマンスの問題が発生する可能性があります。

Contrast アプリケーションのサイジング推奨事項

Contrast のための CPU とメモリのリソースは、接続するエージェントの数と Contrast アプリケーションに通信を返すアプリケーションのトラフィックによって異なります。このページの推奨事項は、アプリケーションのサービスに適用されます。

パフォーマンスに影響を与えるその他の要因には、以下があります。

- **Contrast に報告されるデータの利用者による Web トラフィック**
  Contrast は、高度なトランザクションシステムで、リアルタイムでデータセットを計算しデータの利用者に返します。より多くのユーザがシステムを利用するほど、より多くのコンピューティングリソースとメモリリソースが必要になります。
- **長期間にわたりアプリケーションに保持される大量のデータ**
  時間の経過に合わせてデータを積極的に削除することも、データを保持することもできます。どのトランザクションシステムでも、クエリの対象となるデータセットが大きくなるほど、コンピューティング要件も大きくなります
- **100 を超えるエージェントの接続**
  接続されているエージェントの数が 100 を超えて利用する予定がある場合は、分散環境の構成(886ページ)を検討してください。このような場合に分散構成を使用しないと、パフォーマンスの問題が発生する可能性があります。

次のガイドラインを参考にして、ワークロードに合わせて要件を拡張するために、適切なリソースの組み合わせを選択してください。

- **小規模構成**：Contrast と通信するエージェント数のが 3～30 程度、エンドユーザが 5〜25 人規模の Web トラフィックを処理する場合に推奨される構成です。エンドユーザは、1 日に複数回システムにアクセスし、アラートやレポート、インテグレーションなどを積極的に使用します。
  接続するエージェントの数が多いほど、Contrast が実行中のトレース処理をするためのメモリ要件が大きくなります。ストレージは、トレースデータの寿命とシステム管理者による口グファイルの保存状況によって異なります。

vCPU 数	クロック速度	RAM	ストレージ
4～8 個	2.5 GHz～3.3 GHz	16 GB～24 GB	100 GB～200 GB

- **大規模構成**：Contrast と通信するエージェントが 30～100 程度のより大きな作業負荷や、エンタープライズ規模で展開される 25 ユーザ以上の大規模な Web トラフィックを想定しています。エンドユ

ーザは 1 日に複数回システムにアクセスし、アラートやレポート、インテグレーションなどの Contrast の機能を積極的に使用します。
接続しているエージェントの数やエンドユーザが多いほど、Contrast が実行中のトレース処理をするためのメモリ要件が大きくなります。ストレージは、トレースデータの寿命とシステム管理者によるログファイルの保存状況によって異なります。

vCPU 数	クロック速度	RAM	ストレージ
8 ~ 16 個	2.5 GHz ~ 3.3 GHz	24 GB ~ 48 GB	200 GB ~ 500 GB

重要
作業負荷や構成規模に関係なく、最低 16GB の RAM を Contrast アプリケーションに割り当ててください。

ヒント
Contrast REST API アーキテクチャを自動化またはデータ抽出の目的で使用している場合、および大規模な自動リグレッションスイートでエージェントの継続的インテグレーション(CI)を目的として使用している場合は、大規模構成のガイドラインに従ってください。

curl コマンドによる Contrast ソフトウェアのダウンロード

ライセンスが提供されたら、curl コマンドを使用して Contrast のインストーラとライセンスをダウンロードすることができます。会社のアクセス権を誰が持っているか不明な場合は、サポートにお問い合わせください。

Contrast のその他のソフトウェアのファイルも、curl コマンドを使用して、ダウンロードできます。

また、インストーラやライセンスファイルは Contrast Hub からダウンロードすることもできます。

注記
Contrast 3.9.10 より、インストーラにライブラリ・CVE データが含まれなくなりました。エアギャップでのインストールの場合は、このデータを手動でダウンロードする必要があります。インターネットに接続できる場合は、このデータを自動で更新(906ページ)するようにシステムを設定できます。

インストーラファイルのダウンロード手順

1. Contrast の最新の Linux 用インストーラをダウンロードするには、以下のコマンドを使用します。

   ```
   curl -L https://hub.contrastsecurity.com/h/api/artifacts/installer/sh/
   nocache -u <ContrastHubUsername> -o "contrast-latest-nocache.sh"
   ```

   `<ContrastHubUsername>`は、お客様の Contrast Hub アカウントに置き換えてください。コマンドを実行すると、パスワードの入力を求められます。
2. Contrast の最新の Windows 用インストーラをダウンロードするには、以下のコマンドを使用します。

```
curl -L https://hub.contrastsecurity.com/h/api/artifacts/installer/exe/
nocache -u <ContrastHubUsername> -o "contrast-latest-nocache.exe"
```

<ContrastHubUsername>は、お客様の Contrast Hub アカウントに置き換えてください。コマンドを実行すると、パスワードの入力を求められます。

3. **エアギャップでのインストールの場合：**以下のコマンドを使用して、ライブラリ・CVE データをダウンロードします。

```
curl -JLO https://hub.contrastsecurity.com/h/api/artifacts/ardy_export -
u <ContrastHubUsername>
```

<ContrastHubUsername>は、お客様の Contrast Hub アカウントに置き換えてください。コマンドを実行すると、パスワードの入力を求められます。

4. 最新の Contrast WAR ファイルは、以下のコマンドでダウンロードします。

```
curl -JLO https://hub.contrastsecurity.com/h/api/artifacts/war -u \
<ContrastHubUsername>
```

<ContrastHubUsername>は、お客様の Contrast Hub アカウントに置き換えてください。
この WAR ファイルは、既存の Tomcat サーバがあり、そのサーバに Contrast をインストールする場合に便利です。

5. 以下のコマンドを使用して、ライセンスファイルをダウンロードします。

```
curl --request GET --url https://hub.contrastsecurity.com/h/rest/
customer/license/text -u <ContrastHubUsername> > license.lic
```

<ContrastHubUsername>は、お客様の Contrast Hub アカウントに置き換えてください。コマンドを実行すると、パスワードの入力を求められます。

6. ファイルをダウンロードしたら、Contrast をインストール (879ページ)します。

その他のソフトウェアダウンロード

次の curl コマンドで、Contrast のその他のソフトウェアをダウンロードすることができます。

```
curl -JLO https://hub.contrastsecurity.com/h/api/artifacts/<OtherSoftware>
-u <ContrastHubUsername>
```

<Other Software>を次のいずれかに置き換えてください。

- .NET Framework エージェントの場合は、dotnet
- .NET Core エージェントの場合は、dotnet_core
- IIS 用の.NET Core インストーラの場合は、dotnetcore_installer_for_iis

Contrast インストーラのダウンロード

ライセンスが提供されると、Contrast Hub アカウントにアクセスできるようになり、そのアカウントでインストーラとライセンスをダウンロードできます。会社のアクセス権を誰が持っているか不明な場合は、サポートにお問い合わせください。

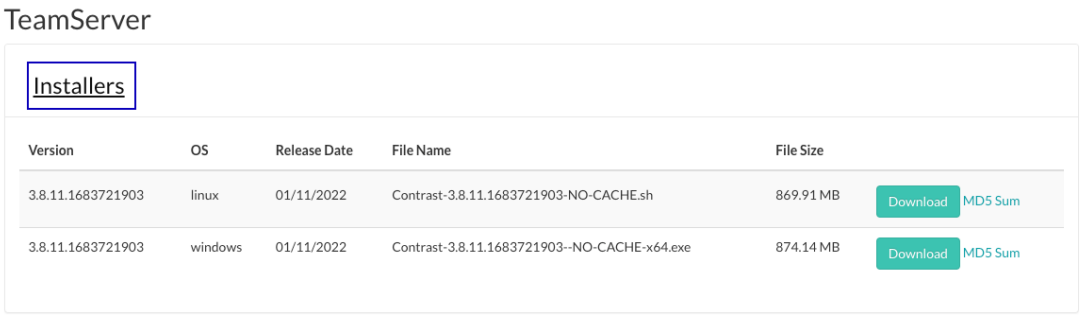
または、Contrast Hub の代わりに curl コマンド (877ページ)を使用してインストーラやライセンスをダウンロードすることもできます。

注記

Contrast 3.9.10 より、インストーラにライブラリ・CVE データが含まれなくなりました。エアギャップでのインストールの場合は、このデータを手動でダウンロードする必要があります。インターネットに接続できる場合は、このデータを自動で更新 (906ページ)するようにシステムを設定できます。

手順

1. 提供された認証情報を使用して Contrast Hub にログインします。
2. お使いのオペレーティングシステムの Contrast インストーラをダウンロードします。
   a. ナビゲーションバーで、**Downloads**(ダウンロード)を選択します。
   b. **Installers** (インストーラ)を選択します。
   c. ダウンロードするファイルを選択します。



   - **Linux インストーラ**：`Contrast<version>-NO-CACHE.sh`
   - **Windows インストーラ**：`Contrast<version>-NO-CACHE-x64.exe`
3. **エアギャップでのインストールの場合**：ライブラリ・CVE データをダウンロード (904ページ)します。
   インターネットに接続できる場合は、データをダウンロードする必要はありません。インストール後に、ライブラリ・CVE データを自動で更新 (906ページ)するようにシステムを設定します。
4. Contrast のライセンスファイルをダウンロードします。
   ライセンスファイルには、SuperAdmin (957ページ)アカウントと通常のユーザアカウントが設定されています。Contrast アプリケーションのインストールを完了するには、ライセンスが必要です。
   a. ナビゲーションバーで、**Home**(ホーム)を選択します。
   b. **Licenses**(ライセンス)を選択します。
   c. ダウンロードするライセンスファイルを選択します。
5. ファイルをダウンロードしたら、Contrast をインストール (879ページ)します。

オンプレミス版 Contrast のインストール

重要
このインストール手順は、オンプレミス版専用です。

SaaS 版の Contrast をお使いのお客様は、本インストールを行わずにアプリケーションを検査 (46ページ)できます。エージェントをインストール (46ページ)すれば、検査を開始できます。

Contrast は、約 24 時間ごとにライブラリデータを更新します。インターネットアクセスがあれば、お使いのオンプレミス版 Contrast は、クラウドにホストされている Contrast データベースからデータを取得します。インターネットに接続できない環境(エアギャップでのインストール)の場合は、手動でデータをダウンロードしてください。

開始する前に

- Contrast Hub または curl コマンド (877ページ)を使用して、Contrast インストーラをダウンロード (878ページ)してください。
- インターネットに接続できない環境(エアギャップでのインストール)の場合は、手動でライブラリデータをダウンロード (904ページ)してください。
- システム要件 (875ページ)とサイジングの推奨事項 (876ページ)を確認してください。

手順

1. お使いのオペレーティングシステムで MySQL を実行するための共有ライブラリを事前に設定します。また、Linux の場合にはフォントをインストールするためにシステムパッケージの `fontconfig` も必要です。お使いのオペレーティングシステムのコマンドを実行してください。
   - **RHEL 8.6** の場合：

     ```
     [contrast@myserver ~]# dnf install -y ncurses-compat-libs libaio \
     fontconfig
     ```
   - **CentOS** または **RHEL 7** の場合：

     ```
     [contrast@myserver ~]# yum install -y libaio fontconfig
     ```
   - **Ubuntu** または **Debian** の場合：

     ```
     [contrast@myserver ~]# apt-get install -y libaio1 libaio-dev fontconfig
     ```
   - **Windows** の場合： MySQL を実行するために Microsoft Visual C++ 2015-2022 再頒布可能パッケージが必要です。
2. 特権ユーザとしてインストーラを実行します。
   - Windows の場合、インストーラを右クリックして**管理者として実行**を選択します。
   - Linux の場合、sudo コマンドを使用してインストーラを起動します。
3. ご利用の環境に合わせて、インストーラの問いに回答します(例えば、MySQL バックアップの作成 (945ページ)や JRE の設定 (892ページ)など)。
   Contrast の起動後でも設定はできます。インストールスクリプトの実行時に、以下のコマンドライン引数を使用して、インストーラの動作をカスタマイズできます。

コマンドライン引数	説明
`-h -help`	一般的なコマンドライン引数のヘルプを表示します。
`-c`	インストールをコンソールモードで実行します。
`-q`	インストーラを無人モードで実行します。
`-g`	インストールを GUI モードで実行します(Windows のみ)。
`-console`	インストーラを無人モードで実行し、Windows で `-console` 引数を指定すると、2 つ目のコンソールにインストーラの出力が表示されます。
`-overwrite`	インストーラに指定した上書きポリシーに関係なく、無人モードでインストーラが全てのファイルを強制的に上書きします。 **注意** これにより、設定が上書きされてデフォルト値に戻る場合があります。
`-dir`	無人モードでのみ有効で、Contrast をインストールするディレクトリを指定します。
`-Dinstall4j.debug`	デフォルトでは、インストーラは全ての例外を取得してクラッシュログを作成し、そのログファイルの場所をユーザに通知します。これはインストーラのデバッグ時に不便な場合があります。そのために、このシステムプロパティがデフォルトの仕組みをオフにして、例外を `stderr` に出力します。

コマンドライン引数	説明
`-Dinstall4j.keepLog=true` `-Dinstall4j.alternativeLogfile=[path]`	インストーラによって、全てのインストールとアンインストールで i4j_log という接頭辞が付いたログファイルが temp ディレクトリに作成されます。このログファイルは、デバッグに役立ちます。インストーラに **Install files** アクションがあり、正常に終了した場合は、ログファイルが *[installation dir]/.install4j/installation.log* にコピーされます。そうでない場合は、デフォルトでは、インストールやアンインストール終了後にファイルは削除されます。 `-Dinstall4j.keepLog=true` option を使用すると、ログファイルは削除されません。`-Dinstall4j.alternativeLogfile=[path]`オプションを使用すると、ログファイルは`[path]`で指定されたファイルにコピーされます。ここには絶対パス名を指定してください。ログファイルが既にインストールディレクトリにコピーされている場合は、どちらのオプションも無効です。
`-varfile (filename)`	使用する変数ファイルを指定します。無人モードでインストールする場合に、インストーラで設定するデフォルト値をカスタマイズできます。
`--skip-preflight`	事前チェック(現在のユーザが root であるか、依存関係があるか)をスキップします。このパラメータを使用する場合は、コマンドラインで渡す最初のパラメータでなければなりません。

注記
Contrast を分散環境で使用する場合は、セットアップ時に分散型の MySQL インスタンスを使用する必要があります。

重要
クライアントであるエージェントは、Contrast URL を使用して Contrast サーバと通信します。Contrast では、ホスト名を決定してこの値を事前に指定できますが、指定したホスト名がネットワーク上のクライアントで解決できない場合、サーバと通信できなくなります。

この値には、エージェントホストから到達可能な Contrast ホストかロードバランサを設定してください。

4. インストールが完了し、Contrast アプリケーションで初期設定が行われます。完了したことを確認するには、上記で指定した URL にアクセスします。

注記
Contrast のバージョンをアップグレード (902ページ)している場合は、ここで必要な更新タスクが行われます。

5. インストール後、初めて Contrast アプリケーションを起動した時に、ユーザインターフェイスにログインできるのは 2 ユーザだけです (889ページ)。デフォルトのスーパー管理者(SuperAdmin)とデフォルトの組織管理者(Admin)です。*http://<ContrastServer>:8080/Contrast* (*<ContrastServer>* はインストール時に指定した IP アドレスかホスト名)にアクセスして、両方のユーザとしてログインして、それぞれのパスワードを変更してください。

重要
アプリケーションを安全に保つために、これらのデフォルトのログインを無効にして新しいユーザを作成するか、少なくともデフォルトのパスワードを変更してください。

6. Contrast のインストールが完了したら、次はシステムの設定 (936ページ)(ライセンスの割当て、認証の設定、ユーザへの通知、レポートの実行など)を行ってください。
   ライブラリデータを最新に保つために、ライブラリデータを自動で更新 (906ページ)するようにシステムを設定してください。インターネットに接続できない環境(エアギャップでのインストール)の場合は、ライブラリデータを手動で更新 (904ページ)してください。

その他のオプション

Contrast のインストール後に、インストールを拡張することができます。

- 分散 MySQL データベース (883ページ)を使用する
- WAR ファイル (882ページ)を使用してデプロイする
- 分散環境 (886ページ)で導入する

WAR ファイルを使用して Contrast をデプロイ

この手順を使用すると、Contrast インストールのさまざまなコンポーネントを個別に管理できます。この方法を使用して Contrast をデプロイした後、既存の WAR ファイルを新しいファイルに置き換えることで、設定をアップグレードできます。

開始する前に

- Contrast をインストール (879ページ)します。

手順

1. Contrast をインストールしたディレクトリ(例、`/usr/local/contrast`)から、以下のファイルを含む圧縮ファイルを作成します。
   - `data/agents/`
   - `data/conf/`
   - `data/esapi/`
   - `data/.contrast`
   - `data/.initialized`
   - `data/cache/`
   - `data/contrast.lic`
   - `webapp/Contrast.war`

   例：以下の例は、Contrast ファイルを含む TAR ファイルを作成する方法を示します。

   ```
   $ cd /usr/local/contrast
   $ tar -czvf ~/ctdc.tar.gz data/agents data/conf data/contrast.lic data/
   esapi/ data/cache/ data/.initialized data/.contrast webapp/Contrast.war
   ```
2. Tomcat と Java をインストールします。
   - Tomcat は、Contrast インストーラに含まれるものと同じバージョンを使用してください。
   - サポート対象の Java バージョン (875ページ)
3. `contrast-data` ディレクトリをセットアップします。
   このディレクトリを作成するボリュームは、ログファイル、キャッシュ、ActiveMQ の永続性(パーシステンス)に対して十分な大きさである必要があります。最適なパフォーマンスを確保するには、システム全体に影響を与えずに、拡張を処理する別のボリュームを使用します。
   a. 以下のようなコマンドを使用して、`contrast-data` ディレクトリを作成します。

      ```
      $ mkdir /opt/contrast-data
      ```

      この例では、`/opt` ディレクトリにディレクトリが作成されますが、任意の場所に作成できます。
   b. 手順 1 で作成した圧縮ファイルを contrast-data ディレクトリに解凍します。
   c. 以下のコマンドなどを使用して、ディレクトリの内容を確認します。

```
ls /opt/contrast-data/conf
```

general.properties や database.properties という名前のファイルなどがあることを確認できるはずです。

d. アクセスの問題が無いようにするために、以下のようなコマンドで contrast-data ディレクトリの所有者とグループを変更します。

```
$ chown -R tomcat7:tomcat7 /opt/contrast-data
```

4. 設定を完了するには、JAVA_OPTS 環境変数で contrast.home と contrast.data.dir の場所を、圧縮ファイルを解凍した場所に設定します。
以下は、JAVA_OPTS 変数を設定する 1 つの例です。お使いの環境のドキュメントを参照して、この変数を設定する最適な方法を決定してください。

```
-XX:+UseTLAB
-XX:+UseCompressedOops
-XX:+UseConcMarkSweepGC
-XX:+UseParNewGC
-XX:CMSFullGCsBeforeCompaction=1
-XX:+CMSParallelRemarkEnabled
-XX:+PrintVMOptions
-XX:+PrintCommandLineFlags
-Xmx4g
-Xms4g
-server
-XX:MaxPermSize=768m
-Dcontrast.data.dir=/opt/contrast-data
-Dcontrast.home=/opt/contrast-data
-XX:+HeapDumpOnOutOfMemoryError
-Xloggc:/opt/contrast-data/gc.out
```

5. Contrast.war ファイルを Tomcat の webapps ディレクトリに配置します。
シンボリックリンクを作成するか、圧縮したファイルを解凍した場所から Contrast.war ファイルを Tomcat の webapps ディレクトリにコピーまたは移動します。
   - 以下は、Ubuntu 環境でファイルへのシンボリックリンクを作成する方法の例です。

```
$ sudo ln -s /opt/contrast-data/webapp/Contrast.war /var/lib/tomcat7/
webapps/Contrast.war
```

   - 以下は、Ubuntu 環境でファイルをコピーする方法の例です。

```
$ cp /opt/contrast-data/webapp/Contrast.war /var/lib/tomcat7/webapps/
Contrast.war
```

分散型 MySQL 環境の作成

オンプレミス版の既存のインストールで、外部の MySQL データベース(Windows と Linux の両方で動作するオープンソースデータベース)を使用することができます。例えば、これは Contrast の分散環境(886ページ)を使用する場合などに必要となります。

ヒント
運用担当者やデータベース担当者と協力して、安全で耐久性のあるインストールを行ってください。

Ansible のスニペットを使用して、Ubuntu 14.04 に MySQL をインストールすることもできます。

また、GPG キーファイルは MySQL からダウンロードすることもできます。Contrast では、バインドアドレス(bind_address)を「*」に変更していますが、ご利用の MySQL サーバとアプリケーションサーバの IP にバインドすることをお勧めします。ユーザを作成して、Contrast スキーマのみへアクセス、ホストの IP アドレスとサブネットに制限する権限を与えてください。

手順

以下の手順で、<jdbc.host><jdbc.port><jdbc.user>、<jdbc.pass>、<jdbc.schema>をご利用のホスト、ポート、ユーザ、パスワード、スキーマに置き換えてください。

1. データベースサーバのホストにサポート対象バージョン (875ページ)の MySQL をインストールして構成します。
2. Contrast のダウンタイムのためのメンテナンス時間を設けます。
3. 組み込みの MySQL データベースをバックアップ (945ページ)します。
4. MySQL に接続します。
   - **Windows：**

     ```
     mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema>
     ```
   - **Linux：**

     ```
     ./mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema>
     ```
5. 次のコマンドで Contrast のデータベースを作成します。

   ```
   create database <jdbc.schema>;
   ```
6. 次のコマンドで MySQL ユーザを作成します。

   ```
   CREATE USER '<jdbc.user>'@'%' IDENTIFIED BY '<jdbc.pass>';
   ```
7. 次のコマンドで Contrast ユーザに権限を付与します。

   ```
   GRANT ALL PRIVILEGES ON *.* to '<jdbc.user>'@'%';.
   ```
8. MySQL を終了(exit)します。
9. MySQL のバックアップを復元します。<backup_location>をバックアップファイルの場所に、<backup_filename>をバックアップファイル名に置き換えてください。
   - **Windows：**

     ```
     mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema> < \
     <backup_location>/<backup_filename>
     ```
   - **Linux：**

     ```
     ./mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema> \
     < <backup_location>/<backup_filename>
     ```
10. 暗号化プロパティエディタ (943ページ)で設定を更新します。 暗号化ファイルの $CONTRAST_HOME/data/conf/database.properties を編集します。database.type を検索してください。存在しない場合は、プロパティを新規に作成してください。このプロパティの値に distributed を指定し、使用する分散データベースを指すようデータベースの接続に関する値も更新します。

```
user@ubuntu:/opt/contrast/bin$ ./edit-properties  -e ../data/esapi/ -
f ../data/conf/database.properties
jdbc.type                                          : MYSQL
database.prod.dir                                  : /opt/contrast/data/db
jdbc.debug                                         : false
jdbc.pass                                          : pass
jdbc.schema                                        : contrast
jdbc.host                                          : ubuntu
database.bk.time                                   : 6:39:14
jdbc.port                                          : 3306
database.bk.enabled                                : false
database.enabled                                   : true
jdbc.url                                           : jdbc:mysql://
ubuntu:3306/contrast
jdbc.user                                          : contrast
database.bk.dir                                    : /opt/contrast/data/
backups/db
jdbc.dialect                                       : \
com.aspectsecurity.contrast.teamserver.persistence.CustomMySQL5Dialect
jdbc.driver                                        : com.mysql.jdbc.Driver

Enter the name of the property to edit [q to Quit]: database.type
Create new Property [database.type](y/N): y
Enter a value for the property: distributed

jdbc.type                                          : MYSQL
database.prod.dir                                  : /opt/contrast/data/db
jdbc.debug                                         : false
jdbc.pass                                          : pass
jdbc.schema                                        : contrast
jdbc.host                                          : ubuntu
database.bk.time                                   : 6:39:14
jdbc.port                                          : 3306
database.bk.enabled                                : false
database.enabled                                   : true
database.type                                      : distributed
jdbc.url                                           : jdbc:mysql://
ubuntu:3306/contrast
jdbc.user                                          : contrast
database.bk.dir                                    : /opt/contrast/data/
backups/db
jdbc.dialect                                       : \
com.aspectsecurity.contrast.teamserver.persistence.CustomMySQL5Dialect
jdbc.driver                                        : com.mysql.jdbc.Driver

Enter the name of the property to edit [q to Quit]:
```

注記

デフォルトの組み込みデータベースから分散環境に切り替える場合は、`database.bk.enabled` を `false` に設定することも必要です。分散データベース構成で Contrast を実行する場合、バックアップの設定は各自の責任で行ってください。

11. オンプレミス版を Windows システムにインストールしている場合は、MySQL に対する `contrast-server` サービスの依存関係を削除します。
    Contrast を再起動する前に、以下のコマンドを使用して、MySQL サービスに対する `contrast-server` サービスの依存関係を削除してください。

    ```
    sc config contrast-server depend= ""
    ```
12. Contrast を再起動 (890ページ)します。

分散環境における Contrast の構成

Contrast を分散環境で構成するには、データベースとアプリケーションサーバを別々のサーバに配置します。分散構成は、次のような場合に使用します。

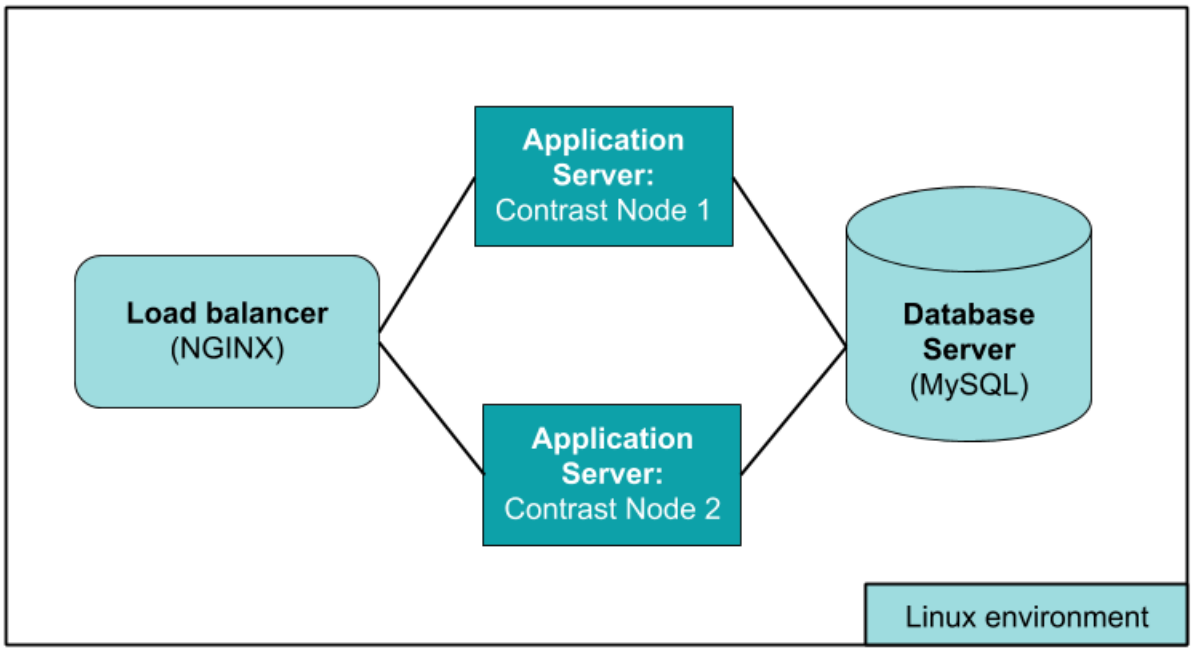
- 接続するエージェント数が 100 以上になる予定がある
  このような場合に分散構成を使用しないと、パフォーマンスの問題が発生する可能性があります。
- パフォーマンスとスケーラビリティの向上のためロードバランシングを利用したい
- 追加の管理や運用が必要

分散構成の例

ここでは、Linux 環境で Contrast を*/usr/local/contrast* にインストールする場合の設定を例として使用します。ご利用の組織で異なる環境を使用する場合や、サードパーティ製ソフトウェアのインストール先に関するガイドラインが異なる場合があります。

本項では、以下のように各サーバを使用する構成を例とします。

- 1 台のロードバランサー
- 1 台のデータベースサーバ
- Contrast アプリケーションを実行する 2 台のアプリケーションサーバ
  必要に応じて、さらにサーバを増やすことができます。



開始する前に

- Contrast を分散環境で構成するには、ご利用の環境およびその環境で問題なく処理できる負荷等について理解している必要があります。
  分散環境の導入が適しているか、もしくはお客様専用の Contrast の SaaS インスタンスを使用するのが最適かを判断するために、まずは弊社サポートまでお問い合わせください。

- **既に Contrast を使用している**場合は、既存のインスタンスを**アプリケーションサーバ 1**(Contrast Node 1)に使用し、分散データベース構成を使用していることを確認してください。続行する前に、次の操作を行う必要があります。
  - MySQL の分散環境をまだ構築していない場合は、分散型 MySQL を作成 (883ページ)します。
  - VERSION ファイル(/usr/local/contrast/VERSION)を参照して、Contrast アプリケーションのバージョン番号を取得します。次のコマンドで、Contrast のデータベースバージョンを取得します。

    ```
    select `version` from schema_version ORDER BY `installed_on` DESC LIMIT \
    1;
    ```

    上記のバージョン番号は、タスクの順序を決めるために必要です。
    例えば、アプリケーションのバージョンが 3.3.2 で、データベースのバージョンが 3.3.2.012 の場合、データベースのバージョン番号から.012 を除いても問題ないため、バージョンは同じと言えます。そのため、既存のアプリケーションサーバ A は、バージョン 3.3.2 の別のデータベース B で実行できます。新しいアプリケーションサーバ C にアプリケーションバージョン 3.4.2 をインストールして、データベース B に接続したい場合は、サーバ C にバージョン 3.4.2 をインストールする前に、サーバ A を停止するか更新する必要があります。
- **Contrast を新規にインストール**する場合は、次の操作が必要です。
- Contrast のシステム要件 (875ページ)に従い、データベースサーバに MySQL をインストールして構成します。リモート接続の権限を持つ MySQL ユーザを作成します。Contrast への接続には、この認証情報を使用します。
- **アプリケーションサーバ 1**(Contrast Node 1)に、分散データベース構成で Contrast をインストール (879ページ)します。
- 分散 MySQL インスタンス(Distributed [2])のオプションを選択します。データベースの作成をデータベースサーバに指定します。以下は、その例です。

```
Choose a MySQL database configuration.
Default [1, Enter], Distributed [2]2
Host
[localhost]
<enter hostname of Mysql server>

Port
[13306]
3306

Credentials
Username
contrast

Password
<enter mysql password for contrast account>
```

各分散サーバの設定

1. アプリケーションサーバ 1(Contrast Node 1)以下の設定ファイルをアーカイブして、TAR ファイルを作成します。
   - data/conf/
   - data/esapi/
   - data/contrast
   - data/initialized
   - data/contrast/lic
   - webapp/Contrast.war
   - VERSION

以下の例のようなコマンドを使用します。

```
$ cd /usr/local/contrast
$ tar -czvf ~/ctdc.tar.gz data/agents data/conf data/contrast.lic data/
esapi/ data/.initialized data/.contrast webapp/Contrast.war/.contrast \
VERSION
```

2. **アプリケーションサーバ 2**(Contrast Node 2)に Contrast をインストール (879ページ)します。
インストールの処理は、特権ユーザとして実行してください。
   - Windows の場合、インストーラを右クリックして、**管理者として実行**を選択します。
   - Linux の場合、sudo コマンドを使用してインストーラを起動します。

> **重要**
>
> ```
> Full or Application Only Install
> Choose the desired installation type:
> Full install [1, Enter], Application Server Only [2]
> 2
> ```

Application Server Only を選択し、前の手順で作成した TAR ファイルを使用します。分散構成に追加する他のアプリケーションサーバについても同じ手順を実行します。

3. Contrast で作成されたデフォルトユーザ (889ページ)でテストして、 Contrast の両方の**アプリケーションサーバ** (**Node 1** および **Node 2**)が機能することを確認してください。
4. 4 台目のサーバにロードバランサ(NGINX など)を設定します。NGINX を選択した場合は、基本的なインストール手順を使用してください。

> **注記**
> Contrast では、パフォーマンスを向上させるためにパーシステンスまたはスティッキーセッションが必要です。例えば、NGINX ロードバランサでは、IP ハッシュ方式を使用して、同じアドレスからのリクエストが使用可能な場合に、同じサーバに届くことを保証します。

5. サーバをセットアップしたら、ロードバランサを指定するように Contrast を設定する必要があります。この設定を行うには、各ノードの */data/conf/general.properties* ファイルを編集します。YAML 設定ファイルの *teamserver.url* の値をロードバランサの値に変更し、Contrast アプリケーションサーバを再起動します。
ロードバランサのヘルスチェックを行う場合は、以下の URL を使用します。
`<CONTRAST_SERVER>/Contrast/api/public/ng/information`
ここで、`<CONTRAST_SERVER>`は Contrast アプリケーションのサーバのホスト名になります。

> **重要**
> エージェントは、Contrast URL を使用してアプリケーションと通信します。Contrast はホスト名を判断することで、この値をあらかじめ設定します。ネットワーク上のクライアントが指定されたホスト名を解決できない場合、クライアントはサーバと通信できなくなります。この値には、エージェントが到達可能な Contrast のホストかロードバランサを設定してください。
>
> インストールが完了すると、Contrast の初期設定が始まります。完全に起動するまでに 2–3 分かかることがあります。

6. 設定の進行状況を確認するには、`server.log` と `contrast.log` をチェックしてください。サーバが正常に起動すると、`server.log` に次のようなメッセージが表示されます。

```
260916 20.18.25,837 {} {} {} INFO  (Server.java:303) Contrast \
TeamServer Ready -
```

Contrast の実行

Contrast を実行するには：

- **Windows**：Windows では、Contrast はシステムサービスとしてインストールされます。サービスは Windows Service Manager アプリケーションを通じて開始および停止できます。
- **Linux**：Contrast デーモンは `init.d` として登録されます。サービスを開始および停止するには以下のコマンドを使用します。

  ```
  /etc/init.d/contrast-server <start|stop|restart|status>
  ```

  または

  ```
  service contrast-server <start|stop|restart|status>
  ```

  親シェルとは別に Contrast サーバを起動するには、以下のコマンドを実行します。

  ```
  nohup /path/to/installation/contrast/bin/contrast-server start >/dev/
  null 2>1
  ```

この時点で、サーバログに対して tail を実行すると便利です。

```
$ tail -f $CONTRAST_HOME/logs/server.log
```

次に、アプリケーションログに対して実行します。

```
$ tail -f $CONTRAST_HOME/logs/contrast.log
```

Contrast が正常に起動すると、server.log に以下のメッセージが表示されます。

```
190116 21.22.15,703 {} {} {} INFO  (ConnectionTester.java:50) Received \
code 200 from TeamServer
190116 21.22.15,707 {} {} {} INFO  (ConnectionTester.java:60) Server start \
has been verified
190116 21.22.15,709 {} {} {} INFO  (Server.java:319) Contrast TeamServer \
Ready - Took 208323ms
```

デフォルト認証情報と SuperAdmin 認証情報の管理

Contrast をインストールするシステム管理者には、管理すべき認証情報が 3 セットあります。

- **Contrast Hub の認証情報**：新規のお客様は、ユーザ名とパスワードを設定するためのリンクが記載された電子メールが届きます。インストーラのダウンロード (879ページ)と Contrast Hub へのログインには、この認証情報が必要です。
  - **ユーザ名**：Contrast から、*example@domain.com* の形式でユーザ名が提供されます。これは、デフォルトの組織管理者(Admin)と同じユーザ名です。
  - **パスワード**：アクティベーションメールのリンクを選択した際に、このパスワードを作成します。
- **デフォルトのスーパー管理者(SuperAdmin)の認証情報**：SuperAdmin の認証情報は、ライセンスに含まれています。SuperAdmin ロール (957ページ)で Contrast アプリケーションを管理するために使用します。
  - **ユーザ名**：Contrast から、*contrast_superadmin@domain.com* の形式でユーザ名が提供されます。*domain* の部分は、お客様の会社のメールドメイン名になります。
  - **パスワード**：デフォルトのパスワードは `default1!` です。
- **デフォルトの組織管理者(Admin)の認証情報**：組織管理者は、この認証情報を使って、インストール (879ページ)後に Contrast にログインして、組織の設定や管理を行います。
  - **ユーザ名**：Contrast から、*example@domain.com* の形式でユーザ名が提供されます。これが、デフォルトの組織管理者のユーザ名です。

- **パスワード**：デフォルトのパスワードは、`default1!`です。

> **重要**
> ログインに成功したら、提供されたデフォルトのパスワードをすぐに変更してください。SuperAdmin のパスワードをリセットするには、Contrast UI を使用 (521ページ)するか、Windows (936ページ)または Linux (935ページ)のコマンドラインを使用してください。

Contrast の再起動

Contrast を再起動するには：

1. 再起動するには、以下のコマンドを使用します。
   - **Windows**：

     ```
     net stop "Contrast Server"
     ```

     サービスが完全にシャットダウンしたら、以下のコマンドを使用します。

     ```
     net start "Contrast Server"
     ```
   - **Linux**：

     ```
     sudo service contrast-server restart
     ```
2. この時点で、サーバログに対して tail を実行すると便利です。

   ```
   $ tail -f $CONTRAST_HOME/logs/server.log
   ```
3. 次に、アプリケーションログに対して実行します。

   ```
   $ tail -f $CONTRAST_HOME/data/logs/contrast.log
   ```
4. Contrast が正常に起動すると、server.log に以下のメッセージが表示されます。

   ```
   190116 21.22.15,703 {} {} {} INFO  (ConnectionTester.java:50) Received \
   code 200 from TeamServer
   190116 21.22.15,707 {} {} {} INFO  (ConnectionTester.java:60) Server \
   start has been verified
   190116 21.22.15,709 {} {} {} INFO  (Server.java:319) Contrast \
   TeamServer Ready - Took 208323ms
   ```

Contrast のアンインストール

インストールには、Contrast と、Java、Tomcat、MySQL などのすべての組み込みコンポーネントを安全にアンインストールするためのスクリプトが含まれます。このスクリプトは、Contrast インストールの root ディレクトリ内にパッケージ化されています。Unix では、このファイルは *uninstall* とラベル付けされた実行可能なスクリプトです。Windows では、コマンドファイルが *uninstall.cmd* というインストールディレクトリ内にパッケージ化されています。

アンインストールする前に以下を実行します。

- Contrast で提供されるデータベースのバックアップツールを使用して MySQL のバックアップを作成 (945ページ)します。
- Windows または Unix のサービススクリプトを使用して Contrast をシャットダウンします。

Windows を使用して Contrast をサーバから削除するには：

1. Windows エクスプローラーを開きます。
2. Contrast のインストールディレクトリに移動します。
3. *uninstall.exe* のファイルをクリックして実行します。インストールを実行したときと同じ(管理者としての)権限でアンインストールを実行します。
4. プロンプトに従ってアンインストールを実行します。

Unix を使用して Contrast をサーバから削除するには：

1. Linux コンソールを開きます。
2. ディレクトリ(cd)を Contrast のインストールディレクトリに変更します。
3. コマンド `uninstall` を実行します。
4. プロンプトに従ってアンインストールを実行します。

注記
アンインストールを実行すると、ファイルの大半は削除されます。ただし、管理者は以下のようないくつかの残りのファイルを手動で削除しなければならない場合があります。

- *Contrast HOME* ディレクトリ
- *Contrast DATA* ディレクトリ
- *Contrast LOGS* ディレクトリ
- *Contrast MYSQL* ディレクトリ

インストール後の作業

Contrast をインストールした後に、Contrast の環境を改善したい場合に設定できるオプションがいくつかあります。

インストール後の作業

インストール後の作業には、以下のようなオプションがあります。

- Tomcat の設定 (891ページ)
- Java ランタイム環境(JRE)の設定 (892ページ)
- HTTPS の設定 (892ページ)
- HTTP ヘッダの設定 (896ページ)
- MySQL のカスタマイズ (896ページ)
- レポート用ストレージの設定 (898ページ)
- ログの設定 (900ページ)
- Redis を共有キャッシュに使う (901ページ)

Tomcat の設定

インストール (879ページ)中、Contrast が実行されている組込みの Tomcat サーバが使用するメモリについて、いくつか値を設定します。

アプリケーションを追加したり、検出する脆弱性を増やしたりすると、パフォーマンスが低下する場合があります。その場合は、このサーバで許可されている最大メモリに達したことを示している可能性があります。

メモリ設定を増やすには：

1. `contrast-server stop` コマンドを実行して、Contrast サーバを停止します。

2. サーバが停止し、*Contrast/logs/contrast-stdout.log* の末尾に`[MysqldResource] shutdown complete` が記録されていれば、メモリ設定を安全に変更できます。
3. Contrast bin ディレクトリ *c:/Program Files/Contrast/bin* またはデフォルトの*/opt/Contrast/bin* にある *contrast-server.vmoptions* という名前のファイルを開きます。このファイルには以下のような内容が記述されています。

```
-XX:+UseG1GC
-XX:+UseStringDeduplication
-XX:+PrintVMOptions
-XX:+PrintCommandLineFlags
-XX:+UseContainerSupport
-XX:InitialRAMPercentage=50.0
-XX:MaxRAMPercentage=50.0
-XX:MinRAMPercentage=50.0
-Dcontrast.data.dir=/opt/contrast-data
-Dcontrast.home=/opt/contrast-data
-XX:+HeapDumpOnOutOfMemoryError
-Xloggc:/opt/contrast-data/gc.out
-server
```

4. 以下の値を更新できます。
   - **`Xms`**：起動時にサーバに割り当てられたメモリの量
   - **`Xmx`**：サーバが使用できる最大メモリ
   - **`MaxPermSize`**

   これらの値は、Contrast をホストするマシンで使用可能なメモリに応じて変わる可能性があります。
5. ファイルを保存し、`contrast-server start` コマンドを使用して Contrast を再度起動します。

ヒント
チューニングのヘルプについては、JVM のドキュメントを参照してください。

Java ランタイム環境(JRE)の設定

インストール (879ページ)中には、組み込みの JRE または事前インストールされた JRE のいずれかを使用できます。

JRE を設定するには：

1. テキストエディタを使用して*$CONTRAST_HOME/.install4j/pref_jre.cfg* を開きます。
2. 使用する Java バージョンへの完全なパスを追加します。例：

```
C:\Program Files\Java\jre11
```

HTTPS の設定

デフォルトでは、Contrast とエージェント間の接続に HTTP が使用されます。必要に応じて、Contrast とエージェントの両方の通信で HTTP を HTTPS に追加または置き換えます。これは Tomcat のコネクタ機能で実現できます。これを行う方法は 2 つあります。

- **Contrast HTTPS コネクタ**：証明書を Java キーストアに追加して、指定したポートで HTTPS 接続するように Contrast を設定します。
- **リバースプロキシ方式**：Apache や NGINX などの標準 Web サーバを Contrast サーバの前面に構築し、Contrast の AJP コネクタを使用してリバースプロキシとなるように設定します。

オンプレミス版の Contrast サーバで TLS バージョンや暗号スイートを変更する方法で説明しているように、設定をさらにカスタマイズすることもできます。

重要
以下の手順では、全体を通して 1 つのパスワードのみを使用することが重要です。CA(認証局)から提供されたファイルのいずれかがパスワードで保護されている場合は、そのパスワードを削除するか(CA はこれをサポートしています)、生成される Java キーストア(JKS)ファイルに同じパスワードを使用する必要があります。

Contrast HTTPS コネクタの使用方法

以下の手順で、オンプレミス版の Contrast サーバが使用する署名付き証明書を含む Java キーストア(JKS)を作成します。

注記
自己署名証明書で HTTPS コネクタを使用することもできます。

証明書署名要求(CSR)が必要な場合

この場合、まずキーストアを作成し、それを証明書署名要求(CSR)のベースとして使用します。

重要
CA(認証局)から、CSR の生成元と同じキーストアにインポートする必要があるファイルが提供されます。

1. Java の keytool コマンドを使用して、`contrast-server` というエイリアス名で証明書の秘密鍵と公開鍵を含む Java キーストア(JKS)ファイル(例、`contrast.jks`)を作成します。

   ```
   keytool -genkeypair -alias contrast-server -keyalg RSA -keystore \
   contrast.jks
   ```

   注記
   キーストアを作成する時に、使用している Java のバージョンによっては、最初のプロンプトで「What is your first and last name?」というメッセージが表示されて、名前と名字を入力するよう要求される場合があります。共通名(Common Name)、つまり証明書が発行される完全修飾名(FQDN)を入力します。例えば、名前と名字ではなく、`mydomain.com` などを使用します。

2. 証明書署名要求(CSR)ファイル(`contrast.csr`)を生成します。必要に応じて、DNS 名や IP アドレスのフィールドを追加して、証明書に Subject Alternative Names(サブジェクトの別名)を含めることができます。

```
keytool -certreq -alias contrast-server -file contrast.csr -keystore \
contrast.jks -ext san=dns:your_hostname.your_company.com,ip:10.0.0.1
```

3. 生成した証明書署名要求(CSR)ファイルを CA(認証局)に送信します。CA(認証局)から、PEM 形式の複数ファイルか、PCKS#7 形式の 1 ファイルのいずれかが提供されます。
4. CA から提供されたファイルを Java キーストア(JKS)にインポートします。以下の手順を、提供されたファイルの形式に合わせて使用してください。
   - **複数の PEM 形式ファイル**：これらのファイルの拡張子は、.CRT または.PEM です(PEM ファイルは読み取り可能なテキストとして開けます)。そのうち 1 つにはサーバ証明書が含まれ、その他のファイルはルート証明書と中間証明書が含まれることになります。
     これらの証明書は、上位(ルート証明書)から順にキーストアにインポートし最後にサーバ証明書をインポートする必要があります。サーバ証明書には、キーストアの作成時に使用したものと同じエイリアス名が必要です。例えば、`root.cer`、`inter.cer`、`server.cer` が提供された場合は次のようにインポートします。

     ```
     keytool -import -trustcacerts -alias root -file root.cer -keystore \
     contrast.jks
     keytool -import -trustcacerts -alias intermediate -file inter.cer -
     keystore contrast.jks
     keytool -import -trustcacerts -alias contrast-server -file server.cer -
     keystore contrast.jks
     ```

   - **PCKS#7 形式のファイル**：このファイルの拡張子は、.P7B か.CER で、場合によっては.CRT となります。このファイルには、必要な全てのルート証明書と中間証明書およびサーバ証明書が含まれています。サーバ証明書には、キーストアの作成時に使用したものと同じエイリアス名が必要です。例えば、`.p7b` は次の様にインポートします。

     ```
     keytool -import -trustcacerts -alias contrast-server -file \
     certificate.p7b -keystore contrast.jks
     ```

5. キーストアの設定が完了したら、テキストエディタで`<YourPath>/data/conf/server.properties` ファイルを開きます。`<YourPath>`は Contrast をインストールしたパスです。
   `<port>`、`<full path to>`、`<password>`をそれぞれ、ポート、JKS ファイルのパス、パスワードに置き換えてください。

   ```
   https.enabled=true
   https.port=<port>
   https.keystore.file=<full path to>/contrast.jks
   https.keystore.pass=<password>
   https.keystore.alias=contrast-server
   ```

   重要
   Windows を使用している場合は、JKS ファイルへのフルパスをエスケープする必要があります。例えば、以下のようになります。

   ```
   https.keystore.file=C:\\Program\ Files\\Contrast\\data\
   \conf\\ssl\\contrast-server.jks
   ```

   `http.enabled` および `ajp.enabled` オプションを `false` に設定し、HTTPS 経由で行われた接続のみを Contrast サーバで許可するようにすると便利です。
6. `<YourPath>/data/conf/general.properties` ファイルをテキストエディタで開き、`teamserver.url` プロパティの値を編集して変更を反映します。この変更を行った後、一度エージェントを手動で更新する必要があります。それ以降のエージェントに対する更新は、自動的に行われます。
7. **任意**：TLS バージョンや暗号スイートを変更します。

8. Contrast server サービスを再起動し、設定した HTTPS ポートになっていることを確認してください。

証明書署名要求(CSR)が不要な場合

この場合、CA(認証局)から提供されるファイルから新しいキーストアを作成します。既存のキーストアがある場合、新しいキーストアを作成する前にそれを削除するか名前を変更してください。

1. キーストアを作成するには、このいずれかを使用します。
   - `server.crt`、`priv.key`、`inter.crt` ファイルがある場合：以下のコマンドを使用して、ファイルを PKCS#12 形式に変換して、キーストアを作成します。

     ```
     openssl pkcs12 -export -out cert.pfx -inkey priv.key -in server.crt -
     certfile inter.crt -name "contrast-server"
     keytool -importkeystore -srckeystore cert.pfx -srcstoretype pkcs12 -
     destkeystore contrast.jks -deststoretype jks
     ```
   - PKCS#12 形式のファイルがある場合：以下のコマンドを使用して、キーストアを作成します。

     ```
     keytool -importkeystore -srckeystore cert.pfx -srcstoretype pkcs12 -
     destkeystore contrast.jks -srcalias <sourcealias> -destalias contrast-
     server -deststoretype jks
     ```
2. キーストアの設定が完了したら、テキストエディタで`<YourPath>/data/conf/server.properties` ファイルを開きます。`<YourPath>`は、Contrast をインストールしたパスです。
   `<port>`、`<full path to>`、`<password>`をお使いのポート、JKS ファイルのパス、パスワードに置き換えます。

   ```
   https.enabled=true
   https.port=<port>
   https.keystore.file=<full path to>/contrast.jks
   https.keystore.pass=<password>
   https.keystore.alias=contrast-server
   ```
3. `<YourPath>/data/conf/general.properties` ファイルをテキストエディタで開き、`teamserver.url` プロパティの値を編集して変更を反映します。この変更を行った後、一度エージェントを手動で更新する必要があります。それ以降のエージェントに対する更新は、自動的に行われます。
4. **任意：**TLS バージョンや暗号スイートを変更します。
5. Contrast server サービスを再起動し、設定した HTTPS ポートになっていることを確認してください。

リバースプロキシ方式の使用方法

リバースプロキシ方式で AJP(Apache JServ Protocol)を使用するには：

1. Contrast サーバが AJP プロトコルを使用して接続できるように設定します。テキストエディタで `CONTRAST_HOME/data/conf/server.properties` ファイルを開き、以下のオプションを設定します。

   ```
   ajp.enabled=true
   ajp.port=8009
   ajp.secretRequired=true|false
   ajp.secret=somesecret
   ```

   `ajp.port` には、サーバが着信接続を待ち受けるポート番号を設定します。サーバへのアクセスを AJP コネクタによる接続のみにする場合は、`http.enabled` と `https.enabled` オプションを無効にします。
   `secretRequired` を `true` にした場合は、`ajp.secret` に秘密キーワードを指定する必要があります。リクエストワーカには、秘密キーワードが必要になり、そうでない場合はリクエストが拒否

されます。ワーカ間で一致する秘密キーワードを設定してください。一致しない場合、`secretRequired` の設定に関わらずリクエストは拒否されます。
2. `server.properties` ファイルを更新したら、Contast server サービスを再起動して変更を有効にします。
3. フロントエンドサーバの構成については、お使いのサーバのドキュメントを参照して、AJP の設定方法を確認してください(Apache や NGINX の AJP に関するドキュメントなど)。

HTTP ヘッダの設定

Contrast で提供している Tomcat ソフトウェアを使用している場合は、以下の手順で HTTP ヘッダを設定できます。

HTTP ヘッダが有効な場合、HTTP レスポンスからのドキュメントをナビゲート可能な子プロセス(例えば`<iframe>`)に読み込ませるかどうかを制御できます。

HTML 標準の X-Frame-Options ヘッダには、HTTP ヘッダの設定に関する詳細があります。

手順

1. テキストエディタで`<YourPath>/data/conf/server.properties` ファイルを開きます。`<YourPath>`は Contrast をインストールしたパスです。
2. `servlet.response.xframe.options` プロパティに、以下の値のいずれかを指定します：
   - `SAMEORIGIN`:同一生成元からの組み込みを許可します。
   - `DENY`:組み込みを許可しません。
   - 値なし： ヘッダは省略されます。組み込みを許可します。

   > **注記**
   > `servlet.response.xframe.options` プロパティがない場合、デフォルト値の `SAMEORIGIN` が使用されます。
3. `Server.properties` ファイルを更新したら、Contrast サーバのサービスを再起動して変更を有効にしてください。

MySQL のカスタマイズ

> **注記**
> ここでの手順は、分散環境用の設定ではなく、組み込みの MySQL データベースに適用されるものです。

ご利用中の環境に合わせて、Contrast が MySQL データベースに指定したデフォルト設定の変更が必要になる場合があります。例えば、アップグレード後に、`innodb_buffer_pool_size` の設定の値の調整が必要になる場合などがあります。

デフォルトの MySQL の設定を変更するには、`my_extra.cnf` オプションファイルを使用します。このファイルは、オンプレミス版インストールの一部として Contrast インストーラに組み込まれています。

Contrast は、最初にデフォルトのオプションファイルをロードしてから、その後 `my_extra.cnf` ファイルに指定されたカスタム設定を適用します。

手順

1. `$CONTRAST_HOME/data/conf/my_extra.cnf` ファイルを検索してください。
2. 必要に応じて、MySQL の設定を追加または変更します。
3. Contrast を再起動 (889ページ)します。または、Microsoft Windows を使用している場合は、MySQL サービスを再起動します。

プロキシ構成の設定

オンプレミス版の Contrast でプロキシ構成を設定することによって、Contrast と同じネットワーク内に存在する(プロキシを必要としない)ツールと、ネットワーク外にある(Web プロキシアクセスを必要とする)ツールとを連携することができます。

設定方法

プロキシの設定を行うには、以下のいずれかを使用します。

- `contrast-server.vmoptions` ファイルにある JVM 引数
- Contrast Web インターフェイスでのプロキシ設定
  オプションファイルの JVM 引数は、Contrast Web インターフェイスの設定を上書きします。

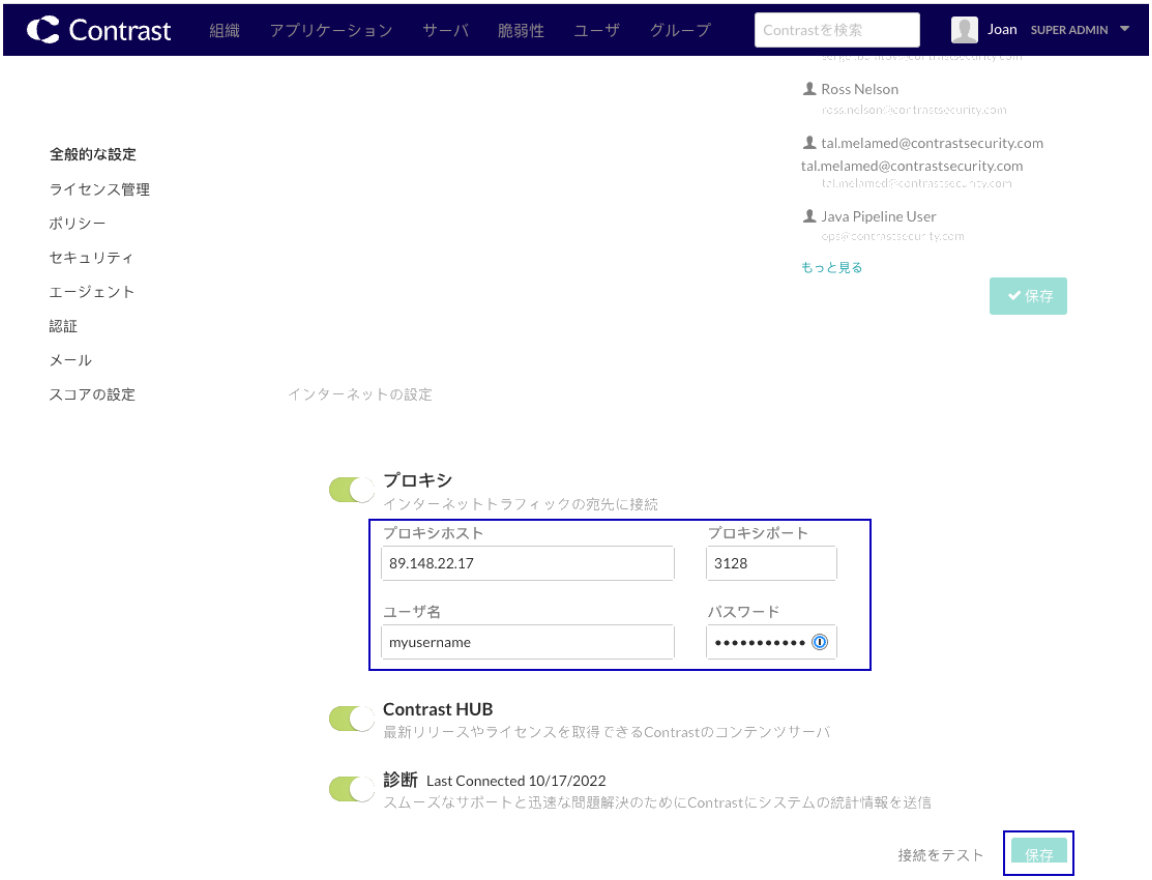
手順

- **JVM 引数を使用する(推奨される方法)：**
  1. Contrast の `bin` ディレクトリ(`c:/Program Files/Contrast/bin` またはデフォルトディレクトリの `/opt/Contrast/bin`)で、`contrast-server.vmoptions` を開きます。
  2. このファイルに、以下の JVM 引数を追加します(セキュリティ上、`https` 引数を使用するのが望ましいです)。
     - `https.proxyHost` または `http.proxyHost`
       プロキシサーバのホスト名です。この引数を使用して、外部にホストされているソフトウェアと通信します。
     - `https.proxyPort` または `http.proxyPort`
       プロキシサーバが、トラフィックをリッスンするポート番号です。
     - `http.nonProxyHosts`
       プロキシサーバを通過しないで直接接続するホストのリストを、縦線(|)で区切って指定します。この引数は、`https` および `http` の両方の設定で機能します。
       この引数を使用すると、Contrast のインストール先と同じネットワークにデプロイされているオンプレミスのホストを除外することができます。
       例えば、オンプレミス版の Jira をインストールしており、クラウド上の MS Teams とインテグレーションしている場合、以下の例のように `http.nonProxyHosts` 引数を使用すれば、Jira との連携にはプロキシサーバを経由しないよう除外することができます。

     ```
     -Dhttps.proxyHost=89.148.22.17
     -Dhttps.proxyPort=3128
     -Dhttp.nonProxyHosts=jira.mycompany.com|*.internal.mycompany.com
     ```

     この例では、プロキシホストは 89.148.22.17 で、トラフィックをリッスンするポート番号は 3128 です。この設定では、`jira.mycompany.com` のホストおよび `*.internal.mycompany.com` に一致するホストと通信する場合にプロキシサーバを経由しなくなります。
- **Contrast Web インターフェイスを使用する：**
  1. ユーザメニューで、**SuperAdmin** を選択します。
  2. **システムの設定**を選択します。
  3. 「インターネットの設定」で、**プロキシ**を選択します。

4. プロキシサーバのホスト名、ポート番号、ユーザ名、パスワードを指定します。



5. **保存**を選択します。

システムのレポート用ストレージを設定する

レポート用ストレージのオプションを設定するには、SuperAdmin として、`contrast/data/conf/general.properties` ファイルに以下のプロパティを追加します。

- **`reporting.storage.mode`**： ストレージモードのオプション値は、`DB` か `FILE_SYS` です。
- **`reporting.storage.path`**： 上記のストレージモードを `FILE_SYS` に設定した場合に必要です。

`reporting.storage.mode` の推奨設定は、`FILE_SYS` です。`DB` を設定する場合、ファイルはデータベースに格納され、不必要な競合がデータベースに追加されることになります。

`FILE_SYS` オプションを使用する場合には、全ての Contrast ノードがファイルパスにアクセスできるファイル共有サービスを設定する必要があります。そのパスを `reporting.storage.path` の値として指定してください。

注記
パスは、*/Users/user1/reporting* のように絶対パスを指定します。

Windows の場合は、コロンをエスケープしないと、パスが機能しません。例えば、次のようなパスは機能しません。

```
reporting.storage.path=C:\Contrast\data\reports
```

パスを機能させるには、以下のようにコロンの前後にスラッシュまたは 2 つのバックスラッシュを使用する必要があります。

```
reporting.storage.path=C\:\\Contrast\\datareports
```

デフォルトの設定では、コンプライアンス対応レポートから 1,250 件の脆弱性をエクスポートできます。1,250 件以上の脆弱性を含む大きなレポートを実行したい場合があるかもしれませんが、インスタンスのサイズによっては、Contrast のヒープ領域の問題が発生する可能性があります。

制限を増減するには、`general.properties` ファイルの `reporting.generation.limit` プロパティを設定して、Contrast を再起動してください。

Contrast のログ

Contrast アプリケーションでは、カスタムレベルに対応する Log4j がログフレームワークとして使用されます。

ユーザはログのしきい値を設定 (900ページ)し、ログファイルの宛先を管理し、Contrast 内で利用可能な各ログの概要を表示できます。

以下のログは、*$CONTRAST_HOME/logs* ディレクトリにあります。

- server.log
- catalina.out

以下のようなその他のログは、*$CONTRAST_HOME/data/logs* にあります。

- contrast.log
- ldap_ad.log
- migration.log
- audit.log
- mysql_error.log

以下の表に、Contrast 内の全てのプライマリログファイルを示します。

ログファイル	説明
audit.log	以下のような監査イベントを記録します。 • アプリケーションへのログインとログアウト • ほかのユーザへのなりすまし • 組織の切替え • 管理者ポータルへのアクセス • SuperAdmin アカウントによる Contrast の設定変更 • ユーザアカウントサービスの問題(アカウントのロック、パスワードの変更、ほか) • トレースの削除 • ライセンスの変更または期限切れライセンスの通知 • API キーの変更
console.log	デフォルトのアプリケーションイベントログ

ログファイル	説明
contrast-error.log	`stderr` に出力されるログメッセージ
contrast-stdout.log	`stdout` に出力されるログメッセージ
contrast.log	Tomcat の stdout または console ログ同様、contrast.log ファイルは Contrast 内で発生する最も重要なイベントを表示して通知またはデバッグをサポートします。これには以下の情報が含まれます。 • アプリケーション • サーバ • ライブラリ • トレース • ユーザ • サーバの例外発生時のデバッグ目的の Java スタックトレース
security.log	このログファイル(旧称 *esapi.log*)は、特定のプロパティファイルの読み込みなど、Contrast での重要なイベントを取得するために使用されます。
migration.log	Contrast アプリケーションに対して更新までの間に発生する全てのデータベース移行の概要が含まれます。Contrast バージョン、実行された移行スクリプト、スクリプトのステータスを参照します。

システムレベルでのログの設定

Contrast では、イベントやメッセージを記録する複数のログファイル (899ページ)が収集されます。

ユーザは、Contrast アプリケーションにパッケージ化された Log4j 設定をホストするために使用される *log4j2.xml* ファイルを設定でき(任意で Log4j カスタムレベルを適用でき)ます。

注意

フォーマットが構文的に正しいことを確認するため、このファイルに変更を加える前に。*変更が正しく入力されている場合は*サーバを再起動する必要はありません。

1. *$CONTRAST_HOME/data/conf* でファイルを検索します。
2. 以下に示すファイルの最初のパラメータは、定義された変数に基づいて設定を更新する監視インターバルです。デフォルトでは、Contrast は 60 秒ごとに口グ設定をチェックして更新を行います。

   ```
   <Configuration monitorInterval="60">
   ```
3. 必要に応じてファイルを編集します。

ヒント

Appender およびログイベントの配信の詳細については Log4j のドキュメントを参照してください。Contrast では主に Rolling File Appender を使用します。これは *OutputStreamAppender* であり、*fileName* パラメータで指定されたファイルに書き込み、*TriggeringPolicy* および *RolloverPolicy* に応じてファイルをロールオーバーします。

contrast.log のアペンダのサンプルファイルを以下に示します。これは最大 1GB のファイルサイズポリシーとロールオーバーするファイル数が 15 以内に指定されたデイリーアペンダです。また、このアペンダでは毎日ファイルが圧縮され名前が変更されます。

```
<RollingFile name="DAILY" fileName="${contrast.logs.dir}/
logs/contrast.log"
             filePattern="${contrast.logs.dir}/logs/
contrast.%d.%i.log.gz" immediateFlush="true">
    <PatternLayout>
        <Pattern>%d{ddMMyy HH.mm.ss,SSS} {%X{session.id}} \
{%X{user.name}} {%X{remote.addr}} %-5p (%F:%L) %m%n
        </Pattern>
    </PatternLayout>
    <Policies>
        <TimeBasedTriggeringPolicy/>
        <SizeBasedTriggeringPolicy size="1 GB"/>
    </Policies>
    <DefaultRolloverStrategy max="15"/>
</RollingFile>
```

ファイルの logger セクションには、どの Java パッケージが特定のログレベルで特定のアペンダにログするかが定義されています。

Redis を共有キャッシュに使う(オンプレミス版)

Redis サーバで共有キャッシュを使用するように Contrast を構成できます。

本項では、Redis サーバの設定についての詳細は説明しません。

Redis の Contrast プロパティ

以下のプロパティを使用します。

プロパティ名	説明
`cache.useredis`	Contrast がキャッシュに Redis を使用するかどうかを示す Boolean 値を指定します。
`contrast.cache.redis.db.index`	Contrast がキャッシュ情報を保存するデータベース· インデックスを整数値で指定します。
`contrast.cache.redis.proto`	Contrast が Redis サーバに接続する際に使用するプロトコルを指定します。
`contrast.cache.redis.host`	Redis サーバのホスト名または IP アドレスを指定します。
`contrast.cache.redis.port`	Redis サーバが新しいクライアントの接続を待ち受けるために使用する TCP/IP ポートの番号を指定します。
`contrast.cache.redis.password`	クライアントから Redis サーバへのアクセスを許可するために使用されるパスワードを指定します。
`contrast.cache.redis.client.name`	クライアントを識別するための文字列を指定します。

開始する前に

- 以下の情報を準備してください。
  - Redis サーバのホスト名または IP アドレス
  - 接続を待ち受けるために使用する TCP/IP ポートの番号
  - Redis サーバのユーザアカウントのパスワード
  - キャッシュの対象となるデータベースのインデックス
- Redis サーバが TLS(REDISS)を使用するように設定されていること を確認してください。

手順

1. `/data/conf/`フォルダ `contrast.properties` ファイルを作成します。
   このファイルに、`contrast_server` ユーザがアクセスできる権限があることを確認してください。
2. プロパティファイルに、Redis の Contrast プロパティを追加します。
   少なくとも、Redis サーバのホスト名、接続ポート番号、パスワードを設定してください。
   プロパティファイルの例：

   ```
   cache.useredis=true
   contrast.cache.redis.db.index=0
   contrast.cache.redis.proto=rediss
   contrast.cache.redis.host=contrast-redis-server.company.com
   contrast.cache.redis.port=6379
   contrast.cache.redis.password=changeme
   contrast.cache.redis.client.name=contrast
   ```
3. Contrast を再起動 (890ページ)します。
4. **設定を確認する**には`/data/logs/contrast.log` ファイルを確認します。
   このファイルには、現在の設定と Redis との操作に関する全ての情報が含まれます。確認すべきキーワードは以下の通りです。
   - RedissonCache
   - Redisson
   - CacheConfiguration

システムの更新およびアップグレード

Contrast Security では、定期的にオンプレミス版 Contrast のソフトウェアの更新やアップグレードを行います。

更新およびアップグレード

以下の各手順を参照してください。

- Contrast のアップグレード (902ページ)
- エージェントのアップグレード (904ページ)
- IP アドレスの更新 (904ページ)
- ライブラリデータの更新 (904ページ)
- ライブラリデータの自動更新 (906ページ)
- ライセンスの更新 (906ページ)

Contrast のアップグレード

Contrast のパッチやアップグレードは、オンプレミス版のインストーラファイルの一部として組み込まれてリリースされ、Contrast Hub(ハブ)からダウンロードできます。

インストーラは、指定されたシステムに Contrast の以前のバージョンがあるかを判断します。アップデートプログラムの処理を実行するか、別の場所でのインストールを実行するかを選択できます。前のバージョンのインストールが存在する場合は、並行してインストールを行い、別のポートで実行するように構成する必要があります。

開始する前に

`$CONTRAST_INSTALLATION/jre/lib/security` ディレクトリ にある `cacerts` ファイルをバックアップしてください。

アップグレードの処理によってこのファイルが上書きされて、ログインの問題が発生する可能性があります。これらの証明書は LDAP を統合 (931ページ)する際に使用されます。

手順

1. MySQL バックアップを作成 (945ページ)し、バックアップファイルを別のファイルシステムまたはドライブに保存して、リストア時の問題を回避します。インストーラはアップグレード処理の一環としてデータベースのバックアップを作成しようとしますが、安全のために手動で行ってください。また、*$CONTRAST_HOME/data/conf* にある全ての設定ファイルをバックアップしてください。
2. インストールプロセスを開始するときに、Contrast アプリケーションが実行されていることを確認します。この間、エージェントは脆弱性やライブラリのメッセージを送信し続けます。アプリケーションが自動的にシャットダウンを開始すると、エージェントはアプリケーションに到達できるまでメッセージの送信を延期します。
3. アップグレードプロセスは、最初に Contrast アプリケーションをインストール (879ページ)したプロセスとほぼ同じです。ただし、既存のインストールを更新するか、新規インストールを実行するかを選択するよう求められます。既存のインストールの更新を選択する必要があります。
4. アップグレードでは、最初にデータベースのバックアップが実行されます。データベースのサイズによって、この処理には数秒から数分までかかる場合があります。この処理の間、エージェントとエンドユーザは、アプリケーションにアクセスできているはずです。
5. 次に、アップデートにより、インストールディレクトリ配下に新しいファイルシステムがデプロイされます。これは主に、*$CONTRAST_HOME/webapps* ディレクトリに Contrast.war ファイルをデプロイすることで行われます。ファイルシステムの更新中は、アプリケーションにアクセスできません。
6. ファイルシステムの更新が正常に完了すると、アプリケーションが起動します。アプリケーションの起動中に、ログファイル(具体的には *migration.log* や *contrast.log* など)の内容を確認できます。ファイルシステムおよびデータベースの両方の更新に関するログエントリが順次書き込まれます。

   > **注記**
   > 設定ファイルやデータベースコンポーネントは、最初の起動ステップまで更新されません。

7. 更新成功の最初の目安となるのは、Contrast の Web インターフェイスにログインするか API リクエストを使用して、Contrast アプリケーションにアクセスできるようになることです。

   > **ヒント**
   > **ユーザメニュー**に、使用中のバージョンのリリース情報へのリンクが表示されます。

8. アップグレードの直後に *migration.log* の内容を確認してください。このログには、更新プロセスの一部で発生した問題が記録されます。

注記
アップグレードの数分後に、デプロイ済のエージェントが最新のエージェントバージョンへの更新を試みる場合があります。これらのエージェントでは、それぞれが再起動されて Contrast アプリケーションとの接続が確立されるまで、エージェント自体の更新は反映されません。

エージェントのアップグレード (オンプレミス版)

ほとんどのエージェントは、パブリックリポジトリからダウンロードすることができます。.NET エージェントと.NET Core エージェントのダウンロードは、Contrast Hub にアクセスします。

ダウンロードしたエージェントを、Contrast サーバの`$CONTRAST_HOME/data/agents` ディレクトリ下にある各エージェント言語のサブディレクトリにコピーします。 Contrast のオンプレミス版では、このディレクトリをサポートするよう自動的に構成されています。

手順

1. 適切なリポジトリから最新版のエージェントをダウンロードします。
   - **.NET：** Contrast Hub からダウンロード
   - **.NET Core:** Contrast Hub から、.NET Core エージェントまたは IIS 用.NET Core エージェントのインストーラをダウンロード
   - **Java：** Maven からダウンロード
   - **Node.js：** NPM からダウンロード
   - **Python：** PyPI からダウンロード
   - **Ruby：** RubyGems からダウンロード
   - **Go：** 直接ダウンロード (479ページ)
2. ダウンロードした各エージェントを、`$CONTRAST_HOME/data/agents` ディレクトリ内にある各エージェント言語に対応するサブディレクトリにコピーします。
   例えば、Java エージェントをダウンロードした場合、Java エージェントを `$CONTRAST_HOME/data/agents/java` ディレクトリにコピーします。
   Contrast を再起動する必要はありません。エージェントは動的にリロードされ、ダウンロードできるようになります。

IP アドレスの更新

Contrast アプリケーションのインストールを移動した場合やホスト名または IP アドレスを変更しなければならなかった場合は、以下のステップを実行する必要があります。

1. SuperAdmin として Contrast にログインします。
2. 右上で、**SuperAdmin > システムの設定 > 一般設定**を選択します。
3. **概要**パネルで、**TeamServer の URL** を *IP:port/Contrast* に変更します。
4. **保存**を選択します。
5. Contrast を再起動 (890ページ)して、変更を適用します。

SCA ライブラリデータを手動で更新

Contrast バージョン 3.7.4 以降、SCA ライブラリデータを Contrast Hub から手動でダウンロードできるようになりました。これは、インターネットにアクセスできない場合(エアギャップでのインストール)に便利です。

開始する前に

- Contrast Hub のアカウントが必要です。

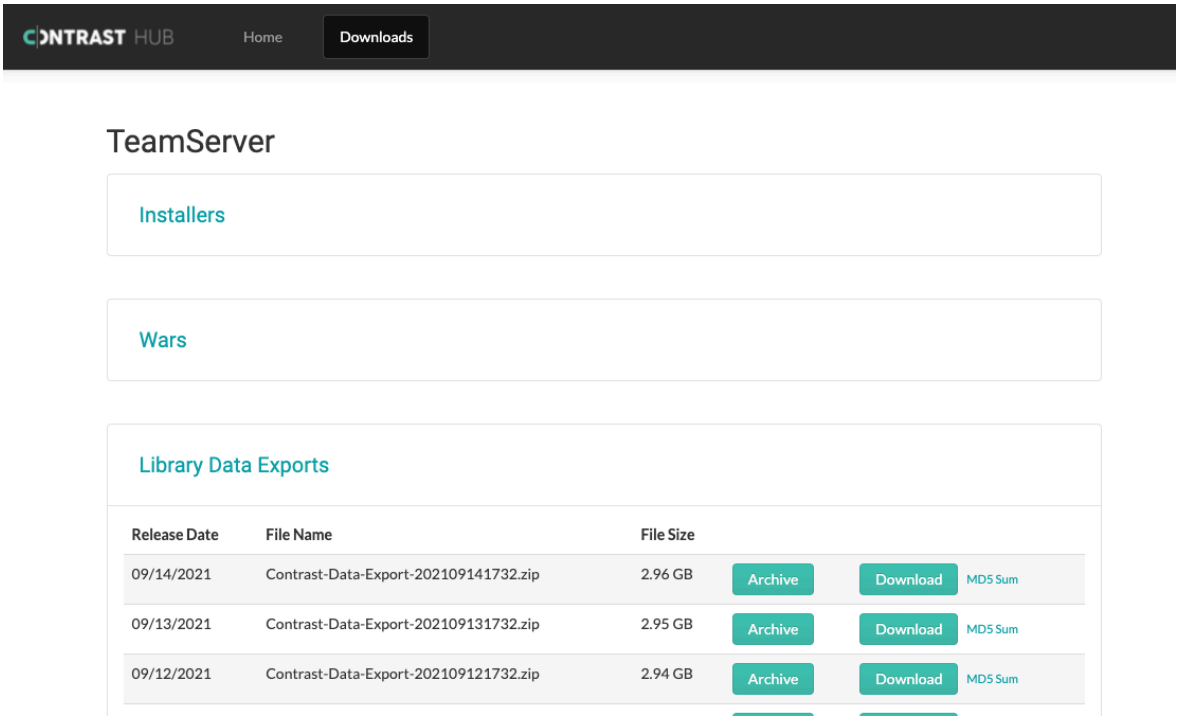
- 最適なパフォーマンスを得るために、ライブラリデータは毎月ダウンロードするよう計画してください。
- 複数のサーバを使用して分散環境でデプロイ (886ページ)している場合は、本項での手順を 1 つのインスタンスに対して使用してください。ダウンロードした同じライブラリデータファイルを複数のインストール先に使用できます。

重要

MySQL 8 を使用しているオンプレミス版のお客様は、Contrast で CSV ファイルを受け取ることができるように、システム変数 `local_infile` を **ON** に設定する必要があります。詳細については、LOAD DATA LOCAL のセキュリティ上の考慮事項を参照してください。

手順

1. Contrast Hub にログインします。
2. **Downloads**(ダウンロード)を選択します。
3. **Library Data Exports**(ライブラリデータのエクスポート)から、必要なアーカイブバージョンをダウンロードしてください。



4. ダウンロードした ZIP ファイルを解凍して、CSV ファイルを Contrast の *data/libraries* ディレクトリに配置します。 例：
   Unix： `/etc/contrast/data/libraries`
   Windows： `C:/ProgramData/contrast/data/libraries`
   ファイル名によっては非表示になっている可能性があるため、解凍した全てのファイルをこのディレクトリに移動したことを確認してください。
5. Contrast を再起動します。 (890ページ)
   Contrast が再起動すると、バックグラウンドでデータがインポートされます。CSV ファイルは、インポートされる度にフォルダから削除されます。
6. 各スクリプトの完了時に、*data/logs/contrast.log* ファイルに成功のメッセージが表示されます。例えば、次のようなメッセージです。

```
Beginning CSV import from 'C:\Program \
Files\Contrast\data\libraries\java.csv' into 'artifacts_java'Import \
temporary table 'artifacts_java' completed, time: 36.6886968s
```

SCA ライブラリデータを自動で更新

Contrast バージョン 3.6.4 以降、Contrast の SCA ライブラリデータを自動で更新するよう設定できるようになりました。

Contrast は、約 24 時間ごとにライブラリデータを更新します。新しく追加された CVE は 30 分ごとに更新され、その後 24 時間のスケージュールに含まれます。お使いのオンプレミス版 Contrast で、クラウドにホストされている Contrast データベースからデータが取得されます。

開始する前に

- 以下の URL へのアクセスを許可するようにファイアウォールを設定してください。
  `https://ardy.contrastsecurity.com/production`
- SuperAdmin ロールが必要です。

手順

1. Contrast Web インターフェイスに SuperAdmin ユーザとしてログインします。
2. ユーザメニューから、**システムの設定**を選択します。
3. **全般的な設定**を選択します。
4. 「インターネットの設定」の下にある、**Contrast HUB** をオンにします。

オンプレミス版の Contrast ライセンスの更新

Contrast のオンプレミス版をご利用のお客様は、ライセンス更新時に新しいライセンスファイルが必要になります。ライセンスファイルを更新する方法は、2 つあります。

- スーパー管理者(SuperAdmin)として Contrast アプリケーションにログインして、Contrast Web インターフェイスでライセンスを更新
- ローカルファイルシステムで、ライセンスファイルを置き換え(ライセンスの期限が切れている場合は、この方法で行う必要があります)

Contrast の Web インターフェイスでライセンスを置き換えるには：

1. SuperAdmin として ログインします。
2. 左側のナビゲーションで**ライセンス管理**を選択します。
3. パネルの下部にある、**このライセンスを更新**をクリックします。
4. デフォルトの認証情報 (889ページ)を入力して、Contrast Hub(ハブ)から最新のライセンスをダウンロードして適用します。
5. にアクセスできない場合は、**ライセンスをアップロード**をクリックして、表示されたフィールドにライセンスをペーストします。

6. **更新**を選択します。
7. Contrast を再起動 (890ページ)して、新しいライセンスの変更を適用します。

Contrast のファイルシステムでライセンスを置き換えるには：

1. 新しいライセンスを、Contrast Hub(ハブ)か御社のアカウント管理者、またはテクニカルサポート担当者から取得します。
2. 新しいライセンスファイルの名前を *contrast.new.lic* に変更します。
3. Contrast アプリケーションのサービスを停止します。
   - **Windows**：サービスコントロールパネルを使用します。
   - **Linux**：`sudo service contrast-server stop` を実行するか、ディストリビューションの設定に合わせて適切なコマンドを実行します。`ps aux | grep contrast` を実行し、Contrast アプリケーションの全てのプロセスが停止しており、実行中のプロセスがないことを確認します。*myslqld* がまだ実行中の場合、サービスの停止後にプロセス自体が終了するまでに数分かかることがあります。終了しない場合は、サポートにお問い合わせください。 **プロセスを強制終了しないでください。**

   重要
   現在の *contrast.lic* ファイルは移動しないでください。Contrast では、ライセンスを更新するために、古いライセンスファイルと新しいライセンスファイルの両方が必要です。
4. 新しいライセンスファイルを*<contrast_home>/data* ディレクトリに配置します。
   Linux の場合は、新しいライセンスファイルの所有者、グループ、権限がそのディレクトリにあるほかのファイルと同一であることを確認します(`ls -l` を実行すると、ディレクトリの内容が権限と所有者の情報とともに表示されます)。起動時に新しいライセンスファイルが使用されると、*contrast.lic.bak* という名前で、現在のライセンスのバックアップが同じディレクトリ内に作成されます。
   ライセンスファイルの所有者とグループを変更するには、`sudo chown contrast_service:contrast_service contrast.new.lic` を実行します。
   権限を変更するには、`sudo chmod 644 contrast.new.lic` を実行します。
5. 通常どおりに Contrast アプリケーションを起動します。
   - **Windows**：サービスコントロールパネルを使用します。
   - **Linux**：`sudo service contrast-server start` を実行するか、ディストリビューションの設定に合わせて適切なコマンドを実行します。
6. 新しいライセンスが有効になります。

Contrast アプリケーションの全てのインスタンスを更新するには、実行中のアプリケーションインスタンスごとに、前述のファイルシステムを使用する方法に従ってください。

システム運用の管理

組織の規模や Contrast をどのように管理するかによって、システムを運用する上で最適なロール (957ページ)を設定できます。

小規模な組織では、1 人のスーパー管理者(SuperAdmin ロール)が全てのシステム管理業務を行うことができます。業務を共有して行いたい場合は、追加のスーパー管理者(SuperAdmin)やサーバ管理者(ServerAdmin)を指定 (913ページ)することができます。

- **SuperAdmin**：スーパー管理者。Contrast のシステム管理を担当します。このロールは、1 人または複数の個人に割り当てることができます。ユーザメニューの **SuperAdmin** オプションを利用し、組織、アプリケーション、サーバ、脆弱性、ユーザおよびグループを設定できます。
- **ServerAdmin**：サーバ管理者。ユーザやグループにアクセスできない以外は、SuperAdmin と同じです。ユーザメニューの ServerAdmin オプションを利用し、組織、アプリケーション、サーバおよび脆弱性を設定できます。

エンドユーザやエージェントのライセンスを管理する個人やグループが別にある場合は、システムのアクセスグループを追加 (913ページ)して、ユーザに **SystemAdmin** ロールや **Observer** を指定できます。

- **SystemAdmin**：システム管理者。組織やグループの管理を担当します。ユーザメニューの SuperAdmin オプションを利用し、組織、アプリケーション、サーバ、脆弱性、ユーザおよびグループを設定できます。また、組織レベルで管理者になりすますことができます。
- **Observer**：システムのオブザーバ。組織、ユーザ、アプリケーション、グループおよびトレースの読取り専用権限があります。ユーザメニューの **Observer** オプションで読取り専用アクセスを利用し、組織、アプリケーション、サーバ、脆弱性およびユーザを参照できます。

注記
No Access を指定されたユーザは、指定された組織へのシステムレベルのアクセスがブロックされます。

複数の組織の管理

マルチテナント環境でデプロイするオンプレミス版のユーザは、同じシステム内で複数の組織をサポートするように Contrast を設定できます。インストールプロセス中に、デフォルトの組織が作成されます。その後、SuperAdmin 権限を持つユーザが追加の組織を作成できます。これを行うには：

1. SuperAdmin として Contrast にログインします。
2. **ユーザメニュー**で **SuperAdmin** を選択して、システム管理の選択肢を表示します。
3. ヘッダで**組織**を選択します。
4. **組織を追加**を選択します。
5. 新しい組織の有効な情報を入力し、組織の管理(Admin)ロールを担うユーザの認証情報を入力して、組織管理者を指定します。
6. ユーザが(上記のステップを通じて、または組織アクセスグループ (822ページ)のメンバーになることで)新しい組織へのアクセス権を付与されると、**ユーザメニュー**で組織名を選択することで組織間を移動できるようになります。

重要
組織管理者(Admin)が特定の組織の設定を変更するには、**組織の設定**を選択する前に、まず**ユーザメニュー**でその組織に切り替える必要があります。アクティブな組織は、**ユーザメニュー**の組織名の横に緑色のチェックマークが表示されます。

組織の追加/編集

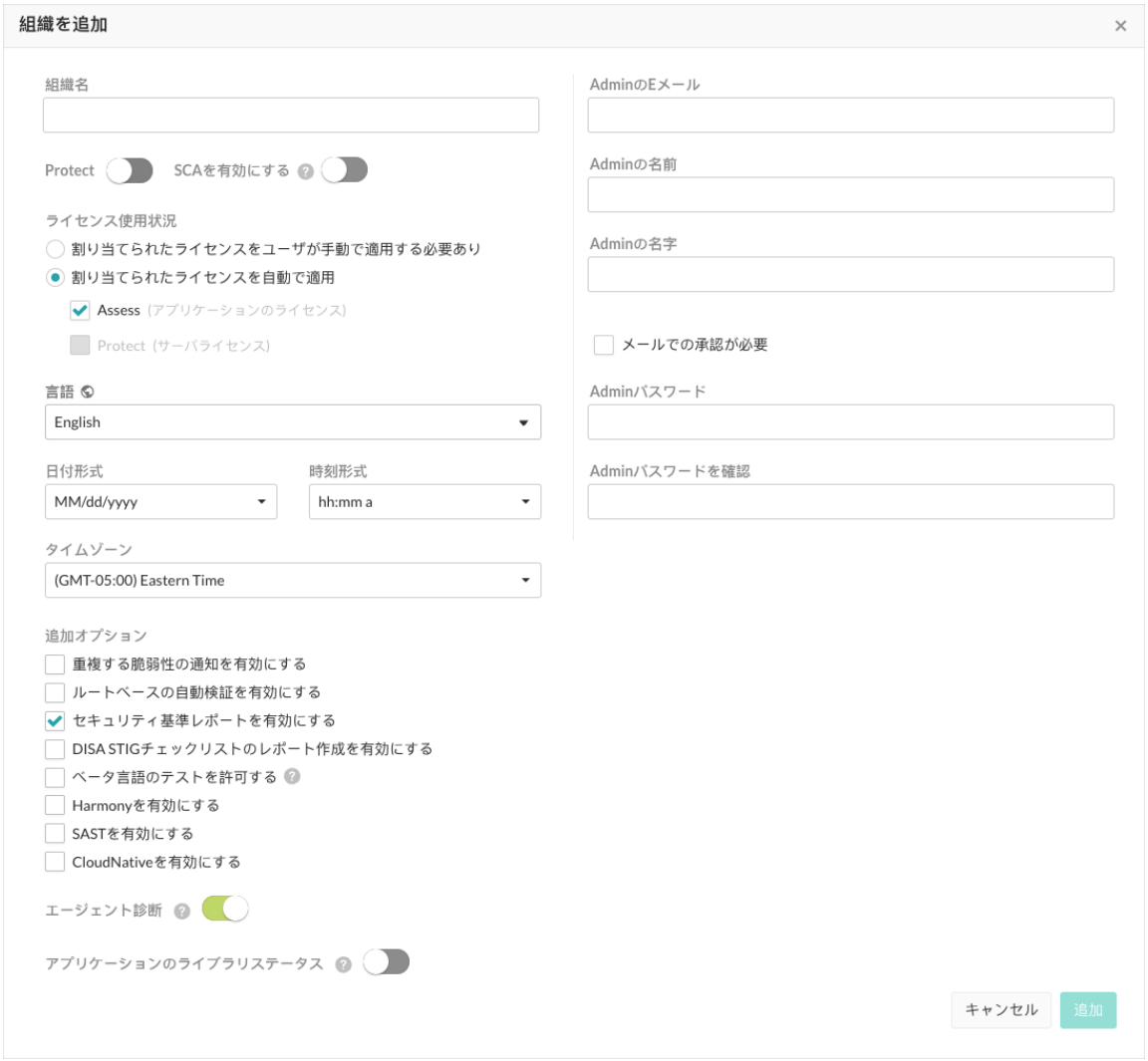
Contrast において、組織とは、ユーザとアプリケーションをビジネス上の共通の目的で関連付けた 1 つのグループです。Contrast はマルチテナントアーキテクチャを採用しており、Contrast のそれぞれのお客様はテナントであり、組織として表現されます。

開始する前に

- 組織を作成するためには、システム管理者 (957ページ)のロールが必要です。
- 全ての組織にはユニーク名が必要であり、また組織を管理する組織管理者 (955ページ)が必要です。

手順

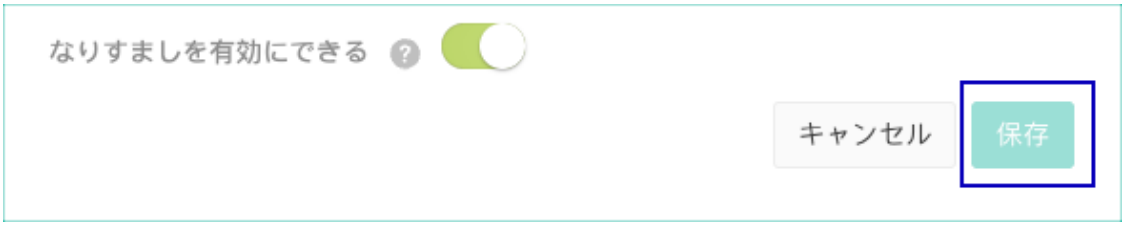
- **組織を追加するには**：
  1. **ユーザメニュー**で **SuperAdmin** を選択します。
  2. ナビゲーションバーで**組織**を選択し、次に**組織を追加**を選択します。
  3. 「組織を追加」の画面で、組織の情報を指定します。



     - 新規に作成する**組織名**を入力します。
     - 必要に応じて、トグルを使用して **Protect** を有効にします。
     - 必要に応じて、**SCA を有効にする**トグルを使用して SCA ライセンスを有効にします。
     - **ライセンス使用状況**の下でラジオボタンを使用して、割り当てられたライセンスの適用を手動と自動のどちらで行うかを指定します。
     - 組織のデフォルトの**言語**を選択します。
     - ドロップダウンを使用して、**日付形式**と**時刻形式**、**タイムゾーン**を選択します。
     - 重複する脆弱性の通知、ルートベースの自動検証 (786ページ)、DISA STIG チェックリストのレポート作成 (720ページ)、診断 (938ページ)など、その他の機能に関する追加オプションを選択します。
     - E メール、名前、パスワードなど、組織の管理者に関するプロファイル情報を入力します。
     - Contrast アプリケーションにメールサーバを設定している場合のみ、**メールでの承認が必要**のチェックボックスをオンにします。
  4. **追加**を選択すると、組織が作成されます。続けて、マルチテナントのサポートで必要な分だけ組織を作成してください。
- **組織を編集するには**：

1. **ユーザメニュー**で、**SuperAdmin** を選択します。
2. 編集する組織の名前を選択します。
3. 必要に応じて情報を更新し、**保存**をクリックして変更内容を保存します。 組織の編集画面でのその他の設定項目には、次のものがあります。
   - **CVSS 3.11** : Contrast は、CVSS 3.1 に対応しています。オンプレミス版(EOP)をご利用の場合は、SuperAdmin ユーザが **CVSS 3.1 を有効にする**をオンにすることで、CVSS 3.1 によるスコア評価を有効にできます。SaaS 版をご利用の場合、この設定を有効にするには Contrast サポートまでご連絡ください。

   - **なりすまし**：SuperAdmin ユーザが、**なりすましを有効にできる**の設定を変更することで、全ての組織に対してなりすまし (872ページ)を無効または有効にできます。デフォルトでは、この設定は有効になっています。
     この設定を無効にすると、組織管理者は自分の組織のなりすましを管理できなくなります。
     SaaS 版をご利用の場合、この設定を変更するには Contrast サポートまでご連絡ください。

システムレベルでのユーザと権限の管理

Contrast を設定してユーザを追加する前に、以下の項目について理解しておいてください。

- **ユーザ**：ユーザは、1 人ずつ (910ページ)登録することも、複数ユーザを一括で登録 (911ページ)することもできます。**ユーザメニュー**の **SuperAdmin** を選択して、ナビゲーションバーで**ユーザ**を選択すると、全てのユーザと各ユーザのステータス(アクティベーション待ち、有効/無効、セキュリティポリシーによりロックアウトなど)が表示されます。
- **認証**：独自の内部ディレクトリを使用するか、LDAP や Active Directory などの外部ディレクトリ (920ページ)を使用するように Contrast を設定できます。
- **グループと権限**：アクセスと権限は、ロール (953ページ)によって決まります。ほとんどのロールは、アクセスグループ (913ページ)で割り当てます。SuperAdmin ロールと ServerAdmin ロールの指定方法 (913ページ)は異なります。Protect 権限の付与 (915ページ)はシステムレベルで行えるだけでなく、新規ユーザ登録 (910ページ)時やユーザの一括登録 (911ページ)時にも行うことができます。

スーパー管理者(SuperAdmin ロール)およびシステム管理者(SystemAdmin ロール)は、システムレベルまたは組織レベルでユーザを追加できます。

システムグループにユーザを追加すると、システム管理用のインターフェイスにアクセスできるようになり、横断的な組織グループ内の全ての組織で操作ができるようになります。

1 つの組織にユーザを追加して、その組織にユーザのアプリケーションへのアクセスと権限を決定するロールを定義することもできます。

システムレベルでのユーザの追加/編集

システムおよび組織の管理者は、個別にユーザを作成したり、グループ、または Microsoft Active Directory (AD) (922ページ)や LDAP (926ページ)などと連携してユーザを作成することができます。

開始する前に

- システム管理者(SystemAdmin)または組織管理者(Admin)ロールが必要です。

- 全てのユーザには、デフォルトの組織とその組織内でのデフォルトロールが必要です。SuperAdmin と ServerAdmin ロール (913ページ)の指定は異なります。
- 個々のユーザを追加する場合、または複数のユーザを一括で追加する場合、ユーザに Protect 権限を付与することもできます。

手順

1. スーパー管理者(SuperAdmin)またはシステム管理者(SystemAdmin)としてログインします。
2. **ユーザメニュー**で **SuperAdmin** を選択します。
3. ナビゲーションバーで、**ユーザ**を選択します。
4. **ユーザ名**を選択して既存ユーザを編集するか、**ユーザを追加**を選択して新規ユーザを追加します。
5. 表示されるフィールドに、ユーザの**名前**、**名字**、**E メールアドレス**を入力します。
6. パスワードを要求する代わりに、メールによるアクティベーションを使用する場合は、**メールでの承認が必要**を選択します。
7. ユーザに適用するシステムロール (957ページ)を選択します。
   デフォルトのロールは、**None**(なし)です。
8. ユーザが所属する**組織**を選択します。
9. デフォルトの**組織ロール** を選択します。
10. カスタム、またはデフォルトの**アプリケーションアクセスグループ**を選択します。
    Contrast には、以下のデフォルトグループがあります。
    - **View**：このグループのメンバーは、Contrast インターフェイスに読み取り専用でアクセスし、スコア、ライブラリ、脆弱性、コメントなどを参照できます。
    - **Edit**：このグループのメンバーは、検出結果の修復、タグの追加、脆弱性の管理、属性の編集、アプリケーションのマージ、アプリケーションの追加· 削除、サーバの作成などが可能です。
    - **Rules Admin**：このグループのメンバーは、アプリケーションのルールとポリシーの編集、Protect の有効化、通知とスコアの管理を行うことができます。
    - **Admin**：このグループのメンバーは、組織の設定を構成し、管理することができます。
11. **日付形式**、**時刻形式**、**タイムゾーン**を選択します。
12. 組織管理者が組織レベルでユーザ設定を変更できるようにするには、**組織の設定を使用**を選択します。
    このチェックボックスはデフォルトで選択されています。
    システムレベルでユーザ設定を作成するには、このチェックボックスをオフにします。
    a. **組織の設定を使用**をオフにします。
    b. ユーザを Contrast Web インターフェイスではなく、API のみの利用に制限するには、**API のみのユーザ**を選択します。
    c. ユーザに Assess データの参照と使用を許可するには、**アクセス**をオンにします。
    d. ユーザに Protect データの参照と使用を許可するには、**Protect** をオンにします。

ヒント
組織レベルで Protect 権限を付与 (915ページ)することもできます。

13. **追加**または**保存**を選択します。

組織への複数ユーザの追加

CSV ファイルを使用して、複数のユーザを組織に追加することができます。

開始する前に

- オンプレミス版をご利用の場合は、SuperAdmin ロールが必要です。
- SaaS 版をご利用の場合は、組織の Admin ロールが必要です。

手順

1. ユーザページにアクセスします。
   a. SaaS 版をご利用の場合、メニューで**組織の設定**を選択したら、**ユーザ**を選択します。
   b. オンプレミス版をご利用の場合、SuperAdmin メニューより、ナビゲーションバーで**ユーザ**を選択します。
2. 推奨情報を入力したスプレッドシートを作成して、CSV ファイルとして保存します。
   - 各ユーザには必須フィールドを含めます。
   - 全フィールドの見出しと値は、以下の表に示す通りのフォーマットにします。
   - 任意のフィールドについては、新しい列を追加してください。

ヒント

アップロードアイコンにカーソルを合わせ、ツールチップのリンクを選択すると、CSV テンプレートをダウンロードできます。

CSV のフィールド：

フィールド名	必須	値
First Name	必須	ユーザの名前
Last Name	必須	ユーザの名字
Email または Username	必須	Contrast 内部ディレクトリ(デフォルト)を使用している場合は、ユーザの E メールを入力します。 外部ディレクトリ使用している場合は、CSV で **Email** を **Username** に変更して、外部ディレクトリと完全に一致するユーザ名を入力します。
Organization UUID	オンプレミス版をご利用の場合は必須	この値は、組織の設定の一般情報 (820ページ)から取得します。
Organization Role	必須	値は、`View`、 `Edit`、 `Rules_admin` または `Admin` になります。
Date Format	任意	デフォルトの値は、組織の設定の日付形式で、`MM/dd/YYYY` などになります。
Time Format	任意	デフォルトの値は、組織の設定の時刻形式で、`hh:mm a` などになります。
Timezone	任意	デフォルトの値は、組織のタイムゾーンです。
Protect	任意	デフォルトの値は、`Off`(無効)です。
Groups	任意	値は、`View`、`Edit`、`Rules Admin`、`Admin` またはカスタムグループ名です。複数のグループ名は、`GroupA&&GroupB&&GroupC` の形式にします。
Language	任意	デフォルトは、組織で設定 (820ページ)した値です。
System Administration	任意	デフォルトの値は、`Off`(無効)です。

フィールド名	必須	値
Email Activation	任意	値が `None`(なし)の場合、デフォルトは `Required Password`(パスワードが必要)になります。
Password	任意	Email Activation フィールドに `false` を指定した場合、このフィールドは必須です。
Api Only	任意	デフォルトの値は、`Off`(無効)です。
Access	任意	デフォルトの値は、`On`(有効)です。

3. **ユーザを追加**の横にある黒い**アップロードアイコン**を選択し、作成した CSV を選択します。スプレッドシートのアップロード処理が開始したら、そのページを離れて Contrast の他の処理を続けることができます。アップロードが成功すると、アップロードされたユーザ数を含む確認メッセージが表示されます。アップロードが失敗すると、スプレッドシートのエラーの原因を示すエラーメッセージが表示されます。

SuperAdmin または ServerAdmin の指定

SuperAdmin には、最高レベルのシステム管理権限があります。

ServerAdmin には、ユーザとグループにアクセスできないことを除いて、SuperAdmin と同じ権限と機能があります。

少なくとも 1 人のユーザを SuperAdmin として指定する必要があります。複数のユーザを SuperAdmin に指定する場合は、ログインを共有しないでください。代わりに以下を実行してください。

1. SuperAdmin としてログインします。
2. **ユーザメニュー > SuperAdmin > ユーザ**を選択します。
3. SuperAdmin に指定するユーザを検索します(名前、E メール、組織で検索するか、一覧で名前を探してください)。
4. ユーザ名を選択すると、**ユーザを編集**の画面が開きます。
5. **システム管理**フィールドで、**SuperAdmin** または **ServerAdmin** を選択します。
6. **保存**を選択します。

ヒント

SuperAdmin または ServerAdmin に指定された全てのユーザは、右上にある**ユーザメニュー**で **SuperAdmin** を利用できます。また、**SuperAdmin > ユーザ**の一覧で、ユーザ名の横に小さい鍵のアイコンが表示されます。鍵にカーソルを合わせると、割り当てられているロールを確認できます。

システムアクセスグループの追加/編集/削除

注記

システムアクセスグループは、オンプレミス版のお客様のみが利用できます。システムアクセスグループを追加するには、SuperAdmin である必要があります。

システムアクセスグループを追加するには：

1. **ユーザメニュー > SuperAdmin > グループ**を選択します。
2. 既存のグループを選択して編集するか、**グループを追加**を選択して新しいグループを作成します。

> **ヒント**
> グループを検索するには、クイックフィルターのドロップダウンまたは左上の検索フィールドを使用します。または、各列の上部にある上下方向の矢印を使用して並び替えを行います。

3. 画面で以下の項目に入力します。
   - **グループ名**：このグループに割り当てる権限、機能、目的を反映したものを選んでください。
   - **種類**： **System**(システム)を選択します。

   > **ヒント**
   > 組織レベルでアクセスグループを追加する (822ページ)こともできます。 ただし、システムレベルでアクセスグループを追加すると、組織間を横断できるグループを作成することができます。
   >
   > 横断的な組織グループは、それぞれが独自の組織を持つ複数の事業部門をサポートするセキュリティチームなどには便利です。
   >
   > 横断的な組織グループのメンバーは、ユーザメニューで名前を選択することで組織を切り替えることができます。

   - **システムアクセス**：このグループでアクセスできる組織を選択します。
   - **ロール**：このグループのメンバーが該当の組織内で持つべきシステムロール (957ページ)を選択します。
   - さらに組織とロールを追加するには、**システムアクセスを追加**を選択します。
4. **メンバー**では、フィールドに文字を打鍵するとユーザが表示されるので、グループに割り当てる 1 人以上のユーザを選択します。メンバーを削除するには、X を選択します。

   > **注記**
   > ユーザは、複数のグループに所属することができます。特定の組織にアクセスするために、その組織内で作成される必要はありません。

5. 完了したら、**追加**を選択して新しいグループを作成するか、既存のグループを編集している場合は**保存**を選択します。このグループに追加したメンバーに、それぞれのロールに対応する権限が付与されます。

重要

ユーザが、全てのアプリケーションや組織のロールに対して競合する 2 つのグループに割り当てられた場合は、アクセスの制限が最も厳しいロールが適用されます。

ユーザに表示される (523ページ)のは、組織レベルとアプリケーションレベルのグループのみであることに注意してください。アクセスレベルが不明な場合は、システムレベルでより厳しい権限が適用されている可能性があります。

ただし、特定のアプリケーションに割り当てられたロールに関しては、全てのアプリケーションに割り当てられたロールより制限が緩くても、特定のアプリケーションに割り当てられたロールが、全てのアプリケーションに割り当てられたロールよりも優先されます。

ユーザが 2 つのカスタムグループに割り当てられ、同じアプリケーションに対してロールが割り当てられている場合は、権限が最も少ないロールが適用されます。

システム (957ページ)、組織 (955ページ)、アプリケーション (953ページ)の各ロールは、制限の緩いものから厳しいものの順に表示されます。

以下はロールの権限が競合する例ですが、グループ 2 の権限が優先されます。

グループ 1	グループ 2(優先されます)
全てのアプリケーションに対して**アプリケーションの Edit** ロール	全てのアプリケーションに対して**アプリケーションの View** ロール
全てのアプリケーションに対して**組織の View** ロール	Red アプリケーションに対して**アプリケーションの Admin** ロール
Red アプリケーションに対して **Rules Admin** ロール	Red アプリケーションに対して **No Access**

ヒント

グループを削除するには、**ユーザメニュー > SuperAdmin > グループ**を選択します。削除したいグループを探し、その行の削除アイコンを選択します。

この操作を確定すると、グループが削除され、そのグループによって提供されたアクセス権が、そのグループに割り当てられた全てのユーザから取り消されます。

Protect 権限の付与(オンプレミス版)

オンプレミス版をご利用のお客様は、1 つまたは複数の組織で、全てのユーザロールや一部のユーザロールに Protect データへのアクセス権限を与えることができます。

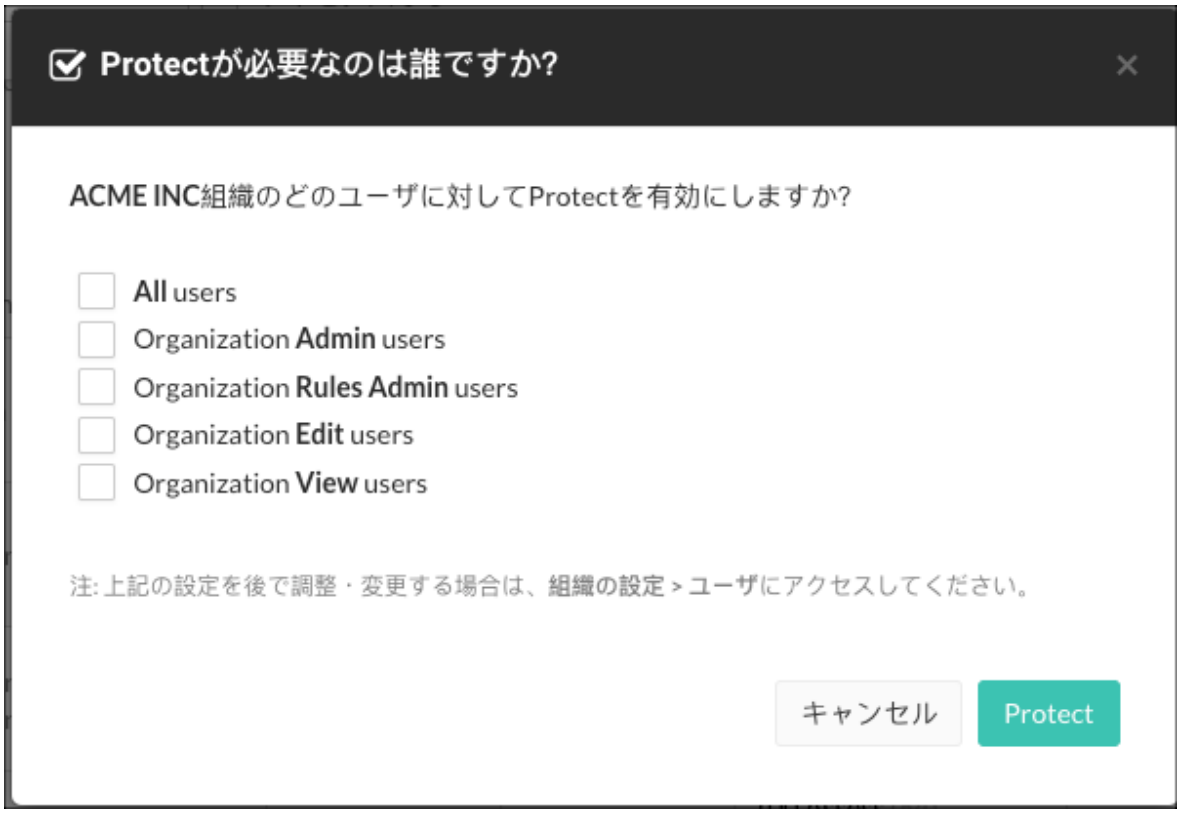
開始する前に

- SuperAdmin ロールが必要です。
- Protect データへのアクセスが必要となるユーザが所属するのは、どの組織であるかを確認してください。
- 組織内で Protect データへのアクセスが必要なのは、どのユーザロールであるか確認してください。

手順

1. SuperAdmin としてログインします。
2. ユーザメニューで **SuperAdmin** を選択します。
3. Contrast Web インターフェイスのナビゲーションバーで**組織**を選択します。
4. Protect を有効にする組織を検索します。該当する組織の行の右端にある Protect 列で、トグルボタンをオン(緑)にします。

5. 「Protect が必要なのは誰ですか？」の画面で、Protect データの参照とアクセス権限が必要なロールを選択します。



All users(全てのユーザ)または特定のユーザロールを選択します。
個々のユーザ (910ページ)に対して Protect のアクセスを有効または無効にすることもできます。

6. **Protect** を選択します。
この変更を行うと、選択したロールのあるユーザが Protect データにアクセスできるようになります。

SSO を使用してユーザをグループに自動追加

シングルサインオン(SSO)を使用してユーザをグループに自動的に追加できます。

1. IDP の SAML 設定を以下のように更新します。

```
<saml2:AttributeStatement \
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
```

```
    <saml2:Attribute Name="contrast_groups" \
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
                              xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
                              xsi:type="xs:string"
                              >GROUP1</saml2:AttributeValue>
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
                              xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
                              xsi:type="xs:string"
                              >GROUP2</saml2:AttributeValue>
        ...
    </saml2:Attribute>
</saml2:AttributeStatement>
```

重要

contrast_groups に列挙する属性値は、既存のグループ名と完全に一致する必要があります。Contrast では、この属性に列挙された値に基づいて新しいグループは作成されません。

2. 次に Contrast Web インターフェイスの組織の設定 (818ページ)で、**シングルサインオン**を選択し、画面の下部にあるチェックボックスを使用して、以下のいずれかまたは両方を有効にします。
   - **SSO ログイン時にユーザを Contrast グループに追加** : ログイン時に、Contrast は SAML アサーションの contrast_groups 属性に列挙されているグループにユーザを追加します。
   - **SSO ログイン時にユーザを Contrast グループから削除** : ログイン時に、Contrast は SAML アサーションの contrast_groups 属性に列挙されていないグループからユーザを削除します。

参考

- NameID としてのユーザの E メール

```
<md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</
md:NameIDFormat>
```

- 名前と名字

```
<saml2:Attribute Name="http://schemas.xmlsoap.org/ws/2005/05/identity/
claims/givenname"
                              \
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified"
                              >
                <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema"
                                      xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
                                      xsi:type="xs:string"
                                      >Dan</saml2:AttributeValue>
            </saml2:Attribute>

<saml2:Attribute Name=" http://schemas.xmlsoap.org/ws/2005/05/identity/
claims/surname"
                              \
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified"
                              >
```

```
                <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema"
                                      xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
                                      xsi:type="xs:string"
                                      >Dan</saml2:AttributeValue>
            </saml2:Attribute>
```

- ユーザグループ管理

```
<saml2:AttributeStatement \
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
<saml2:Attribute Name="contrast_groups" \
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
<saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:type="xs:string">GROUP1</saml2:AttributeValue>
<saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:type="xs:string">GROUP2</saml2:AttributeValue>
...
</saml2:Attribute></saml2:AttributeStatement>
```

関連項目

Okta を使用してユーザとグループのプロビジョニングを設定する

グループにユーザを自動的に追加するように ADFS を構成する

デフォルト認証情報と SuperAdmin 認証情報の管理

Contrast をインストールするシステム管理者には、管理すべき認証情報が 3 セットあります。

- **Contrast Hub の認証情報**：新規のお客様は、ユーザ名とパスワードを設定するためのリンクが記載された電子メールが届きます。インストーラのダウンロード (879ページ)と Contrast Hub への口グインには、この認証情報が必要です。
  - **ユーザ名**：Contrast から、*example@domain.com* の形式でユーザ名が提供されます。これは、デフォルトの組織管理者(Admin)と同じユーザ名です。
  - **パスワード**：アクティベーションメールのリンクを選択した際に、このパスワードを作成します。
- **デフォルトのスーパー管理者(SuperAdmin)の認証情報**：SuperAdmin の認証情報は、ライセンスに含まれています。SuperAdmin ロール (957ページ)で Contrast アプリケーションを管理するために使用します。
  - **ユーザ名**：Contrast から、*contrast_superadmin@domain.com* の形式でユーザ名が提供されます。*domain* の部分は、お客様の会社のメールドメイン名になります。
  - **パスワード**：デフォルトのパスワードは `default1!` です。
- **デフォルトの組織管理者(Admin)の認証情報**：組織管理者は、この認証情報を使って、インストール (879ページ)後に Contrast にログインして、組織の設定や管理を行います。
  - **ユーザ名**：Contrast から、*example@domain.com* の形式でユーザ名が提供されます。これが、デフォルトの組織管理者のユーザ名です。
  - **パスワード**：デフォルトのパスワードは、`default1!` です。

重要
ログインに成功したら、提供されたデフォルトのパスワードをすぐに変更してください。SuperAdmin のパスワードをリセットするには、Contrast UI を使用 (521ページ)するか、Windows (936ページ)または Linux (935ページ)のコマンドラインを使用してください。

ユーザのなりすまし

ユーザになりすます機能を利用すると、なりすましたユーザと同じロールと権限を持っているかのように、組織にアクセスできます。なりすましは、問題のトラブルシューティングを行う場合に便利です。

SuperAdmin ロールがある場合、組織のページより該当の組織の行で**なりすます**を選択すると、その組織の最初の組織管理者に対してなりすますことができます。また、ユーザのページにて**なりすます**を選択して、任意のユーザになりすますことができます。

ServerAdmin または SystemAdmin ロールがある場合、組織のページより該当の組織の行で**なりすます**を選択すると、その組織の最初の組織管理者に対してなりすますことができます。なりすましを利用するには、対象の組織へのアクセス権が必要です。

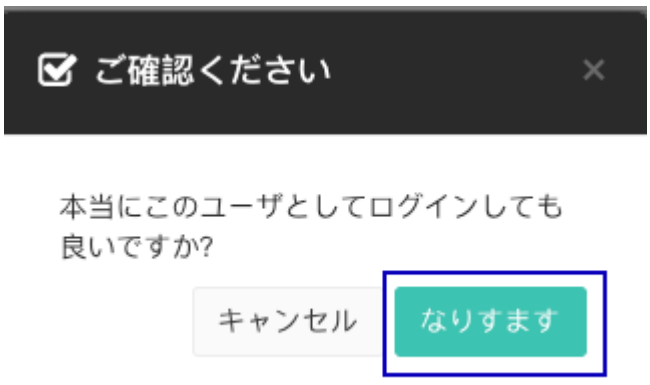
Contrast では、24 時間後に自動的になりすましが無効になります。

開始する前に

- なりすます対象のユーザに組織のロール (955ページ)が必要です。
- なりすましのオプションが組織に表示されていない場合は、SuperAdmin ユーザがその組織のなりすましを有効にできる (908ページ)の設定をオンにする必要があります。
- 組織管理者は、アクセスしたい組織のなりすましを有効にする (872ページ)必要があります。

手順

- **ユーザレベルでなりすましを開始するには：**
  1. Contrast Web インターフェイスに SuperAdmin としてログインします。
  2. **ユーザ**を選択します。
  3. なりすましをしたいユーザの行の最後にある三角形( )を選択し、**なりすます**を選択します。
  4. 「ご確認ください」の画面で、**なりすます**をクリックして、選択したユーザへのなりすましを確定します。



  なりすましたユーザとして、Contrast のセッションが開始します。
- **組織レベルでなりすましを開始するには：**
  1. SuperAdmin、ServerAdmin、または SystemAdmin として Contrast Web インターフェイスにログインします。

2. **組織**を選択します(まだ選択していない場合)。
3. アクセスしたい組織の行の最後にある三角形( )をクリックして、**なりすます**を選択します。
4. 「ご確認ください」の画面で、**はい**を選択して、表示されたユーザになりすますことを確定するか、アクセスしたい組織の管理者である別のユーザを選択します。

認証の設定

注記
SaaS 版をご利用の場合は、Contrast Security が認証を設定します。ただし、SSO の設定を含め、この設定を上書きする権限を組織管理者に付与できる場合があります。

この変更を希望する場合は、Contrast サポートにご連絡ください。

デフォルトでは、ユーザのログイン名、認証情報、およびアプリケーションの認証に関するその他の詳細情を含むユーザ情報がユーザディレクトリに格納されます。ユーザ名とパスワードは Contrast データベースの内部ディレクトリに(一方向ハッシュを使用して)格納されます。Contrast ユーザにパスワードポリシー (934ページ)および 2 段階認証 (921ページ)を設定できます。

また、認証に外部ディレクトリを使用することもできます。この場合、ユーザ名のみが Contrast のデータベースに格納されます。Contrast では、以下をサポートします。

- LDAP (926ページ)
- Active Directory (922ページ)
- シングルサインオン (931ページ)
- 信頼される HTTPS プロキシ (934ページ)

認証の設定を変更した場合は、Contrast を再起動 (890ページ)する必要があります。認証の設定を変更するには、システムの設定 (936ページ)で**認証**を選択します。

重要
認証モードを切り替える時は、以下の点に注意してください。

- 変更前の認証モードで作成されたユーザは、ユーザの E メールアドレスが新旧の認証プロバイダ間で同じでない限り、機能しなくなります。
- 新しい認証モードを設定してサーバを再起動すると、ユーザは自分のアカウントが新しい組織または既存の組織に追加されるまで、Contrast にログインできなくなります。オンプレミス版をご利用の場合は、スーパー管理者(SuperAdmin)が組織管理者(Admin)のアカウントを管理し、その後各組織管理者がその組織内のユーザを管理します。

注記
外部の認証プロバイダ(LDAP または AD)を使用するモードの場合、ユーザを追加する際にユーザ名フィールドはライブ検索として機能し、適切なグループ内のユーザが表示されます。

ヒント
ロールと権限はユーザディレクトリではなくアクセスグループによって管理されるため、認証を設定する前にアクセスグループを作成することをお勧めします。管理者用とユーザ用に、少なくとも 2 つの一意のアクセスグループが必要になります。

システムレベルでの 2 段階認証の有効化

注記
Contrast で SSO を設定し、二要素認証も使用する場合は、Contrast ではなく、ID プロバイダ(idP)を使用して二要素認証を設定してください。Contrast で SSO を設定すると、ユーザを認証する責任は idP に渡されます。

2 段階認証を有効/無効にするには：

1. システムの設定 (936ページ)の左ナビゲーションで**セキュリティ**を選択します。
2. トグルボタンをオン(緑色)にして 2 段階認証を有効にします。
3. **組織の上書きを許可**の横にあるチェックボックスをオンにして、組織管理者がユーザにとってこの機能を必須にするかどうかを選択 (827ページ)できるようにします。

注記
ユーザが複数の組織に属している場合は、デフォルトの組織によって 2 段階認証の設定が決まります。

また、ユーザは 2 段階認証の通知を受け取る方法を選択 (521ページ)することができます。

Microsoft Active Directory の設定

システムの管理者 (957ページ)は、Microsoft Active Directory(AD)に接続するように Contrast を設定できます。このインテグレーションを設定するには、AD Connector を使用します。AD はディレクトリの構造が明確に定義されているため、設定する選択肢も少なくなり、より直接的な設定ができます。

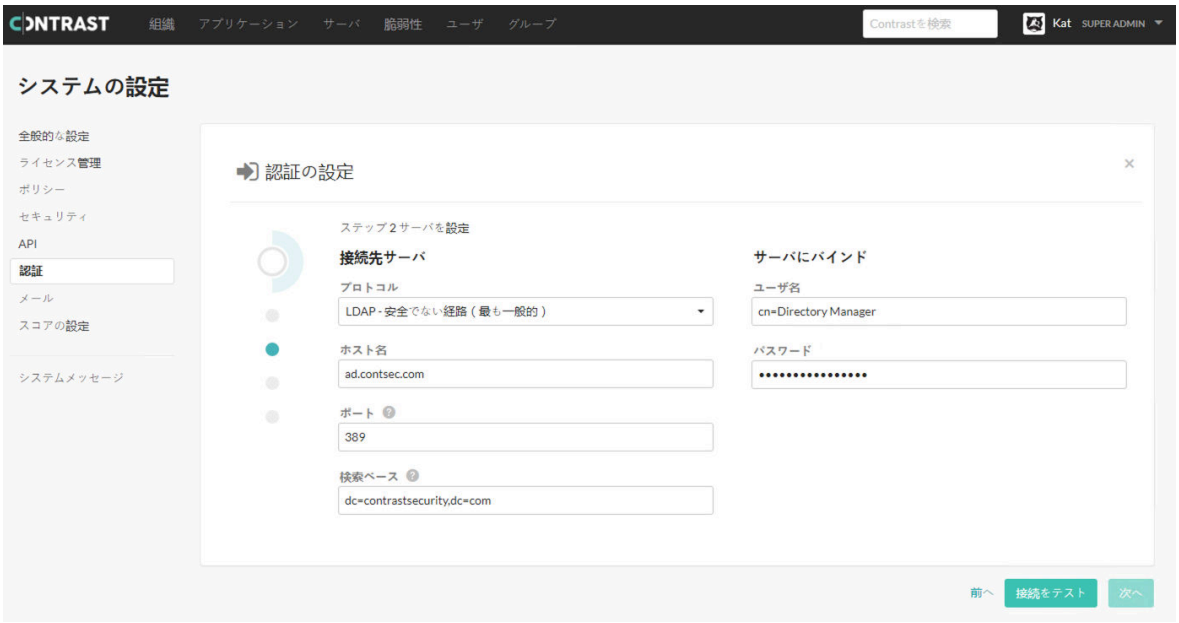
注記
ローカルデータベースなど、別の認証方法から AD に切り替えると、ユーザ ID 属性に一貫性がない場合は問題が発生する可能性があります。

AD がオフラインの場合のアクセス

AD サービスで接続や設定の問題が発生した場合、デフォルトの SuperAdmin アカウントを使用して Contrast にログインしてください。これにより、AD がオフラインの場合でも、引き続き Contrast アカウントに直接アクセスできます。

手順

1. はじめに、Active Directory サーバに読取り専用ユーザを作成します。ユーザには、検索ベース(Search Base)のみの権限を持つユーザを含め、ディレクトリに対する読み取り権限を持つ必要があります。このユーザは、AD のユーザを設定 (924ページ)する際やユーザにバインドする際に、検索ベースを設定するために必要となります。
2. 外部の AD サーバで、ユーザグループを作成します。このグループは、後で Contrast の SuperAdmin 権限を割り当てる (924ページ)ために使用します。
3. システムの設定 (936ページ)で**認証**を選択します。
4. **認証方法を変更**を選択したら手順に従って、サーバ、グループ、詳細を設定します。
5. **Active Directory** を選択します。
6. 以下の値を入力します。LDAPS(LDAP over SSL)では、一部の設定が異なる場合があります。



接続先サーバで、以下の項目に入力します。

- **プロトコル**：LDAP サーバとの通信に使用するプロトコルです。ドロップダウンから、**LDAP** か **LDAPS** を選択します。デフォルトは、**LDAP** です。AD で自己署名証明書またはプライベート証明書を使用 (925ページ)している場合は、**LDAPS** オプションで追加の設定が必要になる場合があります。
- **ホスト名**：LDAP サーバと通信する際に接続するホスト名です。AD サーバの DNS ホスト名または IP アドレスを入力します。マルチテナントフォレストでは、これはグローバルカタログサーバになります。デフォルトは、`localhost` です。
- **ポート**：LDAP サーバと通信する際に接続するポートです。標準(シングルテナント、シングルドメイン)ディレクトリの場合、**389 (LDAP)**番か **636 (LDAPS)**番のポートを指定してください。マルチテナントまたはマルチドメインフォレストでは、**3268 (LDAP)**番か **3269 (LDAPS)**番を指定してください。
- **検索ベース**： LDAP サーバとの通信に使用するベース DN(AD 環境でグローバルベースレベルのコンテナを表す識別名)です。通常、これはドメイン名またはサブドメイン名です。デフォルトは、`dc=contrastsecurity,dc=com` です。ログインドメインが *yourdomain.com* であれば、ベース DN は `dc=yourdomain,dc=com` となります。

サーバにバインドで、以下の項目に入力します。

- **ユーザ名**：検索機能を実行するために、ディレクトリにバインドするユーザの完全な DN 名を入力します。デフォルトは、`cn=Directory Manager` です。
- **パスワード**：アプリケーションが LDAP サーバに接続する際に使用するユーザアカウントのパスワードです。

7. **接続をテスト**を選択して、サーバへの接続を確認します。接続を確認したら、**次へ**を選択します。
8. グループを設定します。 (924ページ)
9. 詳細設定を指定します。 (924ページ)
10. 全ての設定オプションを指定したら、**ログインをテスト**ボタンを使用して、スーパー管理者(SuperAdmin)と組織管理者(Admin)の両方でログインできることを確認します。

> **注記**
> テストに時間がかかり過ぎると思われる場合は、**詳細設定** (924ページ)の**Referral(照会)機能をフォロー**オプションの設定が間違っている可能性があります。設定を切り替えると、ログイン機能の検証が早くなるはずです。

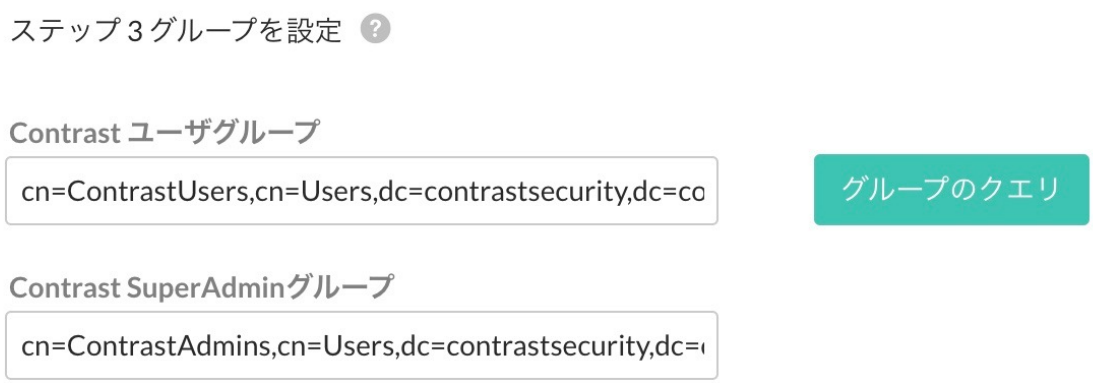
11. **完了**を選択して、設定を完了します。

Active Directory のグループの設定

Active Directory の設定 (922ページ)一環として、グループを設定する必要があります。

Contrast では、連携している AD サーバを使用してのデータアクセス制御は行われません。つまり、ロールやアプリケーション内のデータへのアクセスはアプリケーションによって行われ、組織管理者(Admin)がユーザロールを設定します。ただし、ログイン時や新規ユーザ作成時に、指定されたユーザが Active Directory(AD)内の正しいグループに属しているかどうか、アクセス制御のチェックが行われます。

手順



1. 外部 AD サーバで作成したグループを使用して、ユーザを以下のいずれかの Contrast グループに割り当てます。
   - **SuperAdmin グループ**：このグループを使用すると、ユーザはスーパー管理者用のインターフェイスにログインできます。
     このグループのユーザは、Contrast に初めてログインした時に認証され、権限が付与されます。
   - **ユーザグループ**：このグループを使用すると、ユーザは組織に追加され、Contrast Web インターフェイスにログインできます。このグループは、他の全てのユーザに適しています。
     ユーザがログインできるようにするために、Contrast Web インターフェイスで組織に手動でユーザを追加 (822ページ)してください。

   > **注記**
   > AD インスタンスで両方のグループにユーザを追加した場合、 そのユーザは Contrast で設定時に自動的に SuperAdmin グループに追加されます。

2. **グループのクエリ**を選択すると、入力フィールドに入力する際に既存のグループをライブ検索できます。

   > **注記**
   > アクセス制御のチェックを通さずに Contrast で AD 認証を持つユーザを作成するには、データベースで以下のクエリを実行します。
   >
   > ```
   > UPDATE teamserver_preferences SET property_value='true' \
   > WHERE \
   > property_name='directory.skip.user_existence.validation'
   > ```

Active Directory の設定

Active Directory の設定 (922ページ)では、**詳細設定**で以下の項目を入力してください：

- **ユーザ・ベース DN**：デフォルトは *cn=Users* で、AD のデフォルトコンテナです。ただし、AD が異なる方法で設定されている場合は、これはユーザがディレクトリに格納されている最上位のコンテナへのパスになります。
  例えば、ユーザが DN `CN=Engineering,CN=GlobalUsers,DC=intranet,dc=example,dc=com` に格納されており、ベース DN が `DC=intranet,DC=example,DC=com` である場合、ユーザ DN のサフィックスの値は、`CN=Engineering,DC=GlobalUsers` になります。
- **ユーザ ID の属性**：ユーザが Contrast アプリケーションにログインする際に、ユーザ名として入力する属性を入力します。ユーザが最も使い慣れた属性を指定します。デフォルトは、**E メールアドレス**です。
  - **ログイン ID**：AD の `sAMAccountName` 属性です。これは通常、Windows にログインする際に使用するユーザ名です。
  - **E メールアドレス**：AD の `mail` 属性で、ユーザの E メールアドレスが含まれます。
  - **ユーザプリンシパル**：AD の `userPrincipal` 属性、完全なユーザ名が含まれます。
- **ネストしたグループ内を検索**：ネストしたグループ内の検索を有効または無効にします。このトグルはデフォルトで無効になっています。
- **Referral(照会)機能をフォロー**：マルチテナントまたはマルチドメインのエンタープライズフォレストでは、より詳細な情報を得るために LDAP クエリをほかのサーバに照会する場合があります。このトグルはデフォルトで無効になっています。
- **Referral(照会)の範囲**：AD が `Referral()` の応答を返したときにフォローする Referral(照会)の回数を制限します。デフォルトは「5 回」です。

Active Directory での自己署名/プライベート証明書の使用

SSL を使用してサーバに接続するよう AD インテグレーションを設定 (922ページ)している場合は、サーバの証明書を Contrast JRE が使用する新しい信頼ストアにインポートしなければならない場合があります。

1. 管理者から、サーバの証明書を PKCS#12 フォーマットで取得します。自己署名証明書を使用している場合、これは実際の AD サーバの証明書になります。プライベート認証局がある場合は、そのサーバの CA 証明書が必要になります。
2. このサーバの証明書を入手したら、この証明書を含む信頼ストアを作成します。Contrast がインストールされているディレクトリのコマンドシェルから、管理者として以下のコマンドを実行します。

```
$ mkdir data/conf/ssl
$ jre/bin/keytool -import -file <path to certificate> -alias <hostname> \
  -keystore data/conf/ssl/truststore.jks
```

3. サーバまたは CA の証明書を含む信頼ストアを作成したら、*bin/contrast-server.vmoptions* ファイルに以下のコマンド行を追加します。

```
-Djavax.net.ssl.trustStore=<full path to truststore>
-Djavax.net.ssl.trustStorePassword=<password you set for the \
trustStore, if any>
```

4. Contrast Server サービスを再起動して、AD に対するクエリで SSL が使用されていることを確認します。

LDAP の設定

Contrast は、さまざまな種類の LDAP (Lightweight Directory Access Protocol)サーバと連携します。LDAP は、Web アプリケーションが LDAP ディレクトリサーバに登録されているユーザーやグループを検索するために使用できるインターネットプロトコルです。

Contrast は、以下の種類の LDAP サーバをサポートしています。

- OpenLDAP
- OpenDS
- ApacheDS
- Fedora Directory Server
- Microsoft Active Directory
- 汎用 LDAP サーバ

社内ディレクトリでユーザやグループを管理していて、アプリケーションのユーザアクセスの管理を社内ディレクトリ管理者に委任する場合、LDAP ディレクトリサーバに接続すると便利です。

注記

ローカルデータベースなど、別の認証方法から LDAP に切り替えると、ユーザ ID 属性に一貫性がない場合は問題が発生する可能性があります。

システム管理者は、以下の手順で LDAP サーバを設定できます。

重要

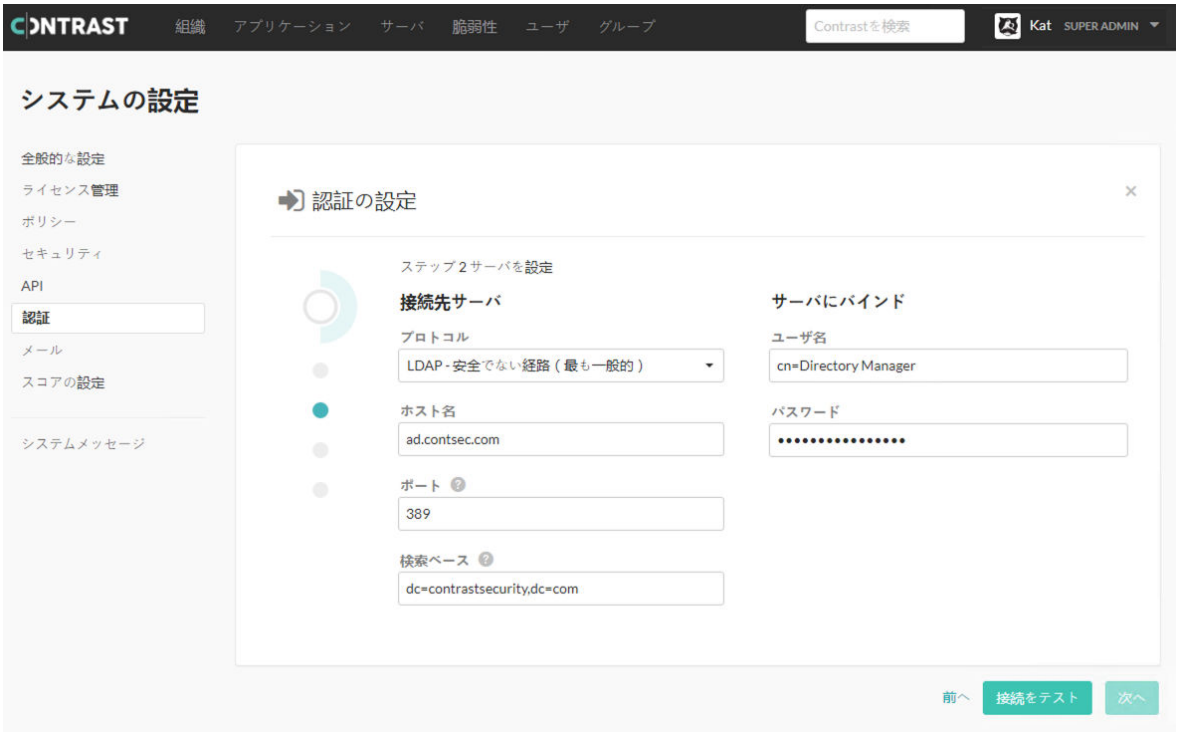
SSL を使用してサーバに接続するように LDAP 連携を設定する場合は、自己署名証明書またはプライベート証明書 (931ページ)の追加手順が必要になることがあります。

LDAP がオフラインの場合のアクセス

LDAP サービスで接続や設定の問題が発生した場合、デフォルトの SuperAdmin アカウントを使用して Contrast にログインしてください。これにより、LDAP がオフラインの場合でも、引き続き Contrast アカウントに直接アクセスできます。

手順

1. はじめに、LDAP サーバに専用のユーザを作成します。Contrast が LDAP ディレクトリと連携する設定方法によりますが、読み取り専用ユーザまたは読み書き可能なユーザを作成してください。ユーザは、検索ベース(Search Base)のみの権限を持つユーザを含め、ディレクトリに対する読み取り権限を持つ必要があります。このユーザは、LDAP にユーザを設定する (929ページ)際やユーザにバインドする際に、検索ベースを設定するために必要になります。
2. 外部の LDAP サーバで、ユーザグループを作成します。これらのグループは、後で Contrast の SuperAdmin 権限を割り当てる (928ページ)ために使用します。
3. システムの設定 (936ページ)で、**認証**を選択します。
4. **認証方法を変更**を選択します。
5. **LDAP** を選択します。
6. **接続先サーバ**と**サーバにバインド**に必要な情報を入力します。



オプション	説明	デフォルト
接続先サーバ		
プロトコル	LDAP サーバとの通信に使用するプロトコルです。**LDAP** または **LDAPS(LDAP with SSL)**を選択します。	**LDAP**
ホスト名	LDAP サーバと通信する際に使用するホスト名です。	`localhost`
ポート	LDAP サーバと通信する際に使用するポートです。	**389 (LDAP)、636 (LDAPS)**
検索ベース	LDAP サーバと通信する際に使用するベース識別名(DN)です。ログインドメインが、*yourdomain.com* であれば、ベース DN は `dc=yourdomain,dc=com` となります。	`dc=contrastsecurity,dc=com`
サーバにバインド		
方式	LDAP サーバと通信する際に使用する方法を選択します。オプションについては、次の表に示します。	Simple(シンプル)
ユーザ名	クエリの実行や認証確認のためにディレクトリにバインドするユーザの完全な DN です。	N/A
パスワード	**ユーザ名**フィールドで指定したバインドユーザのパスワードです。	N/A

Contrast で使用できるバインドメカニズムは 4 つあります。それぞれ必須項目が異なります。

方式	説明	必須項目	オプション項目
匿名(Anonymous)	管理者は、ディレクトリへの匿名の読み取り専用アクセスを提供します。	なし	N/A
シンプル (Simple)	ユーザ名とパスワードによる標準的な認証です。ユーザ名とパスワードは、LDAP サーバから提供される情報で認証されます。	ユーザ名、パスワード	N/A
DIGEST-MD5	認証されるサーバに送信する前に、MD5 を使用してユーザ名とパスワードが提供· ハッシュされます。	ユーザ名、パスワード	SASL レルム
CRAM-MD5	LDAP サーバが事前認証チャレンジを発行し、認証されるべきユーザ名とパスワードを MD5 でハッシュ化して送信します。	ユーザ名、パスワード	SASL レルム

7. LDAP サーバへの接続を設定したら、**接続をテスト**を選択します。接続テストでは、アプリケーションが LDAP サーバに接続してクエリを実行できるかどうかを確認します。
8. LDAP のグループを設定します (928ページ)。
9. LDAP のユーザを設定します。 (929ページ)
10. グループとユーザのマッピングが適切に設定されていることを確認するには、**ログインをテスト**を選択します。
11. スーパー管理者(SuperAdmin)と組織管理者(Admin)の両方が正常にログインできることを確認したら、**完了**を選択して設定を完了します。

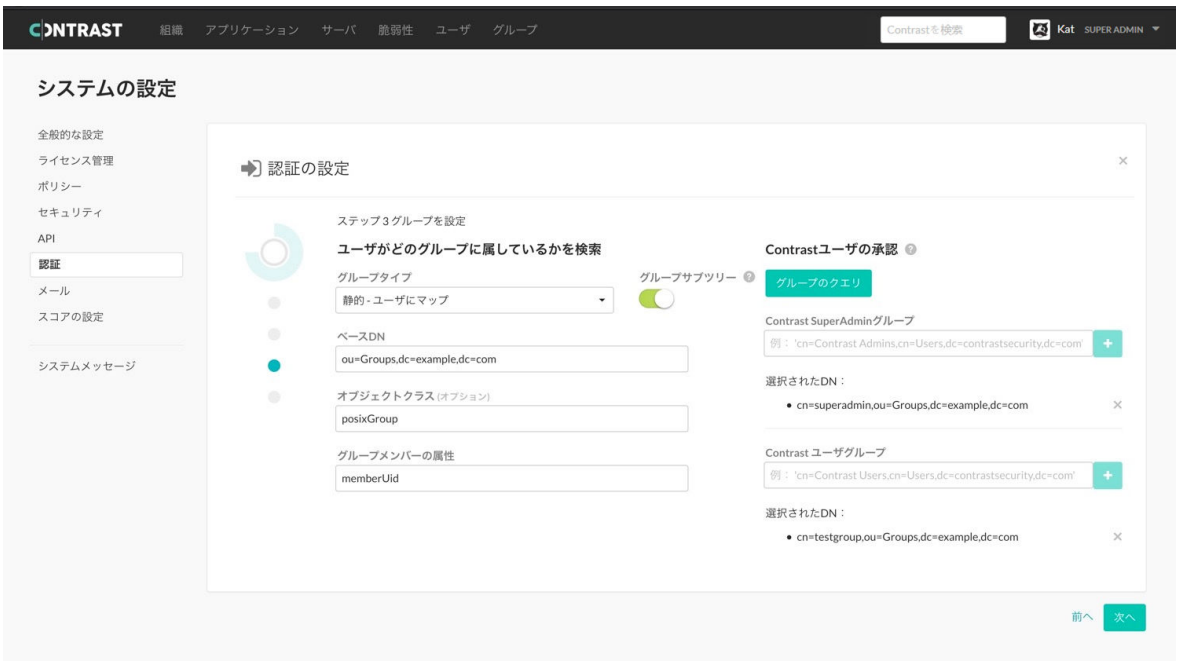
LDAP のグループの設定

LDAP の設定 (926ページ)の一環として、グループを設定する必要があります。

組織管理者がユーザのロールと権限 (953ページ)を設定したら、それぞれのアプリケーションがデータに対するロールとアクセスをそのアプリケーション内で処理します。ユーザを設定するときは、ログイン時にユーザをアクセスグループに追加 (929ページ)するよう選択することができます。ただし、この機能が有効になっていない場合でも、Contrast は LDAP ディレクトリを使用して、そのユーザが適切なグループに割り当てられるようにします。

グループを設定するには：

1. 以下の値を入力します。



オプション	説明	デフォルト
グループタイプ	グループタイプは、サーバの機能と設定によって異なります。グループは以下のいずれかになります。 • **静的**：`uniqueMember` などのオブジェクトの属性を通じて、グループがメンバーを追跡します。この表の残り 4 つのオプションは、静的グループにのみ適用されます。 • **動的**：ユーザオブジェクトが自身のメンバーシップを追跡します。ユーザがグループのメンバーになると、ユーザオブジェクトに対してグループが動的に追加されます。	**静的**
グループサブツリー	ディレクトリ内でグループを検索するときに、ベース DN のサブツリーを含めるかどうかを設定します。	**有効**
ベース DN	これは、(ユーザベース DN と同様に)アプリケーションが LDAP サーバ内のグループを見つけるための識別名(DN)です。	`ou=Groups`
オブジェクトクラス	空欄のままにすると、アプリケーションでは「`group`」、「`groupOfUsers`」、「`groupOfUniqueUsers`」のデフォルト値が使用されます。これは LDAP 環境において標準的であるため、必須のフィールドではありません。	N/A
グループメンバーの属性	ディレクトリのグループオブジェクト内の属性であり、そのグループのメンバーを含めます。これは LDAP 環境によって異なる場合があるため、LDAP 管理者に確認して適切な属性を使用していることを確認してください。 グループの各メンバーは、相対識別名(RDN)ではなく、完全識別名(DN)として、列挙する必要があります(例："cn=smith,ou=Users,cn=support,dc=test,dc=org")。 RDN を使用すると、Contrast では LDAP グループでそのユーザが認識されません。	`uniqueMember`

2. 外部 LDAP サーバで事前に作成したグループを使用して、ユーザを以下のいずれかのグループに割り当てます。
   - **SuperAdmin グループ**：このグループでは、ユーザは SuperAdmin 権限でログインできます。
   - **ユーザグループ**：このグループを使用すると、ユーザは組織に追加され、標準インターフェイスにログインできます。このグループは、他の全てのユーザに適しています。

> **重要**
> ユーザが両方のグループに属していて、プロビジョニングが無効な場合、このユーザは SuperAdmin として作成されます。プロビジョニングが有効な場合は、ユーザは SuperAdmin 権限なしで作成されます。

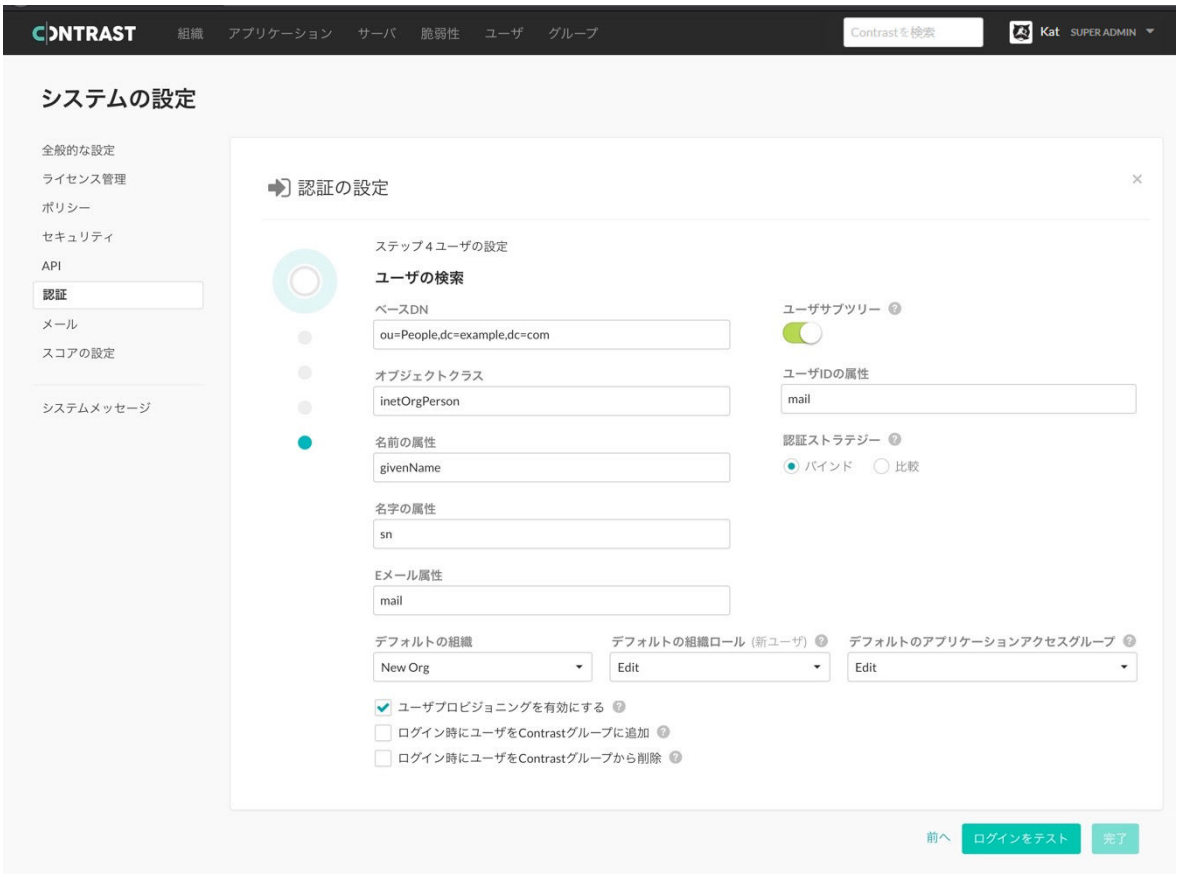
3. **グループのクエリ**を選択すると、入力フィールドに入力する際に既存のグループをライブ検索できます。

LDAP のユーザの設定

LDAP の設定 (926ページ)の一環として、ユーザを設定する必要があります。

LDAP ディレクトリと完全に統合するため、Contrast では LDAP サーバへの接続方法、およびディレクトリ内のユーザとグループの検索方法に関する情報が必要になります。

1. Contrast でディレクトリ内のユーザを検索する方法について、以下の情報を入力します。ほとんどの LDAP 環境ではデフォルト設定が適切です。



オプション	説明	デフォルト
ベース DN	Contrast がユーザの検索を開始する、(グローバルベース DN の下にある)コンテナを示します。多くの組織では、これは単一のコンテナ(例：*ou=Users*)ですが、適切なコンテナを検索していることを LDAP 管理者が確認する必要があります。	`ou=users`
オブジェクトクラス	ディレクトリ内のユーザオブジェクトの LDAP オブジェクトクラスを示します。	`inetOrgPerson`
名前の属性	ユーザの名前を含む LDAP フィールドを示します。	`givenName`
名字の属性	ユーザの名字を含む LDAP フィールドを示します。	`sn`
E メール属性	ユーザの E メールアドレスを含む LDAP フィールドを示します。	`mail`
ユーザサブツリー	有効にすると、ユーザを検索するときにベース DN のサブツリーが含められます。	**有効**
ユーザ ID の属性	ユーザ ID として識別される LDAP フィールドを示します。これは、Contrast に口グインするときに入力するユーザ名です。	`uid`
認証ストラテジー	ユーザが認証情報を提供したときの、Contrast によるユーザの認証方法を選択します。**バインド**とは、認証のために、ユーザの認証情報がアプリケーションからサーバに送信されることを意味します。**比較**とは、サーバによりユーザの認証情報がハッシュ化され、パスワード属性の値と比較されることを意味します。	**バインド**
パスワード属性	ユーザのパスワードを含む LDAP フィールドです。 認証ストラテジーで**比較**を選択した場合、この属性にはユーザのハッシュ化されたパスワードが含まれます。	`userPassword`

2. 誰かが口グインするために LDAP リクエストを行った場合に自動的に新規ユーザアカウントを作成するには、**ユーザプロビジョニングを有効にする**の横にあるチェックボックスをオンにします。ドロップダウンメニューを使用して、新規ユーザの**デフォルトの組織**、**デフォルトの組織ロール**、**デフォルトのアプリケーションアクセスグループ**を選択します。
3. Contrast では、ログイン時にユーザの LDAP グループに応じてユーザを自動的にプロビジョニングしたり、プロビジョニングを解除したりできます。この機能が LDAP ベースの認証について有効になっている場合、ユーザは対応する LDAP グループにマッピングされた Contrast アクセスグループに追加され、禁止された Contrast グループから削除されます。ユーザは複数のグループに追加でき、また複数の組織へのアクセス権が付与されるグループにも追加できます。

重要
これを機能させるには、Contrast グループが既に存在している必要があり、(プロビジョニング目的の)LDAP のグループの名前が Contrast グループの名前と一致する必要があります。

4. Contrast にログインするときにユーザをグループに追加するには、**ログイン時にユーザを Contrast グループに追加**の横にあるチェックボックスをオンにします。Contrast にログインするときにユーザをグループから削除するには、**ログイン時にユーザを Contrast グループから削除**の横にあるチェックボックスをオンにします。

LDAP での自己署名/プライベート証明書の使用

SSL を使用してサーバに接続するよう LDAP インテグレーションを設定 (926ページ)している場合は、サーバの証明書を Contrast JRE が使用する新しい信頼ストアにインポートしなければならない場合があります。

1. はじめに、管理者から、サーバの証明書を PKCS#12 フォーマットで取得します。自己署名証明書を使用している場合、これは実際の LDAP サーバの証明書になります。プライベート証明書(CA)がある場合は、そのサーバの CA 証明書が必要になります。
2. サーバの証明書を入手したら、これを Contrast を実行する JRE が使用する信頼ストアにインポートします。Contrast がインストールされているディレクトリのコマンドシェルから、管理者として以下のコマンドを実行します。

```
$ jre/bin/keytool -import -file <path to certificate> -alias <hostname> \
\-keystore <ts install>/jre/lib/security/cacerts
```

3. Contrast Server サービスを再起動して、LDAP に対するクエリで SSL が使用されるようになったことを確認します。

システムレベルでのシングルサインオン(SSO)の設定

シングルサインオン(SSO)は、ひとつの資格情報を使用して複数のアプリケーションにアクセスできる認証サービスです。システム管理者は、このサービスを SAML 2.0 でサポートされるプロバイダで使用するように Contrast を設定できます。

注記
詳細については、SAML 2.0 仕様を参照してください。

Contrast で SSO を設定し、二要素認証も使用する場合は、Contrast ではなく、ID プロバイダ(idP)を使用して二要素認証を設定してください。Contrast で SSO を設定すると、ユーザを認証する責任は idP に渡されます。

認証は、ID プロバイダ(IDP)を介して行われます。独自の汎用 IDP を使用することもできますし、Okta、 OneLogin、 Ping Identity、 ADFS などの一般的なサードパーティのプロバイダを使用することもできます。

IDP のメタデータ情報を準備したら、XML ファイルまたはメタデータの URL で Contrast に接続するためのメタデータを指定します。

オンプレミス版のお客様の場合、スーパー管理者(SuperAdmin)がシステムレベルで SSO を設定します。SaaS 版をご利用のお客様は、組織レベルで SSO を設定できます。マルチテナントホストのインスタンスでは、1 つの Contrast インスタンスに複数の IDP を設定できます。

注記
ユーザがメールアドレスではなくユーザ ID で識別されている場合、それらのアカウントは自動的に SSO 設定に移行されず、再作成する必要があります。

開始する前に

SSO を使用する場合は、ユーザの電子メールを渡すよう `NameID` を設定する必要があります。

オプションとして、ユーザの名前と名字を設定する場合には、次のように SAML アサーションを使用して追加の属性を渡すように IdP を設定する必要があります。

- **名前**：http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname
- **名字**：http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname

これらのフィールドが存在しないか空白の場合、デフォルトでは `NameID` フィールドが使用されます。また、ユーザプロビジョニングが有効になっている場合は、ユーザの名字と名前が自動的に入力されます。

```
1<saml2:Attribute Name="http://schemas.xmlsoap.org/ws/2005/05/identity/
claims/givenname"
2                                \
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified"
3                                >
4                <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema"
5                                      xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
6                                      xsi:type="xs:string"
7                                      >Dan</saml2:AttributeValue>
8            </saml2:Attribute>
```

手順

1. Contrast では、SAML 認証用のキーは提供されません。秘密キーを指定せずに SSO を有効にした場合、IDP でによるログインのみを実行できます。Java キーツールを使用して、独自の自己署名キーを生成する必要があります。

   ```
   keytool -genkeypair -alias some-alias -keypass changeit -keyalg RSA -
   keystore samlKeystore.jks
   ```
2. 暗号化プロパティエディタ (943ページ)を使用して *saml.properties* を変更し、前の手順で作成したキーストアに対する値を更新します。

   ```
    authenticator.saml.keystore.path                    : /path/to/
   samlKeystore.jks
    authenticator.saml.keystore.default.key             : some-alias
    authenticator.saml.keystore.passwordMap             : some-alias=changeit
    authenticator.saml.keystore.password                : changeit
   ```
3. 変更を行ったら、Contrast を再起動して、新しいキーストアを認識させます。
4. Contrast のシステムの設定 (920ページ)で、**認証**を選択し、次に**認証方法を変更**を選択します。
5. **シングルサインオン**を選択します。
6. 指定した情報を使用して、お使いの IDP と Contrast を設定します(IDP の設定では、**エンティティ ID** と**メタデータ URL** の入力も必要です)。

7. **ID プロバイダ**の名前を入力します。
8. IDP メタデータを入力します。**メタデータ URL にアクセスできる**場合、チェックボックスをオンにし、URL を入力します。
9. Contrast にログインするための SAML リクエストがあった時に、新しいユーザアカウントを自動的に作成する場合は、**ユーザプロビジョニングを有効にする**のチェックボックスをオンにします。
   - ドロップダウンを使用して、新規ユーザの**デフォルトの組織ロール**と**デフォルトのアプリケーションアクセスグループ**を選択します。
   - ユーザプロビジョニングのトリガーに使用する必要のある**承認されているドメイン**を追加します(例：*yourdomain.com*)。

> **注記**
> ユーザをグループに自動追加 (916ページ)することもできます。

10. **保存**を選択します。エラーが発生した場合は、デバッグログを確認 (899ページ)してトラブルシューティングを行ってください。
11. Contrast を再起動 (890ページ)して、変更を適用します。

接続が完了したら、**SSO** タブに戻って設定を確認・編集します(変更を適用するには、再テストして Contrast を再起動 (890ページ)してください)。組織の設定をデフォルトに戻すには、**Contrast 管理認証に戻す**を選択して、変更を確定します。

インストール時に SuperAdmin を無効にした場合は、パブリックノード用とシークレットノード用の 2 つのメタデータが提供されます。両方の設定を Contrast インターフェイスで行う必要があります。

関連項目

SAML インテグレーションのトラブル解決方法

Okta を使用してユーザとグループのプロビジョニングを設定する

グループにユーザを自動的に追加するように ADFS を構成する

HTTPS プロキシ認証の有効化

認証に、信頼できるプロキシを使用することができます。このプロキシを使用して、ユーザを認証し、ユーザ名を HTTP ヘッダで Contrast に送信できます(このタイプの認証は、特に x 509 クライアントで便利です)。

認証の設定を開始する前に、ユーザを Contrast で設定しておく必要があります。また、認証後に Contrast にアクセスできるよう、両方の設定で同じメールアドレスをユーザ名として使用する必要があります。

信頼できる HTTPS プロキシ認証を有効にするには：

1. *~/path_to_contrast_installation/data/conf/auth.properties* で認証モードを *http_header* に変更し、*auth.properties* プロパティファイルを更新します。
2. デフォルトの HTTP ヘッダ名は、*Contrast-Authentication* です。これは、*auth.properties* ファイルで *auth.http.header.field.name* の値を更新することによっても設定できます。
3. Contrast を再起動した後は、それぞれのリクエストに HTTP ヘッダの *Contrast-Authentication: username* を含めてログインする必要があります。

注記
信頼できる HTTPS プロキシ認証を使用できるのは、全ての Contrast ノードが信頼できるプロキシを通じてのみアクセスできる場合に限られます。ノードには直接アクセスできないようにしてください。さもなければ攻撃者が任意の認証されたユーザになりすます恐れがあります。

システムレベルでのパスワードの設定

パスワードポリシーを作成して、システムレベルでパスワードを規制します。

1. システムの設定 (936ページ)で**セキュリティ**を選択します。
2. **デフォルトパスワードポリシー**で、**組織の上書きを許可**する場合は、このチェックボックスをオンにします。これにより、組織管理者は組織のパスワードポリシーを設定 (826ページ)できるようになります。
3. ポリシーについて、以下の設定を入力します。
   - パスワード強度を設定します。**弱**、**中**、**強**、**複雑**または**カスタム**を選択できます。**カスタム**を選択した場合は、**最小限必要な大文字**、**小文字**、**数字**、**記号**の文字数を入力します。
   - 必要な文字数を**最小桁数**フィールドに入力します。
   - ドロップダウンメニューを使用して、**パスワード期限**が切れるまでの時間を選択します。
   - **ログインのロックアウト**までのログイン試行回数を入力します。
   - **非アクティブアカウントの有効期限**が切れるまでの時間を入力します。
   - **パスワードの再利用を制限**のチェックボックスをオンにし、ドロップダウンメニューを使用して、各パスワードを再利用できる回数を選択します。

- **パスワードのリセットを制限**のチェックボックスをオンにし、ドロップダウンメニューを使用して、リセットのリクエストの送信後にユーザがパスワードをリセットできる日数を選択します。
- ドロップダウンメニューを使用して、**アイドルタイムアウト**および**セッションタイムアウト**までの経過時間を選択します。

4. **保存**を選択します。

Linux で SuperAdmin パスワードをリセット

SuperAdmin のパスワードを変更するには、`contrast-server.vmoptions` ファイルを編集するか、環境変数を使用します。

ほとんどの場合、`contrast-server.vmoptions` ファイルを編集するのが最も簡単な方法です。Contrast アプリケーションをコンテナで使用している場合は、パスワードのリセットには環境変数を使用してください。

手順

1. コマンドプロンプトを開き、インストール時に作成した Contrast サービスのアカウントでログインします。例：
   ```
   sudo -su contrast_service
   ```
2. 次のコマンドを実行して、Contrast サーバをシャットダウンします。
   ```
   $CONTRAST_INSTALLATION/bin/contrast-server stop
   ```
3. パスワードをリセットするために `contrast-server.vmoptions` ファイルを使用しますか？
   - **使用する場合**は、手順 4 に進みます。
   - **使用しない場合**は、手順 5 に進みます。
4. `contrast-server.vmoptions` ファイルを使用してパスワードをリセットするには、以下の手順を実行します。
   a. `$CONTRAST_INSTALLATION/bin` ディレクトリに移動します。ほとんどのシステムで、このディレクトリは`/opt/Contrast/bin` です。
   b. テキストエディタで、`contrast-server.vmoptions` ファイルを開きます。
   c. ファイルに以下のオプションを追加します(`youremaildomain.com` を自分の E メールドメインに置き換えてください)。
      ```
      -Dreset.superadmin=true
      -Dsuperadmin.username=contrast_superadmin@<your.email.domain.com>
      -Dsuperadmin.password=<password>
      ```
5. `vmoptions` ファイルの代わりに環境変数を使用してパスワードをリセットするには、以下の手順を実行します。
   a. `$CONTRAST_INSTALLATION` ディレクトリに移動します。ほとんどのシステムで、このディレクトリは`/opt/Contrast` です。
   b. 次のコマンドを入力します。
      ```
      export INSTALL4J_ADD_VM_PARAMS="$INSTALL4J_ADD_VM_PARAMS -
      Dreset.superadmin=true -
      Dsuperadmin.username=contrast_superadmin@<your.email.domain.com> -
      Dsuperadmin.password=<new password>"
      ```
6. 次のコマンドを実行して、Contrast サーバを起動します。
   ```
   $CONTRAST_INSTALLATION/bin/contrast-server start
   ```
   サーバが起動したら、手順 4 または手順 5 で指定したパスワードを使用します。
7. 次のコマンドを実行して、Contrast サーバをシャットダウンします。
   ```
   $CONTRAST_INSTALLATION/bin/contrast-server stop
   ```
8. `contrast-sersver.vmoptions` ファイルを使用した場合、手順 4 で追加したオプションを削除します。

9. 通常通りに Contrast サーバを起動してから、シェルを終了します。
   環境変数を使用してパスワードをリセットした場合、この手順により `INSTALL4J_ADD_VM_PARAMS` 環境変数からパスワードがクリアされます。

Windows で SuperAdmin パスワードをリセット

Contrast アプリケーションで SuperAdmin のパスワードをリセットするには：

1. Contrast Server サービスを停止します。
2. **cmd.exe** を右クリックして**管理者として実行**を選択し、管理者としてコマンドプロンプト (`cmd.exe`)を起動します。
3. *Contrast\bin* ディレクトリに移動します(ほとんどの場合、これは *C:\Program Files\Contrast\bin* です)。
4. 以下のコマンドを使用して、JVM オプションを編集します。

```
notepad contrast-server.vmoptions
```

5. ファイルに以下のオプションを追加します(youremaildomain.com を自分の E メールドメインに置き換えてください)。

```
-Dreset.superadmin=true
 -Dsuperadmin.username=contrast_superadmin@<your.email.domain.com>
 -Dsuperadmin.password=<password>
```

6. ファイルを保存し、メモ帳を終了します。
7. 次のコマンドを使用して、Contrast サービスを起動します。

```
net start "Contrast Server"
```

8. 新しいパスワードでログインできることを確認します。
9. 次のコマンドを入力して Contrast サービスを停止します。

```
net stop "Contrast Server"
```

10. 次のコマンドを入力して、JVM オプションを編集します。

```
notepad contrast-server.vmoptions
```

11. ステップ 5 で追加したオプションを削除します。
12. ファイルを保存し、メモ帳を終了します。
13. コマンドプロンプトを終了します。
14. Contrast Server サービスを通常どおりに(サービスコントロールパネルから)起動します。

キーの管理

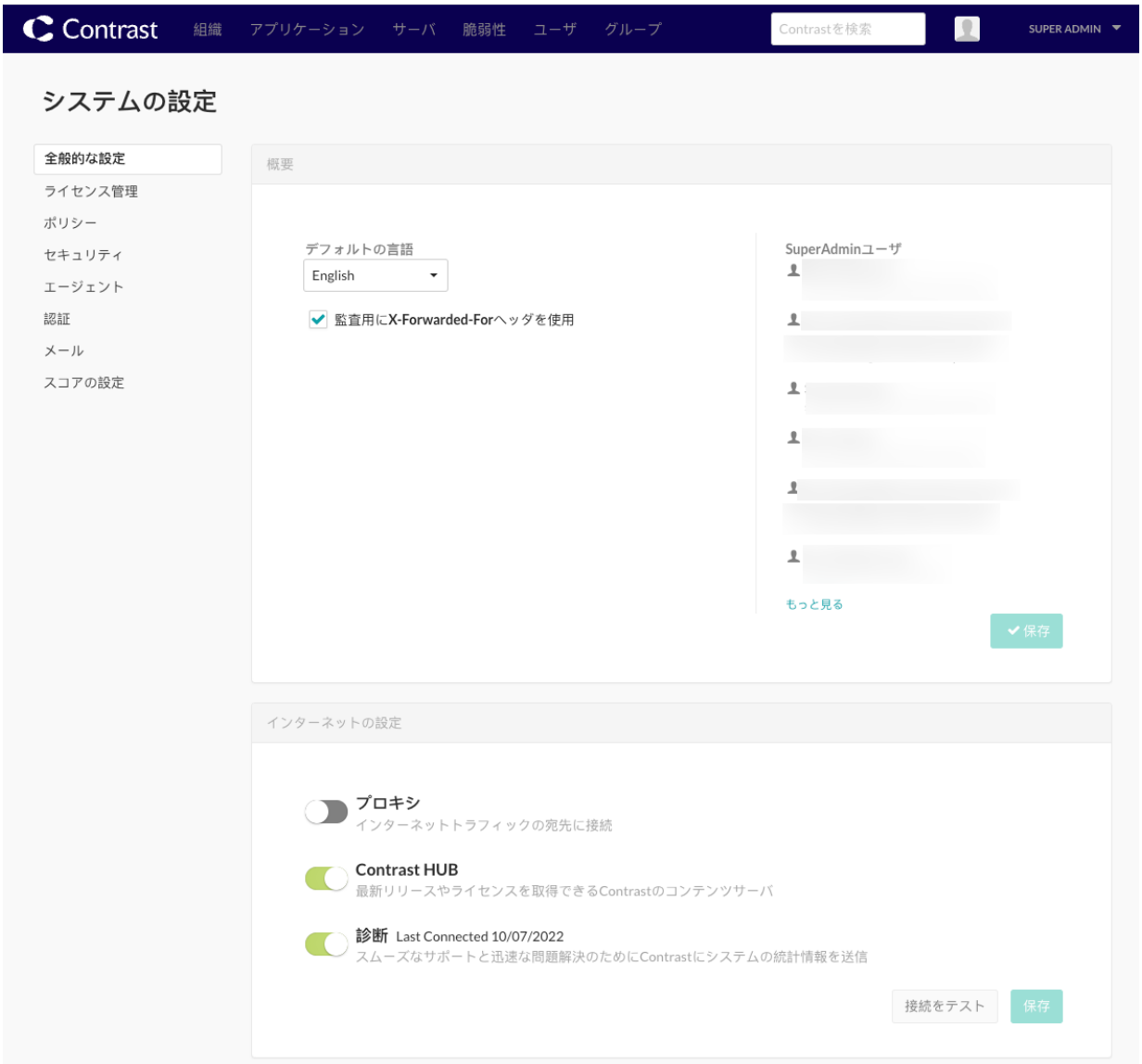
システム管理者は、全ての組織でキーにポリシーを設定して、常にセキュリティ基準を満たすようにすることができます。

1. システムの設定 (936ページ)の左ナビゲーションで**セキュリティ**を選択します。
2. **API キー**のセクションで、 キーに必要な文字数、数字、大文字および小文字の最小数を選択します。これらの規則は、API キーのローテーション (61ページ)時に適用されます。
3. **ログオン時に無効な IP をマスクする**場合は、画面上部のチェックボックスをオンにします。
4. **保存**を選択します。

システムの設定

システムの設定のページでは、システムレベルの標準の設定を指定できます。

Contrast を分散環境で構成する (886ページ)場合には、異なる設定が必要になります。



手順

1. スーパー管理者(SuperAdmin)、サーバ管理者(ServerAdmin)またはシステム管理者(SystemAdmin)としてログインします。
2. **ユーザメニュー**で **SuperAdmin** を選択します。
3. **ユーザメニュー**で**システムの設定**を選択します。
   以下を利用できます。
   - 全般的な設定 (938ページ)
   - ライセンス管理 (939ページ)
   - ポリシー (942ページ) (ライブラリコンプライアンスの設定)
   - セキュリティ(パスワード (934ページ)、2 段階認証 (521ページ)、キーの管理 (936ページ))
   - エージェントキー(組織の設定にあるものと同じエージェントキー (61ページ)です)
   - 認証 (920ページ)
   - メール (943ページ)
   - スコアの設定 (834ページ)

その他のシステム設定

システムの設定は、上記に加えて以下が可能です。

- Tomcat の設定 (891ページ)
- Java ランタイム環境(JRE)の設定 (892ページ)
- HTTPS の設定 (892ページ)
- 組織の追加 (908ページ)
- ログレベルの設定 (900ページ)
- レポート用ストレージの設定 (898ページ)

システムの全般的な設定

全般的な設定は、システムの設定 (936ページ)の一部です。ここでは、以下の設定を構成できます。

- **デフォルトの言語**：ユーザインターフェイスの言語を選択します。
- **X-Forwarded-For ヘッダを使用**：このヘッダを監査に使用する場合は、チェックボックスをオンにします。
- **プロキシ**：インターネットトラフィックの宛先に接続します。
- **Contrast HUB**：Contrast Hub (ハブ)と連携して、ライブラリと CVE を更新します。
- **診断**：スムーズなサポートと迅速な問題解決のために Contrast の稼働状態の統計を送信します。
- **エージェント診断**：ルールやパフォーマンスを改善し、製品の改善に優先順位を付けるために Contrast エージェントのデータを送信します。
- **Heroku の設定**：Heroku 連携のパスワードと SSO ソルトを入力します。

システムレベルの診断情報の管理

Contrast では、お客様の製品の利用状況を測定する診断情報を収集し、より迅速で積極的なサポートや新機能を提供できるようにしています。

Contrast がホストするプラットフォーム上の診断サービスに対して、関連するデータ要素と集計のスナップショットを、定期的に送信することができます。顧客または組織の特定に使用される可能性のあるデータは、一方向ハッシュを使用して隠され、転送中と保存中の両方で暗号化されます。プライバシー保護の観点から、データにはアプリケーション名、個人を特定する情報、コード、脆弱性 ID、お客様のネットワーク識別子は含まれません。

データは Contrast の診断データベースに保存され、アクセス許可のあるサポート担当と開発担当にて分析やレポートに利用されます。データベース内でデータは顧客に帰属し、Contrast サポートがお客様により良いサポートをするための手掛かりとして使用されます。

診断情報により、主に次の 3 つの点でカスタマーサポートが向上します。

- カスタマーサポートは、指標となる診断データを分析し、お客様に積極的に連絡して問題を未然に防ぐことができます。
- データを使用することで、既存の問題を迅速に診断し、サポートケースを正常に解決するためのサイクル時間を短縮できます。
- 導入や使用状況に関する情報を得ることで、Contrast の製品開発部門が、お客様のニーズにより適した新機能を提供できるようになります。

SuperAdmin、ServerAdmin または SystemAdmin として、オンプレミスシステム全体の診断情報の収集を有効または無効にできます。

1. システムの設定の左側のナビゲーションで、**全般的な設定**を選択します。
2. **インターネットの設定**で、**診断**の設定を使用して、この機能を有効または無効にします。診断機能はデフォルトで有効になっています。
   プロキシの設定は、Contrast HUB と診断の設定に適用されます。

ヒント
REST API を使用して、この診断機能で Contrast に送信されるデータをプレビューすることもできます。

システムレベルでのライセンスの割当て

Contrast のライセンスには、次のような種類があります。

- オンプレミス版のお客様は、Contrast サーバをインストール (879ページ)およびアップグレード (902ページ)するためのライセンスが必要です。
- SaaS 版およびオンプレミス版のお客様は、Assess(アプリケーションライセンス)、Protect(サーバライセンス)、SCA(ライブラリライセンス)の各製品のライセンスが必要です。

以下の手順に従って、Assess ライセンスや Protect ライセンスを組織に割り当てます。

開始する前に

- オンプレミス版のお客様の場合、Assess および Protect ライセンスを特定の組織に割り当てるには、SuperAdmin ロールか ServerAdmin ロールが必要です。
  SaaS 版をご利用の場合は、Contrast Security がこのライセンス処理を行います。
- アプリケーションに適用されたライセンスは、最大許容アプリケーション数に対して永続的にカウントされます。ライセンスが適用されたアプリケーションを削除しても、アプリケーションに適用可能な最大ライセンス数に影響はありません。

ライセンスを割り当てる手順

1. **ユーザメニュー**で **SuperAdmin** または **ServerAdmin** を選択します。
   組織の一覧が表示されます。**ライセンス**列には、各組織で使用可能な Assess ライセンスと Protect ライセンスの総数が表示され、続けて未使用のライセンス数が括弧書きで表示されます。
2. 以下のいずれかの方法で「ライセンス概要」を開きます。
   - 組織ページの「ライセンス」列で **Assess** または **Protect** を選択。

   - 組織の行の右端にある三角形( )を選択して**ライセンス概要**を選択。

3. 利用可能な Assess ライセンスを組織に追加するには：

a. Assess のライセンスバーの上にある、**ライセンスを追加**を選択します。
b. **Assess ライセンス**に、この組織で利用できるようにするライセンスの数を入力します。
c. **有効期限**には、追加するライセンスの有効期限を入力します。
d. **割り当て**を選択します。

4. 使用されていない Assess ライセンスを組織から削除する場合は、Assess ライセンスバーの下にある**使用されていないライセンスを取消**を選択します。
5. 組織に割り当てられている、購入済み Protect ライセンスの数を変更するには：
   a. Protect ライセンスバーの上のある**ライセンスの割り当てを変更**を選択します。
   b. **ライセンスの割り当て**に、組織で使用させたいライセンス数を入力します。
   デフォルト値は、購入したライセンス数です。組織のライセンスを取り消すには、購入したライセンス数より少ない値を入力してください。購入したライセンス数以上のライセンスを取り消すことはできます。
   c. **変更を保存**を選択します。

ライセンス情報と設定に関する手順

1. 各組織で利用可能なライセンス数を表示するには、ナビゲーションバーの**組織**を選択します。**ライセンス**列には、各組織で使用可能な Assess ライセンスと Protect ライセンスの総数が表示され、その後に括弧で未使用のライセンス数が表示されます。
   Assess ライセンスの有効期限が近い場合は、赤い警告アイコンが表示されます。アイコンにカーソルを合わせると、失効するライセンス数が表示されます。
2. 利用可能なライセンス数や、ライセンスがないアプリケーション数· サーバ数を参照するには、ユーザメニューから**システムの設定 > ライセンス管理**を選択します。
3. 新しいアプリケーションまたはサーバに自動でライセンスを適用するには、「ライセンス管理」で「Assess ライセンス」または「Protect ライセンス」のトグルを選択します。次に、この設定をすべてのアプリケーションまたはサーバに適用するか、新しいものだけに適用するかを選択します。
   SuperAdmin、ServerAdmin、SystemAdmin は、組織作成 (908ページ)時にライセンスを自動適用するよう指定することもできます。

4. 「ライセンス管理」の右上で**組織の上書きを許可**するボックスを選択すると、組織管理者はこれらの設定を上書きすることが許可されます(これはデフォルトで有効になっています)。

システムレベルでのスコア設定のカスタマイズ

Contrast ではアプリケーションのスコア (958ページ)が算出されますが、 必要に応じてライブラリのスコア (610ページ)に依存させることができます。 システムレベルでスコア設定をカスタマイズするには：

1. システムの設定 (936ページ)で**スコアの設定**を選択します。
2. **総合スコア**でオプションを選択して、Contrast でアプリケーションをどのようにスコア付けするかを決定します。
   - **デフォルトの評価方法**は、アプリケーションのライブラリのスコア (610ページ)とカスタムコードのスコアの平均値です。
   - **カスタムコードのみを評価対象とする**場合は、アプリケーション全体のスコア計算にライブラリのスコアが含まれなくなります。このオプションを選択した場合、特定の言語を選択するか、全ての言語に適用するかを、クリックして選択します。
3. **ライブラリのスコア**でオプションを選択して、Contrast でライブラリをどのようにスコア付けするかを決定します。
   - **デフォルトの評価方法**では、ライブラリの脆弱性だけでなく、ライブラリの古さやバージョン管理を含むアルゴリズムが使用されます。
   - **脆弱性のみを評価対象とする**方法では、ライブラリに存在する脆弱性のみに基づいてスコアが計算されます。
4. **組織の上書きを許可**の横にあるチェックボックスを選択すると、組織管理者が組織レベルでスコアの設定を決定 (834ページ)できるようになります。
5. **保存**を選択します。

注記

ルール管理者(Rules Admin)は、**ポリシーの管理**でポリシーの設定を行い、ポリシーに違反するライブラリを自動的に低いスコア(F)にすることができます。この設定が選択されると、スコアの設定に警告メッセージが表示されます。警告にあるポリシーのリンクをクリックすると、ライブラリポリシーページに移動します。ここで、管理者はそれらの設定を確認して更新ができます。

ライブラリコンプライアンスポリシーの管理

ライブラリコンプライアンスポリシーは、システムレベルで管理できます。ライブラリコンプライアンスポリシーでは、アプリケーションが安全に使用できるライブラリを制限でき、特定のライブラリにバージョン要件を設定できます。Contrast では、制限されたライブラリを使用するアプリケーションや、コンプライアンスポリシーに違反するライブラリにフラグを立てたり、スコアを低くさせることができます。

ライブラリコンプライアンスポリシーを管理するには：

1. **ユーザメニュー**(Contrast の右上にある自分の名前)を開き、システムの設定 (936ページ)を選択します。
2. **ポリシー**を選択します。
3. ポリシーのコンプライアンス要件(使用が制限されたライブラリ、ライブラリバージョンの要件、Contrast でポリシー違反のライブラリを不合格にするか)を設定します。
4. 組織の管理者(Admin)またはルール管理者(Rules Admin)が組織レベルでコンプライアンスポリシーを設定 (807ページ)する場合は、**組織の上書きを許可**を選択します。

システムレベルでの E メール通知の管理

システム管理者は、Contrast が適切な SMTP システムと通信してこれらの通知を受信できるように設定し、有効/無効を切り替えることができます。

通知により、Contrast ユーザはアプリケーションの脆弱性の検出や攻撃、またはパスワードがリセットされた場合など、特定の状況に関するアラートを受信することができます。

組織の管理者は、組織レベルで Contrast の通知のデフォルトを設定 (831ページ)できます。個々のユーザは各自の設定を調整 (523ページ)できます。

システムレベルで通知を設定するには：

1. システムの設定 (936ページ)の左側のナビゲーションで**メール**を選択します。
2. 以下の設定を行います。
   - **メールの有効化**： トグルボタンを使用して、この機能を有効/無効にします。
   - **メールプロトコル**： 値は「SMTP」または「SMTPs」になります。
   - **メールホスト**： SMTP サーバの完全修飾アドレスです。
   - **メールポート**： 可能性のある値は「25」です。
   - **SMTP 認証を使用**：この設定を有効にするには、チェックボックスをオンにします。
   - **メールユーザ**： SMTP システムでの認証目的のユーザアカウントです。
   - **メールパスワード**： SMTP システムに関連付けられたメールユーザのパスワードです。
   - **差出人**： システム通知の送信元の電子メールアドレスを入力します。
   - **STARTTLSL を有効にする**：この設定を有効にするには、チェックボックスをオンにします。
3. **保存**を選択します。

オンプレミス版 Contrast のメンテナンス

システム管理者には、システムの保守に必要な継続的な作業があります。以下の作業が必要になる場合があります。

- MySQL データベースのバックアップ (945ページ)
- パフォーマンスの改善 (30ページ)
- Contrast のアップグレード (902ページ)
- エージェントのアップグレード (904ページ)
- ライブラリデータの更新 (904ページ)
- Contrast ライセンスの更新 (906ページ)
- IP アドレスの更新 (904ページ)
- SSL の管理 (947ページ)
- 暗号化プロパティエディタの使用 (943ページ)

暗号化プロパティエディタの使用

Contrast では、*$CONTRAST_HOME/data/conf* ディレクトリにいくつかの設定ファイルが含まれています。デフォルトでは、Contrast はセキュリティのために設定ファイルを暗号化しますが、Contrast のワークフローを介してこれらのファイルの一部を変更できます。

例えば、以下はオンプレミス版インストール用の暗号化されたプロパティファイルの一部です。

名前	内容
ad.properties	Active Directory グループの認証用に Contrast を接続および構成するための設定。
ldap.properties	LDAP グループの認証用に Contrast を接続および構成するための設定。
database.properties	Contrast と MySQL の間で通信するためのホストと接続の設定。
saml.properties	SAML キーストアのセキュリティ設定

Contrast には、これらのファイルを復号化し設定を支援する編集ツールも含まれています。このツールは、Contrast の実行 (889ページ)中に、アプリケーションの外部の暗号化されたプロパティファイルか

ら値を取得したり、自動パスワードローテーションなど、ファイル内のプロパティを自動的に更新したりする必要がある場合に役立ちます。

暗号化されたプロパティファイルを編集するには：

1. *$CONTRAST_HOME/bin* ディレクトリで以下の復号化ツールを見つけます。
   - **Linux**：`edit-properties` という名前のシェルスクリプト
   - **Windows**：`edit-properties.exe` という名前の Windows コマンドファイル
2. コマンドプロンプトからツールを実行します。これにより、暗号化されたプロパティの値を更新できるアプリケーションが開きます。

   ```
   $CONTRAST_HOME/bin/edit-properties -e $CONTRAST_HOME/data/esapi -f \
   $CONTRAST_HOME/data/conf/ad.properties
   ```
3. 暗号化されたプロパティファイルを表示または編集するには、入力の詳細を指定する必要があります。入力が必要な基本項目は以下のとおりです。
   - *ESAPI.properties* へのパス。
   - 編集するターゲットプロパティファイル。

   暗号化プロパティエディタ用にこの情報を見つけるには、引数を指定せずに `edit-properties` を実行します。

   ```
   contrast@EOP-TeamServer:~/contrast/bin$ ./edit-properties

   usage: property-editor
    -c,--comment <text>      The comment for the top of the file
    -e,--esapi <path>        The path to the ESAPI.properties file
    -f,--targetFile <file>   The properties file to edit
    -o,--print-value         Print out the value of the property and exit
    -p,--property <name>     The name of the property to set
    -v,--value <val>         The value of the property
   ```
4. この例は、暗号化ファイルの編集方法を示しています。*ESAPI.properties* へのパスと編集するターゲットプロパティファイルを指定します。ファイル内で暗号化されている編集可能な既存の値が表示されます。上記の usage オプションでは、単一のプロパティを表示または編集できます。

   ```
   contrast@TeamServer:~/contrast/bin$ ./edit-properties -e ../data/esapi/ -
   f ../data/conf/ad.properties

   ad.userDn                                          : cn=Directory Manager
   ad.identity.attribute.name                         : mail
   ad.password                                        : NotaRealPassword
   ad.nested.groups.enabled                           : false
   ad.group.users                                     : \
   cn=ContrastUsers,cn=Users,dc=contrastsecurity,dc=com
   ad.group.admin                                     : \
   cn=ContrastAdmins,cn=Users,dc=contrastsecurity,dc=com
   ad.url                                             : ldap://localhost:389
   ad.base                                            : \
   dc=contrastsecurity,dc=com
   ```
5. プロパティの暗号化されていない値を取得または更新することもできます。値を取得するには、プロパティエディタに別のパラメータを渡します。この例では、データベースプロパティに関する詳細を調べています。

   ```
   $CONTRAST_HOME/bin/edit-properties \
      -e $CONTRAST_HOME/data/esapi \
      -f $CONTRAST_HOME/data/conf/database.properties \
      -p jdbc.username \
      -o
   ```

暗号化されていない値を更新するには、プロパティエディタに別の引数セットを渡します。

```
$CONTRAST_HOME/bin/edit-properties \
   -e $CONTRAST_HOME/data/esapi \
   -f $CONTRAST_HOME/data/conf/database.properties \
   -p jdbc.username \
   -v joe.user \
   -c "Updating JDBC Password"
```

注記
暗号化されているプロパティファイルを編集した場合は、編集内容を示すコメントを追加しておきます。これは、設定の変更を追跡する必要がある監査担当者などのために役立ちます。

MySQL のバックアップ

次の手順を使用して、Contrast インストーラで作成された MySQL のデータベースを管理します。

これらの手順は、お客様が分散環境用に作成した MySQL データベースには適用できません。

- MySQL の自動バックアップを作成する (945ページ)
- MySQL を手動でバックアップする (945ページ)
- データベースのバックアップを復元する (946ページ)
- 自動バックアップを無効にする (946ページ)

MySQL を手動でバックアップする

Contrast に含まれる `backup-db` スクリプトを使用しても、バックアップが可能です。

開始する前に

- `backup-db` スクリプトを実行するための権限があることを確認してください。通常、これを行うには、Contrast のインストール所有者、root、Windows の管理者アカウントである必要があります。
- Contrast が実行中であり、MySQL が利用可能であることを確認してください。

手順

1. まだ行っていない場合は、データベースバックアップの場所を設定します。暗号化エディタ (943ページ)を使用して*$CONTRAST_HOME/data/conf/database.properties* ファイルを編集することで、`database.bk.dir` の場所を設定します。
2. 環境に応じたバックアップコマンドを実行します。
   - **Windows:**`$CONTRAST_HOME\bin\backup-db.cmd`
   - **Linux:**`$CONTRAST_HOME/bin/backup-db.sh`

MySQL の自動バックアップを作成する

Contrast の MySQL データベースのバックアップを定期的にスケジュールして作成することができます。このオプションはインストール (879ページ)時に選択して、データベースのバックアップの保存先と時刻を指定することができます。

インストール時にこの手順を省略しても、後で Contrast を設定してデータベースのバックアップをスケジュールできます。

手順

1. *$CONTRAST_HOME/data/conf/database.properties* で、 Contrast のデータベースの設定を探してください。
2. 暗号化プロパティエディタを使用 (943ページ)して、データベースの設定を判別します。以下の例は、Contrast データベースのバックアップが有効で、スケジュールされており、特定の場所にあることを示しています。オプションを変更する必要がある場合は、これらの設定を編集することができます。

```
contrast@TeamServer:~/contrast/bin$ ./edit-properties -e ../data/esapi/ -
f ../data/conf/database.properties

database.bk.time                                    : 4:0:0
database.bk.enabled                                 : true
database.bk.dir                                     : /mnt/backups/mysql/
contrast
```

3. Contrast をアップグレードする場合は、最後にスケジュールしたバックアップ以降に作成・更新されたデータを取り込む必要があります。

自動バックアップを無効にする

Contrast の MySQL データベースの自動バックアップは停止することができます。スケジュールされたバックアップは、Windows では `schtasks`、Linux では `crontab` を使用して実行されます。

自動バックアップを無効にするには：

Windows の場合、タスクスケジューラを使用して `ContrastBackup` を無効にするか、削除します。

Linux の場合：

1. Contrast をインストールしたユーザに切り替えて `crontab -l` を実行します。
2. これにより、スケジュールされたジョブの一覧が表示されます。以下のように表示されます。

```
0 2 * * * /usr/local/contrast/bin/backup-db.sh
```

3. バックアップを 1 つ削除する場合には、`crontab -e` を実行します。全てのバックアップを削除するには、`crontab -r` を実行します。

> **注意**
> `-e` オプションでは、Vim で編集して選択したバックアップを削除できます。`-r` オプションを使用すると全てが削除されます。使用の際は注意が必要です。

データベースのバックアップを復元する

次の手順は、Contrast インストーラで作成された MySQL データベース用の復元手順です。

この手順は、お客様が分散環境用に作成した MySQL データベースには適用できません。

開始する前に

データベースの復元は、MySQL データベース管理者が実行する必要があります。

手順

1. 暗号化プロパティエディタ (943ページ)を使用して、MySQL データベースの設定を確認します。

2. Contrast をシャットダウンします。
3. Contrast にパッケージ化された MySQL サービスを使用して、MySQL を個別に起動します。<YourPath>は、Contrast ホームディレクトリへのパスに置き換えてください。
   - **Windows：**

     ```
     "<YourPath>\mysql\bin\mysqld.exe" --defaults-
     file="<YourPath>\data\conf\my.cnf"
     ```
   - **Linux：**

     ```
     sudo -u contrast_service <YourPath>/mysql/bin/mysqld --defaults-
     file=<YourPath>/data/conf/my.cnf --basedir=<YourPath>/mysql --
     datadir=<YourPath>/data/db --plugin-dir=<YourPath>/mysql/lib/plugin --
     lc-messages-dir=<YourPath>/mysql/share --tmpdir=/tmp --lc-
     messages=en_US --log-error=<YourPath>/logs/mysql_error.log --pid-
     file=<YourPath>/data/proc/MysqldResource.pid --port=13306
     ```
4. MySQL に接続します。<jdbc.host>、<jdbc.port>、<jdbc.user>、<jdbc.schema>を、ご利用のホスト、ポート、ユーザ、スキーマに置き換えてください。
   - **Windows：**

     ```
     mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema>
     ```
   - **Linux：**

     ```
     ./mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema>
     ```
5. `drop database <jdbc.schema>;`で、Contrast データベースをドロップします。
6. `create database <jdbc.schema>;`で、Contrast データベースを作成します。
7. `GRANT ALL PRIVILEGES ON *.* to 'contrast'@'%';`で、Contrast ユーザに権限を付与します。
8. MySQL を終了(exit)します。
9. MySQL のバックアップを復元します。<backup_location>をバックアップファイルの場所に、<backup_filename>はバックアップファイル名に置き換えてください。
   - **Windows：**

     ```
     mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema> < \
     <backup_location>/<backup_filename>
     ```
   - **Linux：**

     ```
     ./mysql -h <jdbc.host> -P <jdbc.port> -u <jdbc.user> -p <jdbc.schema> \
     < <backup_location>/<backup_filename>
     ```
10. MySQL をシャットダウンします。
    - **Windows：**
      Windows のサービスマネージャーを使用して、MySQL サービスをシャットダウンします。
    - **Linux：**

      ```
      $CONTRAST_HOME/mysql/bin/mysqladmin.exe shutdown -h localhost -P \
      13306 -u contrast -p
      ```

      暗号化プロパティエディタで設定されたパスワードの入力が必要になります。
11. 完全に復元された Contrast と MySQL を一緒に再起動します。

SSL の管理

オンプレミス版のお客様は、以下の状況で Secure Sockets Layer (SSL)を使用する必要がある場合があります。

- HTTPS プロキシ認証 (934ページ)を設定する場合
- LDAP (926ページ)または Active Directory (922ページ)と連携する場合

- エージェントと Contrast アプリケーション間の通信を保護する場合

リファレンス

Contrast の使用方法に関する参考資料として、以下をご利用ください。

- 用語集 (949ページ)
- ロールと権限 (953ページ)
- エージェントのサポート対象テクノロジ(Java (75ページ)、.NET Framework (167ページ)、(旧).NET Framework、.NET Core (225ページ)、Node.js (285ページ)、Python (360ページ)、Ruby (416ページ)、Go (476ページ))
- アプリケーションのスコアガイド (958ページ)
- ライブラリのスコアガイド (610ページ)
- ログレベル (960ページ)
- 対応ブラウザ (963ページ)
- アプリケーションの例外の正規表現 (806ページ)

用語集

ここでは、Contrast のユーザ向けに各用語の定義を掲載します。他のトピック内でこれらの用語にカーソルを合わせると、文中に定義が表示されます。

Contrast Hub	Contrast Hub は、オンプレミス版をご利用のお客様が Contrast のインストーラとライセンスファイルをダウンロードしたり、エージェントの更新を確認できる場所です。
Contrast コマンドラインインターフェイス (Contrast CLI)	Contrast CLI は、テキストベースのユーザインターフェイスです。開発環境で実行することで、ビルドやデプロイを行う前に早い段階で、オープンソースライブラリのソフトウェアコンポジション解析(SCA)を可視化できます。CLI による結果は、テキストベースのレスポンスで確認することができ、Contrast Web インターフェイスでは依存関係ツリー (609ページ)としても表示されます。
Contrast サービス	Contrast サービスは Go で書かれたプログラムで、Contrast Web インターフェイスと Node.js、Ruby、Python エージェントを接続します。
IP 拒否リスト	IP 拒否リストとは、そのリストにある IP アドレスからの HTTP リクエストをブロックするルールです。
IP 許可リスト	IP 許可リストとは、そのリストにある IP アドレスからの HTTP リクエストを許可するルールです。
LDAP (Lightweight Directory Access Protocol)	LDAP は、ディレクトリサービスに接続・管理するためのクライアントサーバプロトコルの一つです。Contrast のオンプレミス版では、LDAP を使用 (926ページ)してユーザや口グインを管理できます。
OWASP (Open Web Application Security Project)	Open Web Application Security Project®は、セキュリティプロジェクトの共有や会員の教育をサポートするオープンプラットフォームによって、ソフトウェアのセキュリティを向上するための非営利団体です。OWASP Top 10 には、Web アプリケーションにおける最も重大なセキュリティ上の欠陥がリストアップされています。
SAML (Security Assertion Markup Language)	SAML は、シングルサインオン認証など、オンラインパートナー間で認証情報を作成・交換するために使用される XML ベースの標準規格です。

SIEM (Security Incident Event Management)	SIEM システムは、他のシステムからセキュリティイベントや関連データを収集して分析し、脅威の検知やコンプライアンス、セキュリティインシデントの管理をサポートします。Contrast は、いくつかの代表的な SIEM システムと連携できます。
SQL インジェクション (SQLi)	SQL インジェクションは、クライアントからのユーザ入力データ内に SQL クエリを挿入(「インジェクション」)して、アプリケーションに取り込ませる攻撃です。この攻撃の目的は、データベースのデータの読み取りや変更、データベースへのコマンドの送信です(例)。
WAF (Web Application Firewall)	WAF は、Web トラフィックを検査・フィルタリングし、一般的な攻撃からアプリケーションを防御するためのものです。
Webhook	指定したイベントが発生するたびに、あるアプリケーションから別のアプリケーションに対して HTTP 経由でリアルタイムに情報を送信するための仕組みのこと。
[en] dynamic application security testing ([en] DAST)	[en] A security testing technology that is designed to detect conditions that point to a security vulnerability in an application in its running state.
distroless イメージ	Distroless イメージとは、アプリケーションおよびアプリケーション実行時の依存関係のみが含まれるイメージです。パッケージマネージャやシェルなど、標準的な Linux ディストリビューションに含まれるその他のプログラムは含まれていません。
アカウント乗っ取り (ATO)	アカウント乗っ取りは、ログイン認証情報を盗んだり、Web アプリケーションの認証に侵入したりする攻撃の結果です。
アプリケーション	アプリケーションは、Contrast エージェントによって検査されるカスタムコードの論理的なグループです。
アメリカ国立標準技術研究所 (NIST)	NIST は、サイバーセキュリティを含む複数の分野で、計測学、標準規格、産業技術を進歩させることにより、米国の技術革新と産業競争力を促進するための政府機関です。
インストゥルメント (instrument)	ランタイムにデータを観測・報告するソフトウェアエージェントがアプリケーションをモニタリングすることです。Contrast では、アプリケーションに Contrast エージェントを組み込むことを意味します。Contrast エージェントは、疎通されたルートに基づいてアプリケーションの脆弱性データを送信します。
インタラクティブアプリケーションセキュリティテスト (IAST)	実行中のアプリケーション内のデータフローを分析し、セキュリティ上の脆弱性の可能性を検出・報告するセキュリティ技術。
エージェント	エージェントとは、Web アプリケーションにインストールされる言語固有のコードで、セキュリティデータを収集・解析し、必要に応じて検出結果を Contrast に報告します。
クレデンシャルスタッフィング	クレデンシャルスタッフィングはブルートフォース攻撃の一種で、漏洩したユーザ名とパスワードの組み合わせを自動的に実行してユーザアカウントにアクセスします。
クロスサイトスクリプティング (XSS)	クロスサイトスクリプティングとは、ユーザ入力によって悪意のあるスクリプトが Web アプリケーションに注入されて、検証やエンコードを行わずにその出力が生成された場合に発生する攻撃です。
コマンドインジェクション	コマンドインジェクション攻撃は、脆弱なアプリケーションを介してホストのオペレーティングシステムを標的とするものです。この攻撃は、

	ユーザがフォーム、Cookie、HTTP ヘッダ、またはアプリケーションのその他の部分から安全でないデータをシステムシェルに渡すことで発生します。
コンテナイメージ	コンテナイメージは、コンピュータシステム上にコンテナを作成できる実行可能コードが含まれた静的ファイルです。
シンク	シンクとは、データフローの解析 (700ページ)でデータが終着する場所です。シンクは、データが書き込まれる外部フォーマットや出力先のことです。
シングルサインオン (SSO)	シングルサインオンは、ユーザが 1 セットの資格情報だけで複数のシステムにアクセスできるユーザ識別・認証サービスです。
スコア	Contrast では、アプリケーションやライブラリの現在のセキュリティの状況を反映したライブラリ (610ページ)のスコアとアプリケーション (958ページ)のスコアが提供されます。
スタックトレース	スタックトレースは、問題の発生となった一連のイベントを一覧表示するものです。Contrast では、スタックトレースにはセキュリティの脆弱性に至るまでのイベントが表示されます。
ソース	ソースとは、データフローの解析 (700ページ)でデータが発生する場所です。ソースは、システムに入力されるあらゆる入力データやリクエストのことです。
ソフトウェアコンポジション解析 (SCA)	脆弱なライブラリを特定し、CVE の深刻度に基づいてビルドを失敗させ、依存関係ツリーを表示して、ライブラリの依存関係や脆弱性が存在する箇所を把握することができます。
ソフトウェア開発ライフサイクル (SDLC)	企画がソフトウェアの生産に至るまでの一連の工程のこと。
テストカバレッジ	実行されたテストの数を把握するテスト手法です。
パストラバーサル	パストラバーサルは、Web のルートフォルダ外に保存されている重要なシステムファイルやディレクトリにアクセスしようとする攻撃です。この攻撃では、絶対ファイルパスや「ドット-ドット-スラッシュ」(../)の並びでファイルを参照する変数が使用されます。
フーォルスポジティブ(false positive)	誤って脆弱性として報告してしまうこと。誤検知のこと。
フローマップ	フローマップは、Contrast エージェントが組み込まれているアプリケーションを可視化したもので、そのアプリケーションが使用しているすべてのバックエンドシステムと、接続されているその他のアプリケーションを表します。これにより、脆弱なアプリケーションに影響を与えるその他の要素を分析し、リスクを評価できます。
ブルートフォース攻撃	ブルートフォース攻撃は、最終的に正しく推測することを目的として、多くのパスワードやパスフレーズを体系的に送信することです。
プロファイラチェーン	プロファイラチェーンは、パフォーマンスツールや APM ツールなどの他の.NET プロファイラエージェントと一緒に.NET Framework エージェントまたは.NET Core エージェントを実行する方法です。
ポリシー	ポリシーとは、特定のアプリケーションやライブラリについて、指定の条件が満たされたときにセキュリティ違反やステータス変更、通知をト

	リガーする一連のルールのことです。ポリシーにより、アプリケーションやチーム全体でより一貫したセキュリティ基準が保証されます。
マニフェスト	プロジェクトに必要な依存関係を宣言するためにプロジェクトに保存されるファイル。
ライブラリ	ライブラリとは、アプリケーションに含まれるパッケージ化されたコードのことです。ライブラリ (599ページ)には、公開と内製のものがあります。
ランタイムアプリケーションセルフプロテクション (RASP)	Contrast は、RASP の手法を用いて攻撃を監視し、本番環境のアプリケーションを積極的に防御します。
ルール	ルールとは、脆弱性イベントや攻撃イベントを特定するために使用されるセキュリティ制御です。ルールが一致すると、影響を受けるアプリケーションの脆弱性イベントや攻撃イベントをエージェントが Contrast サーバに送信します。
依存関係かく乱	依存関係かく乱は、「置換攻撃(substitution attack)」とも呼ばれ、攻撃者が組織の内部リポジトリに脆弱なコードや悪意のあるコードを入れ込むために、組織の内部ライブラリと同じ名前を公開パッケージのインデックスに登録することで発生します。
環境	Contrast では、アプリケーションの環境として「開発」、「テスト(QA)」、「本番」の 3 つの環境のいずれかを設定できます。
環境変数	環境変数は、実行時にソフトウェアに渡すことができる値で、通常はキーと値のペアであり、アプリケーションの外部で定義します。Contrast では、環境変数はアプリケーションを検査するエージェントを設定する (60ページ)ために使用します。これらの変数によって、利用したいフレームワーク内でアプリケーションが予想どおりに動作し、Contrast で表示したいメタデータが報告されるようになります。
機密データのマスキング	この機能は、Contrast ログや Contrast エージェントから送信されるその他のデータに含まれる機密データを、アプリケーションでのデータ処理に影響を与えることなくデータにマスクをかけます。
共通言語ランタイム (CLR)	共通言語ランタイム(CLR)は、.NET Framework のプログラムを実行するための動作環境です。
共通脆弱性識別子 (CVE)	共通脆弱性識別子(CVE)は一般的に公開されている情報セキュリティの脆弱性のリストで、脆弱性の種類を特定し追跡するために国際的に使用されています。
継続的インテグレーション/継続的デリバリ (CI/CD)	ビルド、テスト、デプロイにおいて継続的な繰返しと自動化を促進するアジャイルの実践を指します。
攻撃	攻撃 (707ページ)は、一定の時間内に発生する 1 つまたは複数の攻撃イベントで構成されます。
攻撃イベント	攻撃イベントとは、Contrast エージェントが組み込まれたアプリケーションにおいて、Protect ルールへの違反やその他の疑わしいアプリケーションアクティビティのことです。イベントは、HTTP リクエストや SQL クエリなどの単一の攻撃ベクトルに該当します。複数の攻撃イベントが 1 つの攻撃を構成しますが、通常は同じコード領域と同じ時間枠内で発生します。

最高情報セキュリティ責任者 (CISO)	最高情報セキュリティ責任者(CISO、Chief Information Security Officer)は、組織の情報セキュリティ対策を指揮し、機密資産が十分に保護されていること、スタッフが脆弱性を管理・防止できることを保証し、実証します。
修復済	*ライブラリのステータス*として、脆弱なライブラリに対応・対策済であることを表します。
静的アプリケーションセキュリティテスト (SAST)	アプリケーションをインストールや実行せずに、アプリケーションに潜在する脆弱性を検出する手法。
報告済	*ライブラリのステータス*として、脆弱性のあるライブラリが Contrast で検出された場合のデフォルトステータスです。
未使用の関数	シャドウ関数とも呼ばれます。90 日以上呼び出されていないクラウド環境上の関数です。
問題無し	*ライブラリのステータス*として、 このライブラリにある脆弱性は認識済みであり、そのリスクは許容できるものを表します。

ロールと権限

ユーザには、割り当てられたロールに応じて権限と機能が付与されます。ロールは、アプリケーション (953ページ)レベルと組織 (955ページ)レベルで、そして(オンプレミス版のお客様の場合は)システム (957ページ)レベルで設定されます。これらのロールの多くは、ユーザがアクセスグループ (822ページ)に割り当てられたときに定義されます。

また、以下が可能です。

- 自分の権限の確認 (523ページ)
- 組織の権限の管理 (822ページ)
- システム権限の管理 (907ページ)
- Protect 権限の付与 (915ページ)

注記

API のみのユーザは、Contrast の REST API にはアクセスできますが、ユーザインターフェイスにはログインできません。Contrast では、API の管理者アカウントの作成を推奨していません。

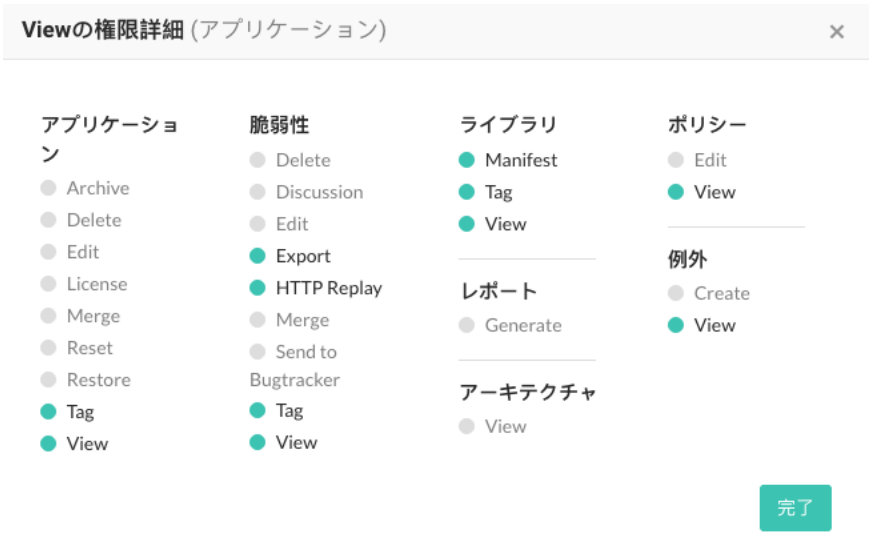
アプリケーションロール

アプリケーションロールは、特定のアプリケーション内で権限や機能をユーザに与えるものです。組織ロールに加えて、アプリケーションロールを定義して、アプリケーションに対するユーザ権限をより細かく制御します。アプリケーションロールは、アクセスグループ (822ページ)に割り当てます。

以下のアプリケーションロールを使用して、アプリケーション内の権限と機能を付与します。

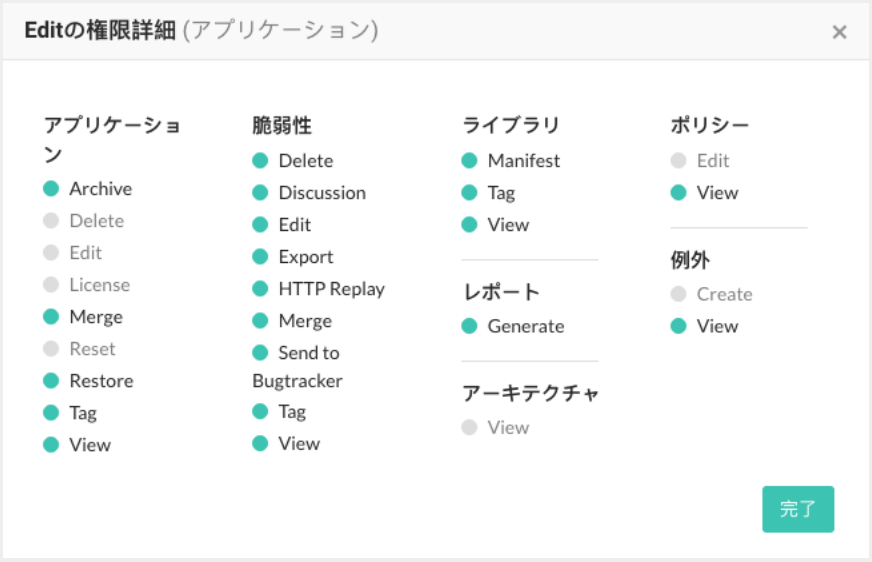
View

アプリケーションの View ロール(アプリケーションの閲覧者)は、Contrast インターフェイスに読み取り専用でアクセスできます。アプリケーション情報の編集はできず、スコア、ライブラリ、脆弱性およびコメントなどを参照するのみに制限されます。



Edit

アプリケーションの Edit ロール(アプリケーションの編集者)は、検出結果の対策、タグの追加、脆弱性の管理、属性の編集、アプリケーションのマージ、アプリケーションの追加・削除、サーバの登録などが可能です。Contrast 利用者の大半に割り当てるのが、このロールです。



Rules Admin	アプリケーションの Rules Admin ロール(アプリケーションのルール管理者)は、Edit ロールの機能に加えて、ルールの編集ができます。アプリケーションのルールやポリシ - の編集、Protect の有効化、アプリケーションに関する通知やスコア付けの管理などができます。
Admin	アプリケーションの Admin ロール(アプリケーション管理者)には制限がなく、他のユーザのアプリケーションへのアクセスを管理できます。

No Access ロールは、アプリケーションへのユーザアクセスをブロックします。

アプリケーションロールの追加は、組織のアクセスグループの作成または編集 (822ページ)で行います。

関連項目

権限の表示 (523ページ)

組織ロール

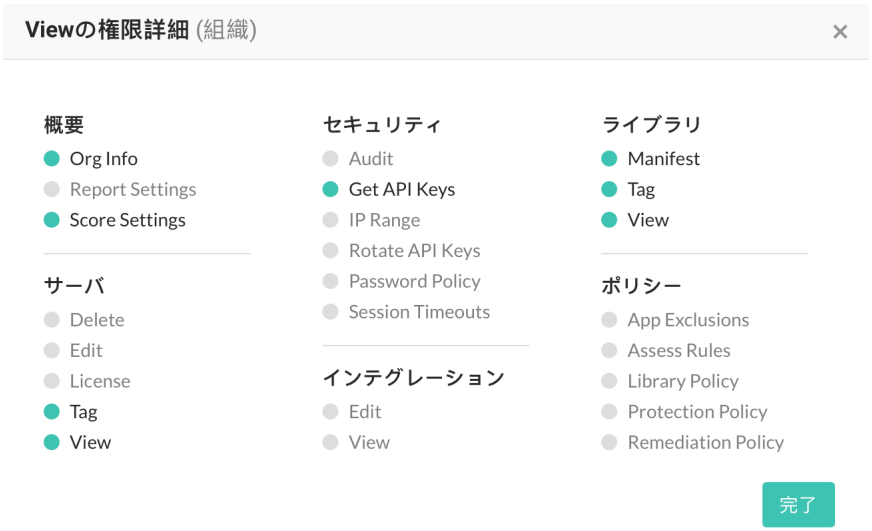
組織ロールは、ユーザが所属する組織内のアクセスを制御するものです。ユーザには、組織によって異なるロールを割り当てることができます。

全てのユーザに、デフォルトの組織のデフォルトロールを指定します。

組織ロールは以下のとおりです。

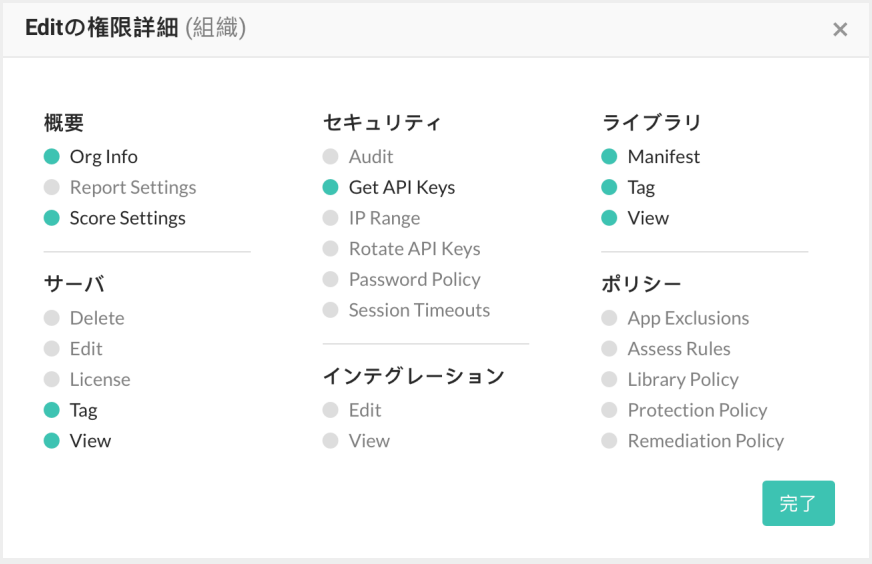
View

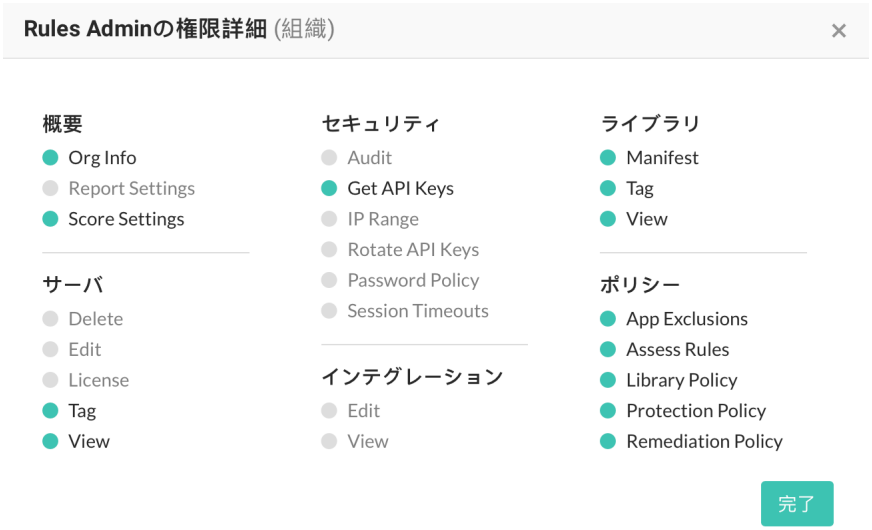
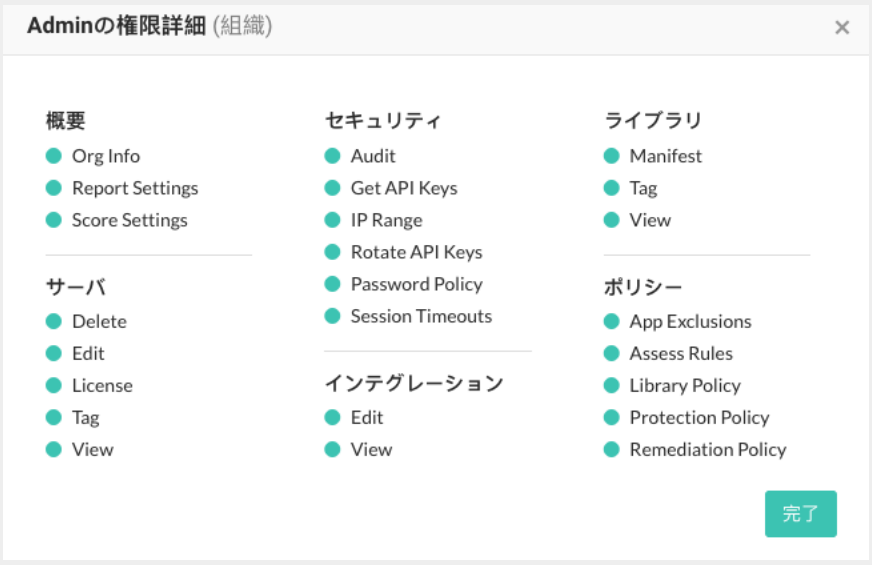
組織の View ロール(組織の閲覧者)は、Contrast インターフェイスに読み取り専用でアクセスできます。アプリケーション情報の編集はできず、スコア、ライブラリ、脆弱性およびコメントなどを参照するのみに制限されます。



Edit

組織の Edit ロール(組織の編集者)は、検出結果の対策、タグの追加、脆弱性の管理、属性の編集、アプリケーションのマージ、アプリケーションの追加・削除、サーバの登録などが可能です。Contrast 利用者の大半に割り当てるのが、このロールです。



Rules Admin	組織の Rules Admin ロール(組織のルール管理者)は、組織レベルでルールやポリシ - を管理できます。また、Protect の有効化、アプリケーションに関する通知やスコア付けの管理ができます。 
Admin	組織の Admin ロール(組織管理者)は、組織の設定と管理を行います。 

組織ロールは、組織のアクセスグループにユーザを追加 (822ページ)することで割り当てられます。

関連項目

権限の表示 (523ページ)

システムロール

注記
システムロールは、オンプレミス版のお客様のみが利用できます。

どのようにシステム管理を行う (907ページ)かを決める際に、システムロールを使用して権限や機能を割り当てることができます。

- **SuperAdmin**：スーパー管理者。オンプレミス版インスタンスのインストールと設定を担当します。このロールは、Contrast のシステム管理も担当するものにもなりますが、1 人以上の管理者に割り当てることができます。**SuperAdmin** は、組織、アプリケーション、サーバ、脆弱性、ユーザ、グループを設定できます。
- **ServerAdmin**：サーバ管理者。ユーザやグループにアクセスできない以外は、SuperAdmin と同じです。ユーザメニューの **ServerAdmin** オプションを利用し、組織、アプリケーション、サーバおよび脆弱性を設定できます。
- **SystemAdmin**：システム管理者。組織やグループの管理を担当します。ユーザメニューの **SuperAdmin** オプションを利用し、組織、アプリケーション、サーバ、脆弱性、ユーザおよびグループを設定できます。
- **Observer**：システムのオブザーバ。組織、ユーザ、アプリケーション、グループ、およびトレースの読み取り専用権限があります。ユーザメニューの **Observer** オプションで読み取り専用アクセスを利用し、組織、アプリケーション、サーバ、脆弱性、ユーザを参照できます。
- **No Access**：アクセス権なし。このロールは、指定の組織へのユーザのアクセスをブロックします。

アプリケーションのスコアガイド

Contrast では、アプリケーションは文字で表したグレードでスコア付けされ、アプリケーションの全体的な評価を把握するのに役立ちます。

アプリケーションのスコアは、アプリケーションがどれだけ疎通されているかによりますが、アプリケーションで検出された脆弱性の数と深刻度に基づき算出されます。

算出されたスコアはグレード(A から F の文字)に置き換えられて、Contrast で表示されます。

- **A**：90-100
- **B**：80-89
- **C**：70-79
- **D**：60-69
- **F**：35-59

アプリケーションのスコアの算出には、アプリケーションのライブラリのスコア (610ページ)とカスタムコードのスコアの平均を求めます。

カスタムコードのスコアの算出は 100 点から始まり、アプリケーションで検出された脆弱性の数に、以下に示す深刻度に応じたペナルティの重み付けを掛けたペナルティ点数が差し引かれます。

- **重大**：脆弱性の数×20
- **高**：脆弱性の数×10
- **中**：脆弱性の数×5
- **低**：脆弱性の数×1

脆弱性の重み付けは、悪用される可能性やその影響の深刻さにより異なります。

例えば、SQL インジェクションは、専門知識が無くても悪用できる自動化ツールなどが存在するため、深刻度を**重大**としています。アプリケーションやスキーマについて何も知らない攻撃者が、データベースの全ての内容を盗み出せる可能性があります。

一方、SHA-1 などのハッシュアルゴリズムの使用は、深刻な弱点があることで知られているため、**低**リスクと見なされます。また、実際に悪用するには、組織や国家レベルでの大規模な支援のもと、非常に熟練した攻撃者のリソースが必要です。

ヒント

アプリケーションのスコアの算出例：

まず、カスタムコードのスコアを決定します。アプリケーションの脆弱性が、**重大**が 0、**高**が 1、**中**が 2、**低**が 1 だった場合、カスタムコードのスコアは以下のようになります。

```
100 - (20 X 0) - (10 X 1) - (5 X 2) - (1 X 1)= 79
```

ライブラリのスコアが 85、カスタムコードのスコアが 79 のアプリケーションの場合、アプリケーションのスコアは 82 となり、B という評価になります。

```
85 + 79 = 164
164/2 = 82
```

スコアを改善するには：

- CVE シールドや Protect ルールを有効 (792ページ)にして、防御した脆弱性をスコアの計算から除きます。
- カスタムコードに存在する**重大**・**高**リスクの脆弱性を修正します。
- 脆弱なライブラリに対処します。
- **高リスク**のライブラリを更新します。

ライブラリのスコアガイド

Contrast では、アプリケーションのライブラリの安全性を文字で表したグレードで提供し、分析時の目安として使用することができます。以下のとおり、スコアをグレード(A から F の文字)に置き換えます。

- A： 90 - 100
- B：80 - 89
- C：70 - 79
- D：60 - 69
- F：35 - 59

スコアは、次の 3 つのペナルティ要素に基づきます。

- **時間**：ライブラリの古さです。アプリケーションで使用されているバージョンのリリース日から最新バージョンのリリース日までの年数に、2.5 を掛けた数が減算されます。
- **ステータス**：ステータスは、ライブラリの日付以降のバージョン数です。アプリケーションの現在のライブラリ以降にリリースされたバージョンの数に、10 を掛けた数が減算されます。
- **セキュリティ**：ライブラリに影響を与える CVE 数です。ライブラリの CVE ペナルティは、このライブラリの全ての既知の CVE の中で最も高い深刻度に、10 を掛けた数が減算されます。

注記

組織の管理者は、セキュリティ要素のみを対象とするようスコアの設定方法を調整 (834ページ)できます。

ヒント

例：

2010 年 1 月にリリースされたライブラリを使用していて、最新版が 2013 年 9 月にリリースされた場合は、経過した年数は 2 となります。そのため、時間のペナルティは次のようになります。

```
2 x 2.5 = 5
```

バージョン 1.1.1 を使用しているが、バージョン 1.1.2 と 1.1.3 がリリースされている場合は、ペナルティは次のようになります。

```
2 x 10 = 20
```

セキュリティのスコアに 2.4 と 2.2 があるライブラリがある場合、ペナルティは次のようになります。

```
2.4 x 10 = 24
```

ライブラリの最終的なスコアは、3 つのペナルティそれぞれの値を 100 から引くことによって計算されます。

```
100 - 5 - 20 - 24 = 51
```

51 のスコアには、「F」という文字が評価として割り当てられます。

ログレベル

ログレベルの設定により、サーバのログ出力で処理されるイベントを制御でき、イベントをより効果的に取得できます。ログレベルはシステムレベルで設定 (900ページ)できるほか、Edit 権限を持つユーザが特定のサーバ (593ページ)に対して設定できます。

ログレベルは、Log4j に準拠し、可能な限り Log4j のレベル指定に沿っています。

デフォルト値は INFO です。問題が発生してより詳細なデータ収集が必要にならない限り、ほとんどの場合は ERROR レベルで十分です。

ログレベル	説明
ERROR	対処する必要のある、不安定な状態を招く恐れのある深刻なエラーに関する情報を提供します。
WARN	予期しないイベントについて、ユーザに警告します。このレベルで発生するメッセージでは、システムの進行が停止しない場合があります。
INFO	進捗と設定された状態についての情報を提供します。このレベルはエンドユーザにとって便利です。
DEBUG	開発者がアプリケーションをデバッグするのに役立ちます。記録されるメッセージのレベルは、アプリケーション開発者にサポートを提供することに重点が置かれています。
TRACE	DEBUG レベルよりも詳細な情報を提供します。

イベントと Generic Webhook 変数

NEW_VULNERABILITY や SERVER_OFFLINE などの Contrast のイベントからのデータを使って、Generic Webhook (750ページ)のレスポンスをカスタマイズできます。各イベントには、共通 (752ページ)変数、アプリケーション (753ページ)変数、サーバ (753ページ)変数、または脆弱性 (753ページ)変数があり、ペイロードのリクエストで呼び出すことができます。

イベント	変数
ATTACK_END	共通 (752ページ)、アプリケーション (752ページ)、サーバ (753ページ)

イベント	変数
ATTACK_EVENT_COMMENT	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
ATTACK_UPDATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
EXPIRING_LICENSE	共通 (752ページ)、アプリケーション (753ページ)
NEW_ASSET (新規アプリケーションの場合)	共通 (752ページ)、アプリケーション (753ページ)およびサーバ (753ページ)(新規アプリケーションの場合)
NEW_ATTACK_APPLICATION	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_ATTACK_UPDATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_ATTACK	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)
NEW_VULNERABILITY_COMMENT	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
NEW_VULNERABILITY	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
NEW_VULNERABLE_LIBRARY	共通 (752ページ)、アプリケーション (753ページ)
SERVER_OFFLINE	共通 (752ページ)、サーバ (753ページ)
VULNERABILITY_CHANGESTATUS_CLOSED	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
VULNERABILITY_CHANGESTATUS_OPEN	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)
VULNERABILITY_DUPLICATE	共通 (752ページ)、アプリケーション (753ページ)、サーバ (753ページ)、脆弱性 (753ページ)

Generic Webhook の変数

NEW_VULNERABILITY や SERVER_OFFLINE などの Contrast イベントからのデータを使用して、Generic Webhook (750ページ)のレスポンスをカスタマイズできます。各イベントには、ペイロードのリクエストで呼び出すことができる変数があります。変数には、一般的な共通情報として利用するもの、もしくは、アプリケーション用、サーバ用、脆弱性用があります。

変数	説明
共通変数	
$EventType	Webhook のトリガーとなるイベントタイプ 例：SERVER_OFFLINE
$Message	Webhook のトリガーとなったイベントの概要を表すメッセージ
$OrganizationId	Contrast で組織の作成時に、組織に割り当てられた一意の ID
$OrganizationName	組織の名前
$Title	常に“Contrast Security”を返す
アプリケーション変数	
$ApplicationChild	アプリケーションが子アプリケーションの場合は真(true)を返し、そうでない場合は偽(false)を返す
$ApplicationCode	アプリケーションのタイトルに表示されるアプリケーション名の省略形、デフォルトでは空白 例：TEST
$ApplicationContextPath	アプリケーションのコンテキストパス 例： /example/somethingelse
$ApplicationFirstSeen	アプリケーションが最初に検知された日時(Unix 時間) 例：1572033840000
$ApplicationHasParentApp	アプリケーションに親がある場合は真(true)を返し、そうでない場合は偽(false)を返す
$ApplicationImportance	アプリケーションの重要度の列挙値(enum 値) 例：MEDIUM
$ApplicationId	Contrast でアプリケーションの登録時に、アプリケーションに割り当てられた一意の ID 例： 49fe2978-1833-4441-83db-2b7o486d9413

変数	説明
$ApplicationImportanceDescription	アプリケーションに割り当てられた重要度、例：Medium
$ApplicationLanguage	アプリケーションのプログラミング言語
$ApplicationLastSeen	アプリケーションが最後に検知された日時(Unix 時間)、例：1572033840000
$ApplicationLicenseLevel	アプリケーションに Assess ライセンスがあるかどうか、値：Licensed、Unlicensed
$ApplicationMaster	アプリケーションがマスターアプリケーションである場合は真(true)を返し、そうでない場合は偽(false)を返す
$ApplicationName	アプリケーションの名前
$ApplicationParentAppId	Contrast でアプリケーション(この場合は親アプリケーション)の登録時に、アプリケーションに割り当てられた一意の ID(存在する場合) 例： 49fe2978-1833-4441-83db-2b7o486d9413
$ApplicationTags	アプリケーションのタグのカンマ区切りリスト
$ApplicationTotalModules	アプリケーションにあるモジュールの数
サーバ変数	
$Environment	サーバの環境、値：DEVELOPMENT、QA、PRODUCTION
$ServerId	イベントに関与しているサーバの ID 複数サーバが関与している場合、カンマ区切りのサーバ ID リスト
$ServerName	イベントに関与しているサーバの名前 複数サーバが関与している場合、カンマ区切りのサーバ名リスト
脆弱性変数	
$Severity	イベントのトリガーが脆弱性の場合、脆弱性の深刻度を返す
$Status	イベントのトリガーが脆弱性の場合、脆弱性のステータスを返す
$TraceId	イベントのトリガーが脆弱性の場合、脆弱性 ID を返す
$VulnerabilityAgentLanguage	脆弱性が検知されたアプリケーションの言語またはフレームワーク名（例：.Java、.NET、Ruby など）
$VulnerabilityAppVersionTags	脆弱性が検出されたアプリケーションのバージョン 例：v1.2.3
$VulnerabilityAutoRemediatedExpirationPeriod	脆弱性が自動修復される有効期間(Unix 時間) 例：1572033840000
$VulnerabilityBugTrackerTickets	脆弱性情報がバグ管理システムに送信されて、作成されたチケットのリスト(カンマ区切り) 例：ticket1, ticket2, ticket3
$VulnerabilityCategory	検出された脆弱性のカテゴリー、例：Injection
$VulnerabilityClosedTime	脆弱性がクローズされた日時(Unix 時間) 例：1572033840000
$VulnerabilityConfidence	脆弱性の信頼度
$VulnerabilityDefaultSeverity	脆弱性のデフォルトの深刻度
$VulnerabilityDiscovered	脆弱性が最初に検出された日時(Unix 時間) 例：1572033840000
$VulnerabilityEvidence	脆弱性のエビデンス
$VulnerabilityInstanceUuid	Contrast で脆弱性インスタンスの作成時に、脆弱性インスタンスに割り当てられた一意の ID 例： R33T-N00B-TGIF-RM6P
$VulnerabilityFirstTimeSeen	脆弱性が最初に検出された日時(Unix 時間)、例： 1572033840000
$VulnerabilityImpact	脆弱性の影響度レベル、値：Low、Medium、High
$VulnerabilityLastTimeSeen	脆弱性が最後に確認された日時(Unix 時間)、例： 1572033840000
$VulnerabilityInstanceLastTimeSeen	脆弱性が最後に確認された日時(Unix 時間)、例：1572033840000
$VulnerabilityLicenseLevel	脆弱性のライセンスレベル

変数	説明
$VulnerabilityLikelihood	脆弱性の可能性レベル 値：Low、Medium、High
$VulnerabilityReportedToBugTracker	脆弱性がバグ管理システムに送信された日時(Unix 時間) 例：1572033840000
$VulnerabilityReportedToBugTrackerTime	脆弱性がバグ管理システムに送信された場合、真(true)を返す
$VulnerabilityRule	脆弱性に関連付けられたルール
$VulnerabilityRuleName	脆弱性に関連付けられたルールの名前
$VulnerabilityRuleTitle	脆弱性に関連付けられたルールのタイトル
$VulnerabilitySubStatus	脆弱性のサブステータス
$VulnerabilityTags	脆弱性に関連付けられたカスタムタグ 例：my-custom-tag
$VulnerabilityTitle	脆弱性のタイトル
$VulnerabilitySubStatusKeyCode	脆弱性サブステータスのキーコード
$VulnerabilityTotalTracesReceived	脆弱性が受信された合計回数
$VulnerabilityUuid	脆弱性を検索するために使用される一意の ID
$VulnerabilityVisible	脆弱性にライセンスがあり参照可能である場合は真(true)を返し、そうでない場合は偽(false)を返す
$VulnerabilityRule	イベントのトリガーが脆弱性の場合、脆弱性が違反したルールを返す
$VulnerabilityTags	イベントのトリガーが脆弱性の場合、脆弱性に関連付けられたタグのリスト(カンマ区切り)を返す

正規表現リファレンス

アプリケーションの例外を追加 (804ページ)する際には、以下の表と例を参考にしてください。

適用	パターン	パターン例	一致例
文字列の先頭	^	^w+	Start of a string
文字列の末尾	$	w+$	End of a string
後に続く文字列の大文字と小文字を区別しない一致	(?i)	(?i)%0a	%0a or %0A
a、b、または c の内の 1 文字	[abc]	[abc]+	a bb ccc
a、b、c 以外の文字	[^abc]	[^abc]+	Anythingbutabc.
a-z までの範囲内の 1 文字	[a-z]	[a-z]+	Only a-z
a-z までの範囲内にない 1 文字	[^a-z]	[^a-z]+	Anythingbuta-z.
a-z または A-Z の範囲内の 1 文字	[a-zA-Z]	[a-zA-Z]+	abc123DEF
任意の 1 文字	.	.+	abc
空白文字	\s	\s	anywhitespacecharacter
空白以外の文字	\S	\S+	any non-whitespace
10 進数字	\d	\d	not 1 not 2
10 進数字以外	\D	\D+	not 1 not 2
0 個または 1 個の a	a?	ba?	ba b a
0 個以上の a	a*	ba*	a ba baa aaa ba b
1 個以上の a	a+	a+	a aa aaa aaaa bab baab
ちょうど 3 個の a	a{3}	a{3}	a aa aaa aaaa
3 個以上の a	a{3,}	a{3,}	a aa aaa aaaa aaaaaa
3 個以上 6 個以下の a	a{3,6}	a{3,6}	a aa aaa aaaa aaaaaa aaaa
ピリオド(ドット)はリテラル文字	.	a.b	string.string

対応ブラウザ

Contrast は、HTML5 の Web ベースアプリケーションです。インターフェイスは React と AngularJS をベースにしています。最新のブラウザであれば、問題なく動作します。Contrast は、以下のブラウザの**最新のメジャーバージョン**でテストを行なっています。

- Chrome
- Edge
- Firefox
- Safari

Opera ブラウザまたは古いバージョンの Internet Explorer、Firefox、Safari ブラウザでも引き続き機能する場合がありますが、一部の機能は意図したとおりに表示されない可能性があります。

Contrast ベータ版利用規約

Contrast のベータ版製品および機能について、次の利用条件を定めます。

- 本製品は現在開発中であり、継続的な改善を行っています。
- 弊社のベータ版製品は現状のまま提供され、いかなる保証もありません。
- ベータ版製品は開発中であるため、問題が発生する可能性があります。 本番環境では、ベータ版を使用しないでください。
- 利用者は、ベータ版製品を使用するために、その他の規約への同意が必要になる場合があります。
- ベータプログラムは、新機能や拡張機能への早期アクセスを希望し、フィードバックの提供に同意頂けるお客様を対象としております。ベータ版製品に関するご意見、ご質問、不具合の報告は、support@contrastsecurity.com までご連絡ください。

プライバシーとデータの収集

Contrast Security は、お客様がどのように弊社の製品を使用しているかを理解し、改善していくことが重要と考えております。この情報は、弊社が問題を解決し、不具合を修正するために役立っています。

情報は次の方法で収集されます。

- システム診断 (938ページ)
- .NET Framework および.NET Core エージェントのテレメトリ (223ページ)
- Ruby エージェントのテレメトリ (475ページ)
- Python エージェントのテレメトリ (414ページ)

収集されるデータは匿名で、アプリケーションのデータは含まれません。この情報は Contrast Security によって収集され、共有されることはありません。Contrast Security のプライバシーポリシーが適用されます。

お客様のプライバシーの保護は、弊社にとって重要です。弊社が機密データを収集していると思われる場合や、データの取り扱いが安全で無いか不適切であると思われる場合は、調査致しますので、security@contrastsecurity.com までメールにてご連絡ください。