# Red Hat Advanced Cluster Security for Kubernetes 4.0

## インストール

Red Hat Advanced Cluster Security for Kubernetes のインストール

Last Updated: 2024-04-02

# Red Hat Advanced Cluster Security for Kubernetes 4.0 インストール

Red Hat Advanced Cluster Security for Kubernetes のインストール

## 法律上の通知



## 概要

本書では、Operator、Helm チャート、または roxctl CLI を使用して Red Hat Advanced Cluster Security for Kubernetes をインストールする方法を説明します。

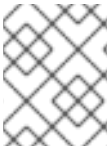
# 目次

# 第1章 サポートされているプラットフォームとインストール方法

Red Hat Advanced Cluster Security for Kubernetes (RHACS) は、OpenShift Container Platform および Kubernetes プラットフォームでサポートされています。サポートされているセルフマネージドおよびマネージドプラットフォームの詳細は、Red Hat Advanced Cluster Security for Kubernetes サポートポリシー を参照してください。

## 1.1. 各種プラットフォームのインストール方法

各種のプラットフォームで各種のインストールを実行できます。



**注記**

すべてのプラットフォームですべてのインストール方法がサポートされているわけではありません。

**表1.1 サポートされているプラットフォームと推奨されるインストール方法**

| プラットフォームタイプ[1] | プラットフォーム[2] | Central でサポート | セキュアなクラスターでサポート | サポート対象のインストール方法 | インストールの手順 |
|---|---|---|---|---|---|
| マネージドサービスプラットフォーム | Red Hat OpenShift Dedicated (OSD) | ○ | ○ | Operator (推奨)、Helm チャート、または **roxctl** CLI [3] | • Red Hat OpenShift への RHACS のインストールの概要 |
| | Azure Red Hat OpenShift (ARO) | ○ | ○ | | |
| | Red Hat OpenShift Service on AWS (ROSA) | ○ | ○ | | |
| | Amazon Elastic Kubernetes Service (Amazon EKS) | 制限付き [4] | ○ | Helm チャート (推奨)、または **roxctl** CLI [3] | • 他のプラットフォームへの RHACS のインストールの概要 |
| | Google Kubernetes Engine (Google GKE) | 制限付き [4] | ○ | | |

| プラットフォームタイプ[1] | プラットフォーム[2] | Central でサポート | セキュアなクラスターでサポート | サポート対象のインストール方法 | インストールの手順 |
|---|---|---|---|---|---|
| | Microsoft Azure Kubernetes Service (Microsoft AKS) | 制限付き[4] | ○ | | |
| セルフマネージドプラットフォーム | Red Hat OpenShift Container Platform (OCP) 4.x | ○ | ○ | Operator (推奨)、Helm チャート、または **roxctl** CLI[3] | • Red Hat OpenShift への RHACS のインストールの概要 |
| | Red Hat OpenShift Kubernetes Engine (OKE) 4.x | いいえ | ○ | | |

1. 各プラットフォームのサポートの可用性は、プラットフォームの包括的なライフサイクルとサポート終了日によって異なります。
2. サポートされているセルフマネージドおよびマネージドプラットフォームの詳細は、Red Hat Advanced Cluster Security for Kubernetes サポートポリシー を参照してください。
3. このインストール方法に従うための特別な要件がない限り、**roxctl** インストール方法を使用しないでください。
4. RHACS Central は OpenShift Container Platform 4 でのみテストおよび認定されており、完全にサポートされています。OpenShift Container Platform 4 ではない環境で Central をデプロイして使用することはできますが、サポートは RHACS 製品ソフトウェアのみに限定され、基盤となるインフラストラクチャープロバイダーには限定されません。問題の診断と切り分けの一環として、OpenShift Container Platform 4 環境で問題を再現する必要があります。問題が OpenShift Container Platform 4 以外のプロバイダーおよびクラスターに固有のものである場合、Red Hat は商業的に合理的なサポートを提供して問題を切り分けます。お客様は、それぞれのプロバイダーにケースを開くことが期待されています。手順は、Red Hat サードパーティーサポートポリシー を参照してください。

## 1.2. 異なるアーキテクチャーのインストール方法

Red Hat Advanced Cluster Security for Kubernetes (RHACS) は、次のアーキテクチャーをサポートします。

**表1.2 サポートされているアーキテクチャーと推奨されるインストール方法**

| サポート対象のアーキテクチャー | Central でサポート | セキュアなクラスターでサポート | サポート対象のインストール方法 |
|---|---|---|---|
| AMD64 | ○ | ○ | Operator (推奨)、Helm チャート、または **roxctl** CLI (非推奨) |
| ppc64le (IBM Power) | いいえ | はい (OpenShift Container Platform バージョン 4.12 以降) | Operator は、サポートされている唯一のインストール方法です。 |
| s390x (IBM zSystems および IBM® LinuxONE) | いいえ | はい (OpenShift Container Platform バージョン 4.10、4.12 以降) | |

## 1.3. RHACS でサポートされているブラウザー

Red Hat Advanced Cluster Security for Kubernetes (RHACS) ブラウザーのサポート は、Red Hat ポリシーに準拠しており、以下のブラウザーが含まれます。

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Microsoft Edge

# 第2章 RED HAT OPENSHIFT への RHACS のインストール

## 2.1. RED HAT OPENSHIFT への RHACS のインストールの概要

Red Hat Advanced Cluster Security for Kubernetes (RHACS) は、自己管理型の Red Hat OpenShift Kubernetes システムにセキュリティーサービスを提供します。

インストールする前に:

- インストールのプラットフォームと方法 を理解します。
- Red Hat Advanced Cluster Security for Kubernetes アーキテクチャー を理解します。
- 前提条件 を確認します。

次のリストは、インストール手順の概要を示しています。

1. Operator、Helm チャート、または **roxctl** CLI を使用して、クラスターに Central サービス をインストールします。
2. init バンドル を生成および適用します。
3. セキュアなクラスターのそれぞれに、セキュアなクラスターリソース をインストールします。

## 2.2. RED HAT OPENSHIFT での RHACS の前提条件

RHACS for OpenShift Container Platform または他の OCP 互換のサポートされる Kubernetes プラットフォームをインストールする前に、前提条件を満たしていることを確認してください。

### 2.2.1. 一般要件

RHACS には、インストールする前に満たす必要のあるシステム要件がいくつかあります。

**警告**

次の場所に Red Hat Cluster Security for Kubernetes をインストールしないでください。

- Amazon Elastic File System (Amazon EFS)。代わりに、デフォルトの **gp2** ボリュームタイプで Amazon Elastic Block Store (Amazon EBS) を使用してください。
- Streaming SIMD Extensions (SSE) 4.2 命令セットを備えていない古い CPU。たとえば、**Sandy Bridge** より古い Intel プロセッサー、および **Bulldozer** より古い AMD プロセッサー。(これらのプロセッサーは 2011 年にリリースされました。)

Red Hat Advanced Cluster Security for Kubernetes をインストールするには、次のものが必要です。

- OpenShift Container Platform バージョン 4.10 以降。サポートされているセルフマネージドおよびマネージド OpenShift Container Platform の詳細は、Red Hat Advanced Cluster Security for Kubernetes サポートポリシー を参照してください。
- サポートされているオペレーティングシステムを備えたクラスターノード。
  - Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL).
  - **プロセッサーとメモリー** :2 つの CPU コアと少なくとも 3GiB の RAM。

**注記**

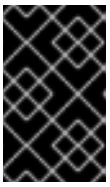
Central をデプロイするには、4 つ以上のコアを備えたマシンタイプを使用し、スケジューリングポリシーを適用して、そのようなノードで Central を起動します。

  - **アーキテクチャー**: AMD64、ppc64le、または s390x。

**注記**

ppc64le または s390x アーキテクチャーの場合、RHACS Secured クラスターサービスは IBM Power、IBM zSystems、および IBM® LinuxONE クラスターにのみインストールできます。現時点では、Central はサポートされていません。

- 永続ボリューム要求 (PVC) を使用した永続ストレージ。



**重要**

Red Hat Advanced Cluster Security for Kubernetes で Ceph FS ストレージを使用しないでください。Red Hat は、Red Hat Advanced Cluster Security for Kubernetes に RBD ブロックモード PVC を使用することをお勧めします。

  - 最高のパフォーマンスを得るには、ソリッドステートドライブ (SSD) を使用してください。ただし、SSD を使用できない場合は、別のタイプのストレージを使用できます。

Helm チャートを使用してインストールするには:

- Helm チャートを使用して Red Hat Advanced Cluster Security for Kubernetes をインストールまたは設定する場合は、Helm コマンドラインインターフェイス (CLI)v3.2 以降が必要です。**helm version** コマンドを使用して、インストールした Helm のバージョンを確認する。
- Red Hat OpenShift CLI (**oc**)。
- Red Hat Container Registry へのアクセスがあること。**registry.redhat.io** からイメージをダウンロードする方法は、Red Hat コンテナーレジストリーの認証 を参照してください。

### 2.2.2. Central をインストールするための前提条件

Central と呼ばれるコンテナー化されたサービスは API インタラクションとユーザーインターフェイス (ポータル) アクセスを処理し、Central DB (PostgreSQL 13) と呼ばれるコンテナー化されたサービスはデータの永続性を処理します。

Central と Central DB の両方に永続ストレージが必要です。

- 永続ボリュームクレーム (PVC) を使用してストレージを提供できます。

**注記**

hostPath ボリュームをストレージに使用できるのは、すべてのホスト (またはホストのグループ) が NFS 共有やストレージアプライアンスなどの共有ファイルシステムをマウントしている場合のみです。それ以外の場合、データは単一のノードにのみ保存されます。Red Hat は、hostPath ボリュームの使用を推奨していません。

- 最高のパフォーマンスを得るには、ソリッドステートドライブ (SSD) を使用してください。ただし、SSD を使用できない場合は、別のタイプのストレージを使用できます。

- Web プロキシーまたはファイアウォールを使用する場合は、**definitions.stackrox.io** ドメインと **collector-modules.stackrox.io** ドメインのトラフィックを許可するバイパスルールを設定し、Red Hat Advanced Cluster Security for Kubernetes が Web プロキシーまたはファイアウォールを信頼できるようにする必要があります。そうしないと、脆弱性定義とカーネルサポートパッケージの更新が失敗します。
  Red Hat Advanced Cluster Security for Kubernetes には、以下へのアクセスが必要です。

  - **definitions.stackrox.io では**、更新された脆弱性定義がダウンロードできます。脆弱性定義の更新により、Red Hat Advanced Cluster Security for Kubernetes は、新しい脆弱性が発見されたとき、または追加のデータソースが追加されたときに、最新の脆弱性データを維持できます。

  - 更新されたカーネルサポートパッケージをダウンロードするには、**collector-modules.stackrox.io** を使用します。更新されたカーネルサポートパッケージにより、Red Hat Advanced Cluster Security for Kubernetes は、最新のオペレーティングシステムをモニターし、コンテナー内で実行されているネットワークトラフィックとプロセスに関するデータを収集できます。これらの更新がないと、クラスターに新しいノードを追加したり、ノードのオペレーティングシステムを更新したりすると、Red Hat Advanced Cluster Security for Kubernetes がコンテナーのモニターに失敗する可能性があります。

**注記**

セキュリティー上の理由から、管理アクセスが制限されたクラスターに Central をデプロイする必要があります。

**メモリーとストレージの要件**
次の表に、Central のインストールと実行に必要な最小メモリーとストレージの値を示します。

| Central | CPU | メモリー | ストレージ |
|---|---|---|---|
| **要求** | 1.5 コア | 4 GiB | 100 GiB |
| **制限** | 4 コア | 8 GiB | 100 GiB |

| Central DB | CPU | メモリー | ストレージ |
|---|---|---|---|
| **要求** | 4 コア | 8 GiB | 100 GiB |

| Central DB | CPU | メモリー | ストレージ |
|---|---|---|---|
| **制限** | 8 コア | 16 GiB | 100 GiB |

**サイジングガイドライン**
クラスター内のノードの数に応じて、次のコンピュートリソースとストレージ値を使用します。

| ノード | デプロイメント | Central CPU | Central Memory | Central Storage |
|---|---|---|---|---|
| 最大 100 | 最大 1000 | 2 コア | 4 GiB | 100 GiB |
| 最大 500 | 最大 2000 | 4 コア | 8 GiB | 100 GiB |
| 500 以上 | 2000 以上 | 8 コア | 12 - 16 GiB | 100 - 200 GiB |

| ノード | デプロイメント | Central DB CPU | Central DB Memory | Central DB Storage |
|---|---|---|---|---|
| 最大 100 | 最大 1000 | 2 コア | 4 GiB | 100 GiB |
| 最大 500 | 最大 2000 | 4 コア | 8 GiB | 100 GiB |
| 500 以上 | 2000 以上 | 8 コア | 12 - 16 GiB | 100 - 200 GiB |

### 2.2.3. Scanner をインストールするための前提条件

Red Hat Advanced Cluster Security for Kubernetes には、Scanner と呼ばれるイメージ脆弱性 Scanner が含まれています。このサービスは、イメージレジストリーに統合されているスキャナーでスキャンされていないイメージをスキャンします。

**メモリーとストレージの要件**

| Scanner | CPU | Memory |
|---|---|---|
| **要求** | 1.2 コア | 2700 MiB |
| **制限** | 5 コア | 8000 MiB |

### 2.2.4. Sensor をインストールするための前提条件

Sensor は、Kubernetes および OpenShift Container Platform クラスターをモニターします。これらのサービスは現在、単一のデプロイメントでデプロイされ、Kubernetes API とのインタラクションを処理し、Collector と連携しています。

**メモリーとストレージの要件**

| Sensor | CPU | Memory |
|---|---|---|
| **要求** | 2 コア | 4 GiB |
| **制限** | 4 コア | 8 GiB |

### 2.2.5. Admission コントローラーをインストールするための前提条件

Admission Controller は、ユーザーが設定したポリシーに違反するワークロードを作成するのを防ぎます。

**メモリーとストレージの要件**
デフォルトでは、アドミッションコントロールサービスは 3 つのレプリカを実行します。次の表に、各レプリカのリクエストと制限を示します。

| 受付コントローラー | CPU | Memory |
|---|---|---|
| **要求** | .05 コア | 100 MiB |
| **制限** | .5 コア | 500 MiB |

### 2.2.6. Collector をインストールするための前提条件

Collector は、セキュアなクラスター内の各ノードのランタイムアクティビティーを監視します。Sensor に接続してこの情報をレポートします。

**注意**

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**メモリーとストレージの要件**

| Collector | CPU | Memory |
|---|---|---|
| **要求** | .05 コア | 320 MiB |
| **制限** | .75 コア | 1 GiB |

**注記**

Collector は変更可能なイメージタグ (**<version>-latest**) を使用するため、新しい Linux カーネルバージョンのサポートをより簡単に取得できます。コード、既存のカーネルモジュール、またはイメージ更新用の eBPF プログラムに変更はありません。更新では、最初のリリース後に公開された新しいカーネルバージョンをサポートする単一のイメージレイヤーのみが追加されます。

## 2.3. RED HAT OPENSHIFT での RHACS のセントラルサービスのインストール

Central は、RHACS アプリケーション管理インターフェイスとサービスを含むリソースです。データの永続性、API インタラクション、および RHACS ポータルアクセスを処理します。同じ Central インスタンスを使用して、複数の OpenShift Container Platform または Kubernetes クラスターをセキュリティー保護できます。

以下のいずれかの方法を使用して、OpenShift Container Platform または Kubernetes クラスターに Central をインストールできます。

- Operator を使用してインストールする
- Helm チャートを使用してインストールする
- **roxctl** CLI を使用してインストールします (この方法を使用する必要がある特定のインストールが必要でない限り、この方法は使用しないでください)。

### 2.3.1. Operator を使用して Central をインストールする

#### 2.3.1.1. Red Hat Advanced Cluster Security for Kubernetes Operator のインストール

OpenShift Container Platform に同梱される OperatorHub を使用するのが、Red Hat Advanced Cluster Security for Kubernetes をインストールする最も簡単な方法です。

**前提条件**

- Operator インストールパーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。
- OpenShift Container Platform 4.10 以降を使用する必要があります。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。

**手順**

1. Web コンソールで、**Operators → OperatorHub** ページに移動します。
2. Red Hat Advanced Cluster Security for Kubernetes が表示されない場合は、**Filter by keyword** ボックスに **Advanced Cluster Security** と入力して、Red Hat Advanced Cluster Security for Kubernetes Operator を検索します。
3. 詳細ページを表示するには、**Red Hat Advanced Cluster Security for Kubernetes Operator** を選択します。
4. Operator に関する情報を読み、**Install** をクリックします。
5. **Install Operator** ページで以下を行います。
   - **Installation mode** のデフォルト値を **All namespaces on the cluster**として保持します。
   - **Installed namespace** フィールドの Operator をインストールする特定の namespace を選択します。Red Hat Advanced Cluster Security for Kubernetes Operator を **rhacs-operator** namespace にインストールします。
   - **Update approval** には、自動更新または手動更新を選択します。

自動更新を選択した場合、Operator の新しいバージョンが利用可能になると、Operator Lifecycle Manager (OLM) は Operator の実行中のインスタンスを自動的にアップグレードします。

手動による更新を選択する場合は、新しいバージョンの Operator が利用可能になると、OLM は更新リクエストを作成します。クラスター管理者は、更新リクエストを手動で承認して、Operator を最新バージョンに更新する必要があります。

**重要**

手動更新を選択した場合、Central がインストールされているクラスターで RHACS Operator を更新するときに、すべてのセキュアなクラスターで RHACS Operator を更新する必要があります。セキュアなクラスターと、Central がインストールされているクラスターは、最適な機能を確保するために同じバージョンである必要があります。

6. **Install** をクリックします。

**検証**

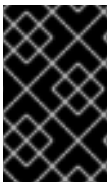
- インストールが完了したら、**Operators → Installed Operators** に移動して、Red Hat Advanced Cluster Security for Kubernetes Operator が **Succeeded** のステータスで一覧表示されていることを確認します。

**次の手順**

- **Central** カスタムリソースをインストール、設定、およびデプロイします。

### 2.3.1.2. Operator メソッドを使用した Central のインストール

Red Hat Advanced Cluster Security for Kubernetes の主要コンポーネントは Central と呼ばれます。**Central** カスタムリソースを使用して、OpenShift Container Platform に Central をインストールできます。Central は 1 回だけデプロイし、同じ Central インストールを使用して複数の個別のクラスターをモニターできます。



**重要**

Red Hat Advanced Cluster Security for Kubernetes を初めてインストールする場合、**SecuredCluster** カスタムリソースのインストールは Central が生成する証明書に依存するため、最初に **Central** カスタムリソースをインストールする必要があります。

**前提条件**

- OpenShift Container Platform 4.10 以降を使用する必要があります。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。

**手順**

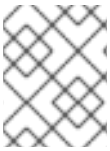
1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** ページに移動します。
2. インストールされている Operator のリストから、Red Hat Advanced Cluster Security for Kubernetes Operator を選択します。

3. 推奨される namespace に Operator をインストールした場合、OpenShift Container Platform はプロジェクトを **rhacs-operator** としてリストします。**Project: rhacs-operator** を選択し → **Create project** を選択します。

**警告**

- 別の namespace に Operator をインストールした場合、OpenShift Container Platform は **rhacs-operator** ではなくその namespace の名前を表示します。
- Red Hat Advanced Cluster Security for Kubernetes **Central** カスタムリソースは、**rhacs-operator** および **openshift-operator** プロジェクトではなく、独自のプロジェクト、または Red Hat Advanced Cluster Security for Kubernetes Operator をインストールしたプロジェクトにインストールする必要があります。

4. 新しいプロジェクト名 (たとえば、**stackrox**) を入力し、**Create** をクリックします。Red Hat では、プロジェクト名として **stackrox** を使用することをお勧めします。
5. **Provided APIs** セクションで、**Central** を選択します。**Create Central** をクリックします。
6. **Central** カスタムリソースの名前を入力し、適用するラベルを追加します。それ以外の場合は、使用可能なオプションのデフォルト値を受け入れます。
7. **Create** をクリックします。



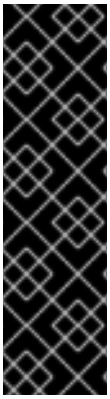
**注記**

クラスター全体のプロキシーを使用している場合、Red Hat Advanced Cluster Security for Kubernetes は、そのプロキシー設定を使用して外部サービスに接続します。

**次のステップ**

1. Central インストールを確認します。
2. オプション: Central オプションを設定します。
3. **Central** リソースと **SecuredCluster** リソース間の通信を可能にするクラスターシークレットを含む init バンドルを生成します。このバンドルをダウンロードし、それを使用してセキュリティー保護するクラスター上にリソースを生成し、安全に保存する必要があります。
4. モニターする各クラスターに、セキュアなクラスターサービスをインストールします。

### 2.3.1.3. Operator メソッドを使用した外部データベースを使用した Central のインストール

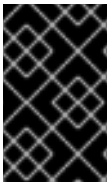


**重要**

外部 PostgreSQL サポートはテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

Red Hat Advanced Cluster Security for Kubernetes の主要コンポーネントは Central と呼ばれます。**Central** カスタムリソースを使用して、OpenShift Container Platform に Central をインストールできます。Central は 1 回だけデプロイし、同じ Central インストールを使用して複数の個別のクラスターをモニターできます。



**重要**

Red Hat Advanced Cluster Security for Kubernetes を初めてインストールする場合、**SecuredCluster** カスタムリソースのインストールは Central が生成する証明書に依存するため、最初に **Central** カスタムリソースをインストールする必要があります。

**前提条件**

- OpenShift Container Platform 4.10 以降を使用する必要があります。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。
- PostgreSQL 13 をサポートするデータベースをプロビジョニングしており、それを RHACS にのみ使用している。
- データベースを作成および削除する権限を持つスーパーユーザーのロールが必要です。

**注記**

RHACS 4.0 は、マルチテナントデータベースと PgBouncer をサポートしていません。

**手順**

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** ページに移動します。
2. インストールされている Operator のリストから、Red Hat Advanced Cluster Security for Kubernetes Operator を選択します。
3. 推奨される namespace に Operator をインストールした場合、OpenShift Container Platform はプロジェクトを **rhacs-operator** としてリストします。**Project: rhacs-operator** を選択し → **Create project** を選択します。

**警告**

- 別の namespace に Operator をインストールした場合、OpenShift Container Platform は **rhacs-operator** ではなくその namespace の名前を表示します。
- Red Hat Advanced Cluster Security for Kubernetes **Central** カスタムリソースは、**rhacs-operator** および **openshift-operator** プロジェクトではなく、独自のプロジェクト、または Red Hat Advanced Cluster Security for Kubernetes Operator をインストールしたプロジェクトにインストールする必要があります。

4. 新しいプロジェクト名 (たとえば、**stackrox**) を入力し、**Create** をクリックします。Red Hat では、プロジェクト名として **stackrox** を使用することをお勧めします。
5. OpenShift Container Platform Web コンソールまたはターミナルを使用して、デプロイされた namespace にパスワードシークレットを作成します。
   - OpenShift Container Platform Web コンソールで、**Workloads → Secrets** ページに移動します。キー **password**、およびプロビジョニングされたデータベースのスーパーユーザーのパスワードを含むプレーンテキストファイルのパスとしての値を使用して、**キー/値のシークレット** を作成します。
   - または、ターミナルで次のコマンドを実行します。

```
$ oc create secret generic external-db-password \ 1
  --from-file=password=<password.txt> 2
```

1. Kubernetes を使用する場合は、**oc** の代わりに **kubectl** を入力します。
2. **password.txt** をプレーンテキストのパスワードが含まれるファイルのパスに置き換えます。

6. OpenShift Container Platform Web コンソールの Red Hat Advanced Cluster Security for Kubernetes オペレーターページに戻ります。**Provided APIs** セクションで、**Central** を選択します。**Create Central** をクリックします。
7. **Central** カスタムリソースの名前を入力し、適用するラベルを追加します。
8. **Central Component Settings → Central DB Settings** に移動します。
9. **Administrator Password** には、参照されるシークレットを **external-db-password** (または以前に作成したパスワードのシークレット名) として指定します。
10. **Connection String (Technology Preview)** に、**keyword=value** 形式で接続文字列を指定します。たとえば、**host=<host> port=5432 user=postgres sslmode=verify-ca** です。
11. **Persistence → PersistentVolumeClaim → Claim Name** の場合は、**central-db** を削除します。

12. 必要に応じて、Central がデータベース証明書を信頼する認証局を指定できます。これを追加するには、次の例に示すように、YAML ビューに移動し、最上位仕様の下に TLS ブロックを追加します。

```
spec:
  tls:
    additionalCAs:
    - name: db-ca
      content: |
        <certificate>
```

13. **Create** をクリックします。

**注記**

クラスター全体のプロキシーを使用している場合、Red Hat Advanced Cluster Security for Kubernetes は、そのプロキシー設定を使用して外部サービスに接続します。

**次のステップ**

1. Central インストールを確認します。
2. オプション: Central オプションを設定します。
3. **Central** リソースと **SecuredCluster** リソース間の通信を可能にするクラスターシークレットを含む init バンドルを生成します。このバンドルをダウンロードし、それを使用してセキュリティー保護するクラスター上にリソースを生成し、安全に保存する必要があります。
4. モニターする各クラスターに、セキュアなクラスターサービスをインストールします。

**関連情報**

- Central 設定オプション
- PostgreSQL 接続文字列のドキュメント

### 2.3.1.4. Operator メソッドを使用した Central インストールの検証

Central のインストールが完了したら、RHACS ポータルにログインして、Central が正常にインストールされたことを確認します。

**手順**

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** ページに移動します。
2. インストールされている Operator のリストから、Red Hat Advanced Cluster Security for Kubernetes Operator を選択します。
3. **Central** タブを選択します。
4. **Centrals** リストから、**stackrox-central-services** を選択して詳細を表示します。
5. **admin** ユーザーのパスワードを取得するには、以下のいずれかを行います。

- **Admin Password Secret Reference** のリンクをクリックします。
- Red Hat OpenShift CLI を使用して、**Admin Credentials Info** にリストされているコマンドを入力します。

```
$ oc -n stackrox get secret central-htpasswd -o go-template='{{index .data "password" | base64decode}}'
```

6. Red Hat OpenShift CLI コマンドを使用して、RHACS ポータルへのリンクを見つけます。

```
$ oc -n stackrox get route central -o jsonpath="{.status.ingress[0].host}"
```

または、Red Hat Advanced Cluster Security for Kubernetes Web コンソールを使用して、次のコマンドを実行することにより、RHACS ポータルへのリンクを見つけることができます。

a. **Networking → Routes** に移動します。

b. **central** ルートを見つけて、**Location** 列の下にある RHACS ポータルリンクをクリックします。

7. ユーザー名 **admin** と、前の手順で取得したパスワードを使用して、RHACS ポータルにログインします。RHACS の設定が完了するまで (たとえば、**Central** リソースと少なくとも 1 つの **SecuredCluster** リソースをインストールして設定する)、ダッシュボードでデータを使用できません。**SecuredCluster** リソースは、**Central** リソースと同じクラスターにインストールおよび設定できます。**SecuredCluster** リソースを備えたクラスターは、Red Hat Advanced Cluster Management (RHACM) のマネージドクラスターに似ています。

**次のステップ**

1. オプション: central 設定を設定します。
2. **Central** リソースと **SecuredCluster** リソース間の通信を可能にするクラスターシークレットを含む init バンドルを生成します。このバンドルをダウンロードし、それを使用してセキュリティー保護するクラスター上にリソースを生成し、安全に保存する必要があります。
3. モニターする各クラスターに、セキュアなクラスターサービスをインストールします。

## 2.3.2. Helm チャートを使用して Central をインストールする

カスタマイズせずに Helm チャートを使用するか、デフォルト値を使用するか、設定パラメーターをさらにカスタマイズして Helm チャートを使用することにより、Central をインストールできます。

### 2.3.2.1. カスタマイズせずに Helm チャートを使用して Central をインストールする

カスタマイズせずにクラスターに RHACS をインストールできます。Helm チャートリポジトリーを追加し、**Central-Services** Helm チャートをインストールして、Central と Scanner の一元化されたコンポーネントをインストールする必要があります。

#### 2.3.2.1.1. Helm チャートリポジトリーの追加

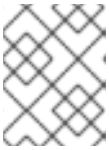
**手順**

- RHACS チャートリポジトリーを追加します。

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

Red Hat Advanced Cluster Security for Kubernetes の Helm リポジトリーには、異なるコンポーネントをインストールするための Helm チャートが含まれています。

- 集中型コンポーネント (Central および Scanner) をインストールするためのセントラルサービス Helm チャート (**central-services**)。



**注記**

一元化されたコンポーネントを 1 回だけデプロイし、同じインストールを使用して複数の個別のクラスターをモニターできます。

- クラスターごと (Sensor および Admission Controller) およびノードごと (Collector) のコンポーネントをインストールするための Secured Cluster Services Helm チャート (**secured-cluster-services**)。

**注記**

モニターする各クラスターにクラスターごとのコンポーネントをデプロイし、モニターするすべてのノードにノードごとのコンポーネントをデプロイします。

**検証**

- 次のコマンドを実行して、追加されたチャートリポジトリーを確認します。

```
$ helm search repo -l rhacs/
```

#### 2.3.2.1.2. カスタマイズせずにセントラルサービス Helm チャートをインストールする

次の手順を使用して、**Central-Services** Helm チャートをインストールし、集中型コンポーネント (Central および Scanner) をデプロイします。

**前提条件**

- Red Hat Container Registry へのアクセスがあること。**registry.redhat.io** からイメージをダウンロードする方法は、Red Hat コンテナーレジストリーの認証 を参照してください。

**手順**

- 次のコマンドを実行して Central services をインストールし、ルートを使用して Central を公開します。

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password> \
  --set central.exposure.route.enabled=true
```

- または、次のコマンドを実行して Central services をインストールし、ロードバランサーを使用して Central を公開します。

```
$ helm install -n stackrox \
```

```
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password> \
  --set central.exposure.loadBalancer.enabled=true
```

- または、次のコマンドを実行して Central services をインストールし、port forward を使用して Central を公開します。

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
   --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password>
```

**重要**

- 外部サービスに接続するためにプロキシーが必要なクラスターに Red Hat Cluster Security for Kubernetes をインストールする場合は、**proxyConfig** パラメーターを使用してプロキシー設定を指定する必要があります。以下に例を示します。

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

- インストール先の namespace に 1 つ以上のイメージプルシークレットをすでに作成している場合は、ユーザー名とパスワードを使用する代わりに、**--set imagePullSecrets.useExisting="<pull-secret-1;pull-secret-2>"** を使用できます。

- イメージプルシークレットは使用しないでください。

  - **quay.io/stackrox-io** または認証を必要としないプライベートネットワークのレジストリーからイメージを取得する場合。ユーザー名とパスワードを指定する代わりに、**--set imagePullSecrets.allowNone=true** を使用します。

  - インストールする namespace のデフォルトサービスアカウントでイメージプルシークレットをすでに設定している場合。ユーザー名とパスワードを指定する代わりに、**--set imagePullSecrets.useFromDefaultServiceAccount=true** を使用します。

インストールコマンドの出力は次のとおりです。

- 自動的に生成された管理者パスワード。
- すべての設定値を保存するための手順。
- Helm が生成する警告。

#### 2.3.2.2. カスタマイズされた Helm チャートを使用して Central をインストールする

helm **install** および **helm upgrade** コマンドで Helm チャート設定パラメーターを使用することにより、Red Hat OpenShift クラスターに RHACS をカスタマイズしてインストールできます。これらのパラメーターは、**--set** オプションを使用するか、YAML 設定ファイルを作成することで指定できます。

以下のファイルを作成して、Red Hat Advanced Cluster Security for Kubernetes をインストールするための Helm チャートを設定します。

- パブリック設定ファイル **values-public.yaml**: このファイルを使用して、機密性の低いすべての設定オプションを保存します。
- プライベート設定ファイル **values-private.yaml**: このファイルを使用して、機密性の高いすべての設定オプションを保存します。このファイルを安全に保管してください。

#### 2.3.2.2.1. プライベート設定ファイル

このセクションでは、**values-private.yaml** ファイルの設定可能なパラメーターをリストします。これらのパラメーターのデフォルト値はありません。

##### 2.3.2.2.1.1. イメージプルのシークレット

レジストリーからイメージをプルするために必要な認証情報は、以下の要素によって異なります。

- カスタムレジストリーを使用している場合、以下のパラメーターを指定する必要があります。
  - **imagePullSecrets.username**
  - **imagePullSecrets.password**
  - **image.registry**
- カスタムレジストリーへのログインにユーザー名とパスワードを使用しない場合は、以下のいずれかのパラメーターを指定する必要があります。
  - **imagePullSecrets.allowNone**
  - **imagePullSecrets.useExisting**
  - **imagePullSecrets.useFromDefaultServiceAccount**

| パラメーター | 説明 |
| --- | --- |
| **imagePullSecrets.username** | レジストリーへのログインに使用されるアカウントのユーザー名。 |
| **imagePullSecrets.password** | レジストリーへのログインに使用されるアカウントのパスワード |
| **imagePullSecrets.allowNone** | カスタムレジストリーを使用していて、クレデンシャルなしでイメージをプルできる場合は、**true** を使用します。 |

| パラメーター | 説明 |
| --- | --- |
| **imagePullSecrets.useExisting** | 値としてのシークレットのコンマ区切りリスト。たとえば、**secret1, secret2, secretN** です。ターゲット namespace に指定された名前で既存のイメージプルシークレットを既に作成している場合は、このオプションを使用します。 |
| **imagePullSecrets.useFromDefaultServiceAccount** | 十分なスコープのイメージプルシークレットを使用してターゲット namespace にデフォルトのサービスアカウントをすでに設定している場合は、**true** を使用します。 |

#### 2.3.2.2.1.2. プロキシー設定

外部サービスに接続するためにプロキシーが必要なクラスターに Red Hat Cluster Security for Kubernetes をインストールする場合は、**proxyConfig** パラメーターを使用してプロキシー設定を指定する必要があります。以下に例を示します。

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

| パラメーター | 説明 |
| --- | --- |
| **env.proxyConfig** | プロキシー設定。 |

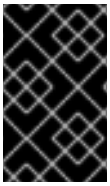
#### 2.3.2.2.1.3. Central

Central の設定可能なパラメーター。

新規インストールの場合、次のパラメーターをスキップできます。

- **central.jwtSigner.key**
- **central.serviceTLS.cert**
- **central.serviceTLS.key**
- **central.adminPassword.value**
- **central.adminPassword.htpasswd**
- **central.db.serviceTLS.cert**
- **central.db.serviceTLS.key**
- **central.db.password.value**

- これらのパラメーターの値を指定しない場合、Helm チャートはそれらの値を自動生成します。
- これらの値を変更する場合は、**helm upgrade** コマンドを使用し、**--set** オプションを使用して値を指定できます。

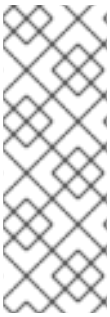


**重要**

管理者パスワードの設定には、**central.adminPassword.value** または **central.adminPassword.htpasswd** のいずれかのみを使用できますが、両方を使用することはできません。

| パラメーター | 説明 |
|---|---|
| **central.jwtSigner.key** | Red Hat Advanced Cluster Security for Kubernetes が認証用の JSON Web トークン (JWT) に署名するために使用する必要がある秘密鍵。 |
| **central.serviceTLS.cert** | セントラルサービスが Central をデプロイするために使用する必要がある内部証明書。 |
| **central.serviceTLS.key** | セントラルサービスが使用する必要がある内部証明書の秘密鍵。 |
| **central.defaultTLS.cert** | Central が使用する必要のあるユーザー向けの証明書。Red Hat Advanced Cluster Security for Kubernetes は、RHACS ポータルにこの証明書を使用します。<br>● 新規インストールの場合は、証明書を提供する必要があります。提供しない場合、Red Hat Advanced Cluster Security for Kubernetes は自己署名証明書を使用して Central をインストールします。<br>● アップグレードする場合、Red Hat Advanced Cluster Security for Kubernetes は既存の証明書とそのキーを使用します。 |
| **central.defaultTLS.key** | Central が使用する必要のあるユーザー向け証明書の秘密鍵。<br>● 新規インストールの場合は、秘密鍵を指定する必要があります。指定しない場合、Red Hat Advanced Cluster Security for Kubernetes は自己署名証明書を使用して Central をインストールします。<br>● アップグレードする場合、Red Hat Advanced Cluster Security for Kubernetes は既存の証明書とそのキーを使用します。 |
| **central.db.password.value** | Central DB の接続パスワード。 |

| パラメーター | 説明 |
| --- | --- |
| **central.adminPassword.value** | Red Hat Advanced Cluster Security for Kubernetes にログインするための管理者パスワード。 |
| **central.adminPassword.htpasswd** | Red Hat Advanced Cluster Security for Kubernetes にログインするための管理者パスワード。このパスワードは、bcrypt を使用してハッシュ形式で保存されます。 |
| **central.db.serviceTLS.cert** | Central DB サービスが Central DB をデプロイするために使用する内部証明書。 |
| **central.db.serviceTLS.key** | Central DB サービスが使用する内部証明書の秘密キー。 |
| **central.db.password.value** | Central DB への接続に使用されるパスワード。 |



**注記**

**Central.adminPassword.htpasswd** パラメーターを使用している場合は、bcrypt でエンコードされたパスワードハッシュを使用する必要があります。コマンド **htpasswd -nB admin** を実行して、パスワードハッシュを生成できます。以下に例を示します。

```
htpasswd: |
  admin:<bcrypt-hash>
```

2.3.2.2.1.4. Scanner

Scanner の設定可能なパラメーター。

新規インストールの場合、次のパラメーターをスキップでき、Helm チャートがそれらの値を自動生成します。それ以外の場合、新しいバージョンにアップグレードする場合は、以下のパラメーターの値を指定してください。

- **scanner.dbPassword.value**
- **scanner.serviceTLS.cert**
- **scanner.serviceTLS.key**
- **scanner.dbServiceTLS.cert**
- **scanner.dbServiceTLS.key**

| パラメーター | 説明 |
| --- | --- |

| パラメーター | 説明 |
| --- | --- |
| **scanner.dbPassword.value** | Scanner データベースでの認証に使用するパスワード。Red Hat Advanced Cluster Security for Kubernetes はその値を内部で自動的に作成して使用するため、このパラメーターは変更しないでください。 |
| **scanner.serviceTLS.cert** | Scanner サービスが Scanner のデプロイに使用する必要がある内部証明書。 |
| **scanner.serviceTLS.key** | Scanner サービスが使用する必要がある内部証明書の秘密鍵。 |
| **scanner.dbServiceTLS.cert** | Scanner-db サービスが Scanner データベースをデプロイするために使用する必要がある内部証明書。 |
| **scanner.dbServiceTLS.key** | Scanner-db サービスが使用する必要がある内部証明書の秘密鍵。 |

#### 2.3.2.2.2. パブリック設定ファイル

このセクションでは、**values-public.yaml** ファイルの設定可能なパラメーターをリストします。

##### 2.3.2.2.2.1. イメージプルのシークレット

イメージプルシークレットは、レジストリーからイメージをプルするために必要なクレデンシャルです。

| パラメーター | 説明 |
| --- | --- |
| **imagePullSecrets.allowNone** | カスタムレジストリーを使用していて、クレデンシャルなしでイメージをプルできる場合は、**true** を使用します。 |
| **imagePullSecrets.useExisting** | 値としてのシークレットのコンマ区切りリスト。たとえば、**secret1, secret2**。ターゲット namespace に指定された名前で既存のイメージプルシークレットを既に作成している場合は、このオプションを使用します。 |
| **imagePullSecrets.useFromDefaultServiceAccount** | 十分なスコープのイメージプルシークレットを使用してターゲット namespace にデフォルトのサービスアカウントをすでに設定している場合は、**true** を使用します。 |

##### 2.3.2.2.2.2. Image

は、ヘルムチャートが **central.image**、**scanner.image**、および **scanner-db.image** パラメーター

Image は、Helm チャートか **central.image**、**scanner.image**、および **scanner.dbimage** パラメーターのイメージを解決するために使用するメインレジストリーをセットアップするための設定を宣言します。

| パラメーター | 説明 |
| --- | --- |
| **image.registry** | イメージレジストリーのアドレス。**Registry.redhat.io** などのホスト名、または **us.gcr.io/stackrox-mirror** などのリモートレジストリーホスト名のいずれかを使用します。 |

#### 2.3.2.2.2.3. 環境変数

Red Hat Advanced Cluster Security for Kubernetes は、クラスター環境を自動的に検出し、**env.openshift**、**env.istio**、および **env.platform** の値を設定します。クラスター環境の自動検出をオーバーライドするには、これらの値のみを設定してください。

| パラメーター | 説明 |
| --- | --- |
| **env.openshift** | OpenShift Container Platform クラスターにインストールし、クラスター環境の自動検出をオーバーライドする場合は、**true** を使用します。 |
| **env.istio** | **true** を使用して、Istio が有効化されたクラスターにインストールし、クラスター環境の自動検出をオーバーライドします。 |
| **env.platform** | Red Hat Advanced Cluster Security for Kubernetes をインストールするプラットフォーム。その値を **default** または **gke** に設定して、クラスタープラットフォームを指定し、クラスター環境の自動検出をオーバーライドします。 |
| **env.offlineMode** | オフラインモードで Red Hat Advanced Cluster Security for Kubernetes を使用するには、**true** を使用します。 |

#### 2.3.2.2.2.4. 追加の信頼された認証局

Red Hat Advanced Cluster Security for Kubernetes は、信頼するシステムルート証明書を自動的に参照します。Central または Scanner が、組織内の機関またはグローバルに信頼されているパートナー組織によって発行された証明書を使用するサービスに到達する必要がある場合、次のパラメーターを使用して信頼するルート認証局を指定することにより、これらのサービスの信頼を追加できます。

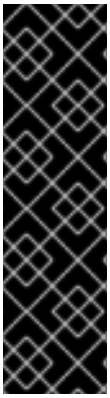
| パラメーター | 説明 |
| --- | --- |
| **additionalCAs.<certificate_name>** | 信頼するルート認証局の PEM エンコード証明書を指定します。 |

#### 2.3.2.2.2.5. Central

Central の設定可能なパラメーター。

- **hostPath** または **PersistentVolumeClaim** のいずれかとして永続ストレージオプションを指定する必要があります。
- 外部アクセス用の Central のデプロイメントを公開するため。1 つのパラメーター、**central.exposure.loadBalancer**、**central.exposure.nodePort**、または **central.exposure.route** のいずれかを指定する必要があります。これらのパラメーターに値を指定しない場合は、手動で Central を公開するか、ポート転送を使用して Central にアクセスする必要があります。

次の表には、外部 PostgreSQL データベース (テクノロジープレビュー) の設定が含まれています。



**重要**

外部 PostgreSQL サポートはテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

| パラメーター | 説明 |
| --- | --- |
| **central.endpointsConfig** | Central のエンドポイント設定オプションです。 |
| **central.nodeSelector** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Central の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Central の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.exposeMonitoring** | ポート番号 **9090** で Central の Prometheus メトリックエンドポイントを公開するには、**true** を指定します。 |
| **central.image.registry** | Central イメージのグローバル **image.registry** パラメーターをオーバーライドするカスタムレジストリーです。 |

| パラメーター | 説明 |
| --- | --- |
| **central.image.name** | デフォルトの Central イメージ名 (**main**) をオーバーライドするカスタムイメージ名。 |
| **central.image.tag** | Central イメージのデフォルトタグをオーバーライドするカスタムイメージタグです。新規インストール時に独自のイメージタグを指定した場合は、**helm upgrade** コマンドを実行して新しいバージョンにアップグレードするときに、このタグを手動でインクリメントする必要があります。独自のレジストリーで Central イメージをミラーリングする場合は、元のイメージタグを変更しないでください。 |
| **central.image.fullRef** | Central イメージのレジストリーアドレス、イメージ名、およびイメージタグを含む完全なリファレンスです。このパラメーターの値を設定すると、**central.image.registry**、**central.image.name**、および **central.image.tag** パラメーターがオーバーライドされます。 |
| **central.resources.requests.memory** | Central がデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **central.resources.requests.cpu** | Central がデフォルト値をオーバーライドするための CPU リクエストです。 |
| **central.resources.limits.memory** | Central がデフォルト値をオーバーライドためのメモリー制限です。 |
| **central.resources.limits.cpu** | Central がデフォルト値をオーバーライドための CPU 制限です。 |
| **central.persistence.hostPath** | RHACS がデータベースボリュームを作成するノード上のパス。Red Hat はこのオプションの使用を推奨していません。 |
| **central.persistence.persistentVolumeClaim.claimName** | 使用している永続ボリューム要求 (PVC) の名前です。 |
| **central.persistence.persistentVolumeClaim.createClaim** | 新しい PVC を作成するには **true** を使用し、既存のクレームを使用するには **false** を使用します。 |
| **central.persistence.persistentVolumeClaim.size** | 指定された要求による管理対象の永続ボリュームのサイズ (GiB 単位) です。 |
| **central.exposure.loadBalancer.enabled** | ロードバランサーを使用して Central を公開するには、**true** を使用します。 |

| パラメーター | 説明 |
| --- | --- |
| **central.exposure.loadBalancer.port** | Central を公開するポート番号です。デフォルトのポート番号は 443 です。 |
| **central.exposure.nodePort.enabled** | **true** を使用して、ノードポートサービスを使用して Central を公開します。 |
| **central.exposure.nodePort.port** | Central を公開するポート番号です。このパラメーターをスキップすると、OpenShift Container Platform は自動的にポート番号を割り当てます。Red Hat では、ノードポートを使用して Red Hat Advanced Cluster Security for Kubernetes を公開する場合、ポート番号を指定しないことを推奨しています。 |
| **central.exposure.route.enabled** | ルートを使用して Central を公開するには、**true** を使用します。このパラメーターは、OpenShift Container Platform クラスターでのみ使用できます。 |
| **central.db.external** | (テクノロジープレビュー) Central DB をデプロイメントせず、外部データベースを使用することを指定するには、**true** を使用します。 |
| **central.db.source.connectionString** | (テクノロジープレビュー) Central がデータベースへの接続に使用する接続文字列。これは、**central.db.external** が true に設定されている場合にのみ使用されます。接続文字列は、PostgreSQL ドキュメントの追加リソースで説明されているように、キーワード/値の形式である必要があります。<br>● PostgreSQL 13 のみがサポートされています。<br>● PgBouncer を介した接続はサポートされていません。<br>● ユーザーは、データベースを作成および削除できるスーパーユーザーである必要があります。 |
| **central.db.source.minConns** | 確立されるデータベースへの接続の最小数。 |
| **central.db.source.maxConns** | 確立されるデータベースへの接続の最大数。 |
| **central.db.source.statementTimeoutMs** | 単一のクエリーまたはトランザクションがデータベースに対してアクティブにできるミリ秒数。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.postgresConfig** | PostgreSQL ドキュメントの「追加リソース」で説明されているように、Central DB に使用される postgresql.conf。 |
| **central.db.hbaConfig** | PostgreSQL ドキュメントの「追加リソース」で説明されているように、Central DB に使用される pg_hba.conf。 |
| **central.db.nodeSelector** | ノードセレクターのラベルを **label-key: label-value** として指定して、Central DB が指定されたラベルを持つノードのみをスケジュールするように強制します。 |
| **central.db.image.registry** | Central DB イメージのグローバル **image.registry** パラメーターをオーバーライドするカスタムレジストリー。 |
| **central.db.image.name** | デフォルトの Central DB イメージ名 (**central-db**) をオーバーライドするカスタムイメージ名。 |
| **central.db.image.tag** | Central DB イメージのデフォルトのタグをオーバーライドするカスタムイメージタグ。新規インストール時に独自のイメージタグを指定した場合は、**helm upgrade** コマンドを実行して新しいバージョンにアップグレードするときに、このタグを手動でインクリメントする必要があります。Central DB イメージを独自のレジストリーにミラーリングする場合は、元のイメージタグを変更しないでください。 |
| **central.db.image.fullRef** | Central DB イメージのレジストリーアドレス、イメージ名、イメージタグを含む完全なリファレンス。このパラメーターの値を設定すると、**central.db.image.registry**、**central.db.image.name**、および **central.db.image.tag** パラメーターがオーバーライドされます。 |
| **central.db.resources.requests.memory** | Central DB がデフォルト値をオーバーライドするためのメモリー要求。 |
| **central.db.resources.requests.cpu** | Central DB がデフォルト値をオーバーライドするための CPU 要求。 |
| **central.db.resources.limits.memory** | Central DB がデフォルト値をオーバーライドするためのメモリー制限。 |
| **central.db.resources.limits.cpu** | Central DB がデフォルト値をオーバーライドするための CPU 制限。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.persistence.hostPath** | RHACS がデータベースボリュームを作成するノード上のパス。Red Hat はこのオプションの使用を推奨していません。 |
| **central.db.persistence.persistentVolumeClaim.claimName** | 使用している永続ボリューム要求 (PVC) の名前です。 |
| **central.db.persistence.persistentVolumeClaim.createClaim** | **true** を使用して新しい永続ボリューム要求を作成するか、**false** を使用して既存の要求を使用します。 |
| **central.db.persistence.persistentVolumeClaim.size** | 指定された要求による管理対象の永続ボリュームのサイズ (GiB 単位) です。 |

#### 2.3.2.2.2.6. Scanner

Scanner の設定可能なパラメーター。

| パラメーター | 説明 |
| --- | --- |
| **scanner.disable** | Scanner を使用せずに Red Hat Advanced Cluster Security for Kubernetes をインストールする場合は **true** を使用します。**helm upgrade** コマンドで使用すると、Helm は既存の Scanner のデプロイメントを削除します。 |
| **scanner.exposeMonitoring** | **true** を指定すると、ポート番号 **9090** でスキャナーの Prometheus メトリックエンドポイントが公開されます。 |
| **scanner.replicas** | Scanner のデプロイメント用に作成するレプリカの数です。**Scanner.autoscaling** パラメーターと一緒に使用する場合、この値はレプリカの初期数を設定します。 |
| **scanner.logLevel** | Scanner のログレベルを設定します。Red Hat では、ログレベルのデフォルト値 (**INFO**) を変更しないことをお勧めしています。 |
| **scanner.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Scanner をスケジュールするように強制します。 |
| **scanner.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.autoscaling.disable** | **true** を使用した Scanner のデプロイメントの自動スケーリングを無効にします。自動スケーリングを無効にすると、**minReplicas** パラメーターと **maxReplicas** パラメーターは効果がありません。 |
| **scanner.autoscaling.minReplicas** | 自動スケーリングのレプリカの最小数です。 |
| **scanner.autoscaling.maxReplicas** | 自動スケーリングのレプリカの最大数です。 |
| **scanner.resources.requests.memory** | Scanner がデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **scanner.resources.requests.cpu** | Scanner がデフォルト値をオーバーライドするための CPU リクエストです。 |
| **scanner.resources.limits.memory** | Scanner がデフォルト値をオーバーライドするためのメモリー制限です。 |
| **scanner.resources.limits.cpu** | Scanner がデフォルト値をオーバーライドするための CPU 制限です。 |
| **scanner.dbResources.requests.memory** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **scanner.dbResources.requests.cpu** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするための CPU リクエストです。 |
| **scanner.dbResources.limits.memory** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするためのメモリー制限です。 |
| **scanner.dbResources.limits.cpu** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするための CPU 制限です。 |
| **scanner.image.registry** | Scanner イメージのカスタムレジストリーです。 |
| **scanner.image.name** | デフォルトの Scanner イメージ名 (**scanner**) をオーバーライドするカスタムイメージ名です。 |
| **scanner.dbImage.registry** | Scanner DB イメージのカスタムレジストリーです。 |
| **scanner.dbImage.name** | デフォルトの Scanner DB イメージ名 (**scanner-db**) をオーバーライドするカスタムイメージ名です。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.dbNodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Scanner DB が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |

#### 2.3.2.2.2.7. カスタマイズ

これらのパラメーターを使用して、Red Hat Advanced Cluster Security for Kubernetes が作成するすべてのオブジェクトの追加の属性を指定します。

| パラメーター | 説明 |
| --- | --- |
| **customize.labels** | すべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.annotations** | すべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.podLabels** | すべてのデプロイメントにアタッチするカスタムラベルです。 |
| **customize.podAnnotations** | すべてのデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.envVars** | すべてのオブジェクトのすべてのコンテナーのカスタム環境変数です。 |
| **customize.central.labels** | Central が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.central.annotations** | Central が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.central.podLabels** | すべての Central のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.central.podAnnotations** | すべての Central のデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.central.envVars** | すべての Central コンテナーのカスタム環境変数です。 |

| パラメーター | 説明 |
| --- | --- |
| **customize.scanner.labels** | Scanner が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.scanner.annotations** | Scanner が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.scanner.podLabels** | すべての Scanner のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.scanner.podAnnotations** | すべての Scanner のデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.scanner.envVars** | すべての Scanner コンテナーのカスタム環境変数です。 |
| **customize.scanner-db.labels** | Scanner DB が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.scanner-db.annotations** | Scanner DB が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.scanner-db.podLabels** | すべての Scanner DB のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.scanner-db.podAnnotations** | すべての Scanner DB のデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.scanner-db.envVars** | すべての Scanner DB コンテナーのカスタム環境変数です。 |

以下のように使用することもできます。

- すべてのオブジェクトのラベルとアノテーションを指定するための **customize.other.service/*.labels** および **customize.other.service/*.annotations** パラメーターです。
- または、特定のサービス名を指定します。たとえば、**customize.other.service/central-loadbalancer.labels** と **customize.other.service/central-loadbalancer.annotations** をパラメーターとして指定し、それらの値を設定します。

#### 2.3.2.2.2.8. 高度なカスタマイズ

**重要**

このセクションで指定されているパラメーターは、情報提供のみを目的としています。Red Hat は、namespace とリリース名が変更された Red Hat Advanced Cluster Security for Kubernetes インスタンスをサポートしていません。

| パラメーター | 説明 |
| --- | --- |
| **allowNonstandardNamespace** | **true** を使用して、Red Hat Advanced Cluster Security for Kubernetes をデフォルトの namespace **stackrox** 以外の namespace にデプロイします。 |
| **allowNonstandardReleaseName** | **true** を使用して、Red Hat Advanced Cluster Security for Kubernetes をデフォルトの **stackrox-central-services** 以外のリリース名でデプロイします。 |

#### 2.3.2.2.3. セントラルサービス Helm チャートのインストール

**values-public.yaml** ファイルと **values-private.yaml** ファイルを設定した後、**central-services** Helm チャートをインストールして、集中型コンポーネント (Central と Scanner) をデプロイします。

**手順**

- 以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> 1
```

1 **-f** オプションを使用して、YAML 設定ファイルのパスを指定します。

### 2.3.2.3. central-services Helm チャートをデプロイした後の設定オプションの変更

**central-services** Helm チャートをデプロイした後、任意の設定オプションに変更を加えることができます。

**手順**

1. **values-public.yaml** および **values-private.yaml** 設定ファイルを新しい値で更新します。
2. **helm upgrade** コマンドを実行し、**-f** オプションを使用して設定ファイルを指定します。

```
$ helm upgrade -n stackrox \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> \
  -f <path_to_values_private.yaml>
```

**注記**

**--set** または **--set-file** パラメーターを使用して設定値を指定することもできます。ただし、これらのオプションは保存されないため、変更を加えるたびにすべてのオプションを手動で再度指定する必要があります。

## 2.3.3. roxctl CLI を使用して Central をインストールする

**警告**

実稼働環境では、Red Hat は Operator または Helm チャートを使用して RHACS をインストールすることを推奨しています。この方法を使用する必要がある特定のインストールがない限り、**roxctl** のインストール手法を使用しないでください。

#### 2.3.3.1. roxctl CLI のインストール

Red Hat Advanced Cluster Security for Kubernetes をインストールするには、バイナリーをダウンロードして **roxctl** CLI をインストールする必要があります。**roxctl** は、Linux、Windows、または macOS にインストールできます。

##### 2.3.3.1.1. Linux への roxctl CLI のインストール

次の手順を使用して、Linux に **roxctl** CLI バイナリーをインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Linux/roxctl
```

2. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

3. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

##### 2.3.3.1.2. macOS への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを macOS にインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Darwin/roxctl
```

2. バイナリーからすべての拡張属性を削除します。

```
$ xattr -c roxctl
```

3. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

4. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 2.3.3.1.3. Windows への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを Windows にインストールできます。

**手順**

- **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Windows/roxctl.exe
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

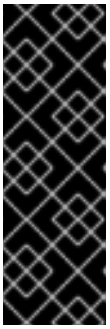
### 2.3.3.2. 対話型インストーラーの使用

対話型インストーラーを使用して、お使いの環境に必要なシークレット、デプロイメント設定、およびデプロイメントスクリプトを生成します。

**手順**

1. 対話型インストールコマンドを実行します。

```
$ roxctl central generate interactive
```



**重要**

**roxctl** CLI を使用して Red Hat Advanced Cluster Security for Kubernetes をインストールすると、下位互換性のためにデフォルトで PodSecurityPolicy (PSP) オブジェクトが作成されます。RHACS を Kubernetes バージョン 1.25 以降または OpenShift Container Platform バージョン 4.12 以降にインストールする場合、PSP オブジェクトの作成を無効にする必要があります。これを行うには、**roxctl central generate** コマンドと **roxctl sensor generate** コマンドで **--enable-pod-security-policies** オプションを **false** に指定します。

2. **Enter** を押してプロンプトのデフォルト値を受け入れるか、必要に応じてカスタム値を入力します。

```
Enter path to the backup bundle from which to restore keys and certificates (optional):
Enter PEM cert bundle file (optional): 1
Enter administrator password (default: autogenerated):
Enter orchestrator (k8s, openshift): openshift
Enter the directory to output the deployment bundle to (default: "central-bundle"):
Enter the OpenShift major version (3 or 4) to deploy on (default: "0"): 4
Enter Istio version when deploying into an Istio-enabled cluster (leave empty when not
running Istio) (optional):
Enter the method of exposing Central (route, lb, np, none) (default: "none"): route 2
Enter main image to use (default: "stackrox.io/main:3.0.61.1"):
Enter whether to run StackRox in offline mode, which avoids reaching out to the Internet
(default: "false"):
Enter whether to enable telemetry (default: "true"):
Enter the deployment tool to use (kubectl, helm, helm-values) (default: "kubectl"):
Enter Scanner DB image to use (default: "stackrox.io/scanner-db:2.15.2"):
Enter Scanner image to use (default: "stackrox.io/scanner:2.15.2"):
Enter Central volume type (hostpath, pvc): pvc 3
Enter external volume name (default: "stackrox-db"):
Enter external volume size in Gi (default: "100"):
Enter storage class name (optional if you have a default StorageClass configured):
```

1. カスタム TLS 証明書を追加する場合は、PEM でエンコードされた証明書のファイルパスを指定します。カスタム証明書を指定すると、対話型インストーラーは、使用しているカスタム証明書の PEM 秘密鍵を提供するように要求します。

2. RHACS ポータルを使用するには、ルート、ロードバランサー、またはノードポートを使用して Central を公開する必要があります。

3. hostPath ボリュームを使用して OpenShift Container Platform に Red Hat Cluster Security for Kubernetes をインストールする場合は、SELinux ポリシーを変更する必要があります。

**警告**

OpenShift Container Platform で、hostPath ボリュームを使用するには、SELinux ポリシーを変更して、ホストとコンテナーが共有するディレクトリーへのアクセスを許可する必要があります。これは、SELinux がデフォルトでディレクトリー共有をブロックしているためです。SELinux ポリシーを変更するには、次のコマンドを実行します。

```
$ sudo chcon -Rt svirt_sandbox_file_t <full_volume_path>
```

ただし、Red Hat は SELinux ポリシーの変更を推奨していません。代わりに、OpenShift Container Platform にインストールするときに PVC を使用してください。

完了すると、インストーラーは central-bundle という名前のフォルダーを作成します。このフォルダーには、Central をデプロイするために必要な YAML マニフェストとスクリプトが含まれています。さらに、信頼できる認証局である Central と Scanner をデプロイするために実行する必要があるスクリプトの画面上の説明と、RHACS ポータルにログインするための認証手順、プロンプトに答える際にパスワードを入力しなかった場合は自動生成されたパスワードも表示されます。

#### 2.3.3.3. Central インストールスクリプトの実行

対話型インストーラーを実行したら、**setup.sh** スクリプトを実行して Central をインストールできます。

**手順**

1. **setup.sh** スクリプトを実行して、イメージレジストリーアクセスを設定します。

```
$ ./central-bundle/central/scripts/setup.sh
```

2. 必要なリソースを作成します。

```
$ oc create -R -f central-bundle/central
```

3. デプロイメントの進行状況を確認します。

```
$ oc get pod -n stackrox -w
```

4. Central の実行後、RHACS ポータルの IP アドレスを見つけて、ブラウザーで開きます。プロンプトに応答するときに選択した公開方法に応じて、次のいずれかの方法を使用して IP アドレスを取得します。

| 公開方法 | コマンド | アドレス | 例 |
|---|---|---|---|
| ルート | **oc -n stackrox get route central** | 出力の **HOST/PORT** 列の下のアドレス | **https://central-stackrox.example.route** |

| 公開方法 | コマンド | アドレス | 例 |
|---|---|---|---|
| ノードポート | **oc get node -owide && oc -n stackrox get svc central-loadbalancer** | サービス用に表示されたポート上の任意のノードの IP またはホスト名 | **https://198.51.100.0:31489** |
| ロードバランサー | **oc -n stackrox get svc central-loadbalancer** | EXTERNAL-IP、またはポート 443 でサービスに表示されるホスト名 | **https://192.0.2.0** |
| なし | **central-bundle/central/scripts/port-forward.sh 8443** | **https://localhost:8443** | **https://localhost:8443** |

**注記**

対話型インストール中に自動生成されたパスワードを選択した場合は、次のコマンドを実行して、Central にログインするためのパスワードを確認できます。

```
$ cat central-bundle/password
```

## 2.4. オプション - OPERATOR を使用した RHACS の CENTRAL 設定オプションの設定

このトピックでは、Operator を使用して設定できる任意の設定オプションについて説明します。

### 2.4.1. Operator を使用した Central 設定オプション

Central インスタンスを作成すると、Operator は **Central** カスタムリソースの次の設定オプションを一覧表示します。

次の表には、外部 PostgreSQL データベース (テクノロジープレビュー) の設定が含まれています。

**重要**

外部 PostgreSQL サポートはテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

#### 2.4.1.1. Central 設定

| パラメーター | 説明 |
| --- | --- |
| **central.adminPasswordSecret** | **password** データ項目に管理者パスワードを含むシークレットを指定します。省略した場合、operator はパスワードを自動生成し、**central-htpasswd** シークレットの **password** 項目に保存します。 |
| **central.defaultTLSSecret** | デフォルトでは、Central は内部 TLS 証明書のみを提供します。つまり、入力レベルまたはロードバランサーレベルで TLS termination を処理する必要があります。Central で TLS を終了し、カスタムサーバー証明書を提供する場合は、証明書と秘密鍵を含むシークレットを指定できます。 |
| **central.adminPasswordGenerationDisabled** | 管理者パスワードの自動生成を無効にするには、このパラメーターを **true** に設定します。代替認証方法の初回設定を行った後のみこれを使用します。これを初期インストールに使用しないでください。それ以外の場合は、カスタムリソースを再インストールして再度ログインする必要があります。 |
| **central.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Central の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.exposure.loadBalancer.enabled** | ロードバランサーを介して Central を公開するには、これを **true** に設定します。 |
| **central.exposure.loadBalancer.port** | このパラメーターを使用して、ロードバランサーのカスタムポートを指定します。 |
| **central.exposure.loadBalancer.ip** | このパラメーターを使用して、ロードバランサー用に予約されている静的 IP アドレスを指定します。 |
| **central.exposure.route.enabled** | これを **true** に設定すると、OpenShift ルートを介して Central が公開されます。デフォルト値は **false** です。 |
| **central.exposure.route.host** | Central のルートに使用するカスタムホスト名を指定します。OpenShift Container Platform のデフォルト値を受け入れるには、これを未設定のままにします。 |
| **central.exposure.nodeport.enabled** | これを **true** に設定すると、ノードポートを介して Central が公開されます。デフォルト値は **false** です。 |
| **central.exposure.nodeport.port** | これを使用して、明示的なノードポートを指定します。 |
| **central.monitoring.exposeEndpoint** | Central の監視を有効にするには、**Enabled** を使用します。監視を有効にすると、RHACS はポート番号 **9090** に新しい監視サービスを作成します。デフォルト値は、**Disabled** です。 |
| **central.nodeSelector** | このコンポーネントを特定のノードでのみ実行する場合は、このパラメーターを使用してノードセレクターを設定できます。 |

| パラメーター | 説明 |
| --- | --- |
| **central.persistence.hostPath.path** | ホスト上のディレクトリーに永続データを保存するためのホストパスを指定します。Red Hat はこれの使用を推奨していません。ホストパスを使用する必要がある場合は、ノードセレクターで使用する必要があります。 |
| **central.persistence.persistentVolumeClaim.claimName** | 永続データを管理するための PVC の名前。指定された名前の PVC が存在しない場合は、作成されます。設定されていない場合、デフォルト値は **stackrox-db** です。データの損失を防ぐために、PVC は Central の削除によって自動的に削除されません。 |
| **central.persistence.persistentVolumeClaim.size** | クレームを通じて作成されたときの永続ボリュームのサイズ。これはデフォルトで自動的に生成されます。 |
| **central.persistence.persistentVolumeClaim.storageClassName** | PVC に使用するストレージクラスの名前。クラスターがデフォルトのストレージクラスで設定されていない場合は、このパラメーターの値を指定する必要があります。 |
| **central.resources.limits** | このパラメーターを使用して、Central のデフォルトのリソース制限をオーバーライドします。 |
| **central.resources.requests** | このパラメーターを使用して、Central のデフォルトのリソースリクエストをオーバーライドします。 |
| **central.imagePullSecrets** | このパラメーターを使用して、Central イメージのイメージプルシークレットを指定します。 |
| **central.db.passwordSecret.name** | **password** データ項目にデータベースパスワードを持つシークレットを指定します。このパラメーターは、接続文字列を手動で指定する場合にのみ使用します。省略した場合、Operator はパスワードを自動生成し、**central-db-password** シークレットの **password** 項目に保存します。 |
| **central.db.connectionString** | (テクノロジープレビュー): このパラメーターを設定すると Central DB はデプロイされず、Central は指定された接続文字列を使用して接続します。このパラメーターの値を指定する場合は、**central.db.passwordSecret.name** の値も指定する必要があります。このパラメーターには次の制約があります。<br>● 接続文字列は、PostgreSQL ドキュメントで説明されているキーワード/値形式である必要があります。詳細については、**追加リソース** セクションのリンクを参照してください。<br>● PostgreSQL 13 のみがサポートされています。<br>● PgBouncer を介した接続はサポートされていません。<br>● ユーザーは、データベースを作成および削除できるスーパーユーザーである必要があります。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.tolerations** | ノードセレクターが Taint されたノードを選択する場合、このパラメーターを使用して、Central DB の Taint Toleration キー、値、および効果を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.db.persistence.hostPath.path** | ホスト上のディレクトリーに永続データを保存するためのホストパスを指定します。Red Hat はこれの使用を推奨していません。ホストパスを使用する必要がある場合は、ノードセレクターで使用する必要があります。 |
| **central.db.persistence.persistentVolumeClaim.claimName** | 永続データを管理するための PVC の名前。指定された名前の PVC が存在しない場合は、作成されます。設定されていない場合、デフォルト値は **central-db** です。データ損失を防ぐため、PVC は Central DB の削除によって自動的に削除されません。 |
| **central.db.persistence.persistentVolumeClaim.size** | クレームを通じて作成されたときの永続ボリュームのサイズ。これはデフォルトで自動的に生成されます。 |
| **central.db.persistence.persistentVolumeClaim.storageClassName** | PVC に使用するストレージクラスの名前。クラスターがデフォルトのストレージクラスで設定されていない場合は、このパラメーターの値を指定する必要があります。 |
| **central.db.resources.limits** | このパラメーターを使用して、Central DB のデフォルトのリソース制限をオーバーライドします。 |
| **central.db.resources.requests** | このパラメーターを使用して、Central DB のデフォルトのリソース要求をオーバーライドします。 |

#### 2.4.1.2. Scanner 設定

| パラメーター | 説明 |
| --- | --- |
| **scanner.analyzer.nodeSelector** | この Scanner を特定のノードでのみ実行する場合は、このパラメーターを使用してノードセレクターを設定できます。 |
| **scanner.analyzer.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **scanner.analyzer.resources.limits** | このパラメーターを使用して、scanner のデフォルトのリソース制限をオーバーライドします。 |
| **scanner.analyzer.resources.requests** | このパラメーターを使用して、scanner のデフォルトのリソースリクエストをオーバーライドします。 |
| **scanner.analyzer.scaling.autoScaling** | 有効にすると、アナライザーレプリカの数は、指定された範囲内で、負荷に応じて動的に管理されます。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.analyzer.scaling.maxReplicas** | アナライザーの自動スケーリング設定で使用するレプリカの最大数を指定します |
| **scanner.analyzer.scaling.minReplicas** | アナライザーの自動スケーリング設定で使用する最低限のレプリカを指定します |
| **scanner.analyzer.scaling.replicas** | 自動スケーリングが無効になっている場合、レプリカの数は常にこの値に一致するように設定されます。 |
| **scanner.db.nodeSelector** | このコンポーネントを特定のノードでのみ実行する場合は、このパラメーターを使用してノードセレクターを設定できます。 |
| **scanner.db.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **scanner.db.resources.limits** | このパラメーターを使用して、scanner のデフォルトのリソース制限をオーバーライドします。 |
| **scanner.db.resources.requests** | このパラメーターを使用して、scanner のデフォルトのリソースリクエストをオーバーライドします。 |
| **scanner.monitoring.exposeEndpoint** | スキャナーの監視を有効にするには、**Enabled** を使用します。監視を有効にすると、RHACS はポート番号 **9090** に新しい監視サービスを作成します。デフォルト値は、**Disabled** です。 |
| **scanner.scannerComponent** | Scanner をデプロイしない場合は、このパラメーターを使用して Scanner を無効にできます。Scanner を無効にすると、このセクションの他のすべての設定は effect を持ちません。Red Hat は、Red Hat Advanced Cluster Security for Kubernetes Scanner を無効にすることを推奨していません。 |

#### 2.4.1.3. 一般およびその他の設定

| パラメーター | 説明 |
| --- | --- |
| **tls.additionalCAs** | セキュリティーで保護されたクラスターが信頼する追加の Trusted CA 証明書。これらの証明書は、通常、プライベート認証局を使用してサービスと統合するときに使用されます。 |
| **misc.createSCCs** | Central の **SecurityContextConstraints** (SCC) を作成するには、**true** を指定します。**true** に設定すると、一部の環境で問題が発生する可能性があります。 |
| **customize.annotations** | Central デプロイメントのカスタムアノテーションを指定できます。 |
| **customize.envVars** | 環境変数を設定するための詳細設定。 |

| パラメーター | 説明 |
| --- | --- |
| **egress.connectivityPolicy** | RHACS をオンラインモードまたはオフラインモードのどちらで実行するかを設定します。オフラインモードでは、脆弱性定義とカーネルモジュールの自動更新は無効になります。 |

**関連情報**

- 接続文字列 - PostgreSQL ドキュメント
- 設定ファイルを介したパラメーターのやりとり - PostgreSQL ドキュメント
- pg_hba.conf ファイル - PostgreSQL ドキュメント

## 2.5. RED HAT OPENSHIFT での RHACS の INIT バンドルの生成と適用

**SecuredCluster** リソースをクラスターにインストールする前に、init バンドルを作成する必要があります。**SecuredCluster** がインストールおよび設定されているクラスターは、このバンドルを使用して Central で認証します。RHACS ポータルまたは **roxctl** CLI を使用して、init バンドルを作成できます。次に、それを使用してリソースを作成することにより、init バンドルを適用します。

RHACS Cloud Service の init バンドルを設定するには、次のリソースを参照してください。

- セキュアなクラスター用の init バンドルの生成 (Red Hat Cloud)
- セキュアなクラスターへの init バンドルの適用 (Red Hat Cloud)
- Kubernetes のセキュアなクラスターの init バンドルの生成
- Kubernetes のセキュアなクラスターに init バンドルを適用する

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

### 2.5.1. init バンドルの生成

#### 2.5.1.1. RHACS ポータルを使用した init バンドルの生成

RHACS ポータルを使用して、シークレットを含む init バンドルを作成できます。

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

**手順**

1. 公開方法に基づいて RHACS ポータルのアドレスを見つけます。
   a. ルートの場合。

```
$ oc get route central -n stackrox
```

b. ロードバランサーの場合。

```
$ oc get service central-loadbalancer -n stackrox
```

c. port forward の場合:

i. 以下のコマンドを実行します。

```
$ oc port-forward svc/central 18443:443 -n stackrox
```

ii. **https://localhost:18443/** に移動します。

2. RHACS ポータルで、**Platform Configuration → Integrations** に移動します。

3. **Authentication Tokens** セクションに移動し、**Cluster Init Bundle** をクリックする。

4. **Generate bundle** をクリックする。

5. クラスター初期化バンドルの名前を入力し、**Generate** をクリックする。

a. Helm チャートを使用してインストールする場合は、**Download Helm Values File** をクリックして、生成されたバンドルをダウンロードします。

b. Operator を使用してインストールする場合は、**Download Kubernetes Secret File** をクリックして、生成されたバンドルをダウンロードします。

**重要**

このバンドルにはシークレットが含まれているため、セキュアに保管してください。同じバンドルを使用して、複数のセキュアなクラスターを作成できます。

**次のステップ**

1. セキュアなクラスターでリソースを作成して、init バンドルを適用します。

2. 各クラスターにセキュアなクラスターサービスをインストールします。

#### 2.5.1.2. roxctl CLI を使用した init バンドルの生成

**roxctl** CLI を使用して、シークレットを含む init バンドルを作成できます。

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

**前提条件**

**ROX_API_TOKEN** および **ROX_CENTRAL_ADDRESS** 環境変数が設定されている。

- **ROX_API_TOKEN** および **ROX_CENTRAL_ADDRESS** 環境変数を設定します。

```
$ export ROX_API_TOKEN=<api_token>
```

```
$ export ROX_CENTRAL_ADDRESS=<address>:<port_number>
```

**手順**

- 次のコマンドを実行して、シークレットを含むクラスター初期化バンドルを生成します。
  Helm のインストールの場合:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output cluster_init_bundle.yaml
```

  Operator のインストールの場合:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output-secrets cluster_init_bundle.yaml
```

**重要**

このバンドルにはシークレットが含まれているため、安全に保管してください。同じバンドルを使用して、複数のセキュアなクラスターを設定できます。

**次の手順**

- Red Hat OpenShift CLI を使用して、init バンドルを使用してリソースを作成します。

### 2.5.1.3. init バンドルを使用したリソースの作成

セキュアなクラスターをインストールする前に、init バンドルを使用して、セキュアなクラスター上のサービスが Central と通信できるようにする必要なリソースをクラスター上に作成する必要があります。

**注記**

Helm チャートを使用してインストールする場合は、この手順を実行しないでください。Helm を使用してインストールを完了します。関連情報セクションの 「Helm チャートを使用した保護されたクラスターへの RHACS のインストール」を参照してください。

**前提条件**

- シークレットを含む init バンドルを生成している必要があります。

**手順**

リソースを作成するには、次のいずれかの手順を実行します。

- OpenShift Container Platform Web コンソールのトップメニューで + をクリックし、**Import YAML** ページを開きます。init バンドルファイルをドラッグするか、その内容をコピーしてエディターに貼り付け、**Create** をクリックします。
- Red Hat OpenShift CLI を使用して、次のコマンドを実行してリソースを作成します。

```
$ oc create -f <init_bundle>.yaml \ 1
  -n <stackrox> 2
```

1 シークレットを含む init バンドルのファイル名を指定します。

2 Central サービスがインストールされているプロジェクトの名前を指定します。

**次の手順**

- 監視するすべてのクラスターに RHACS のセキュアなクラスターサービスをインストールします。

**関連情報**

- Helm チャートを使用したセキュアなクラスターへの RHACS のインストール

## 2.6. RED HAT OPENSHIFT での RHACS 用のセキュアなクラスターサービスのインストール

このセクションでは、セキュアなクラスターに Red Hat Advanced Cluster Security for Kubernetes をインストールするためのインストール手順について説明します。

次のいずれかの方法を使用して、セキュアなクラスターに RHACS をインストールできます。

- Operator を使用してインストールする
- Helm チャートを使用してインストールする
- **roxctl** CLI を使用してインストールします (この方法を使用する必要がある特定のインストールが必要でない限り、この方法は使用しないでください)。

### 2.6.1. Operator を使用したセキュアなクラスターへの RHACS のインストール

#### 2.6.1.1. セキュアなクラスターサービスのインストール

**SecuredCluster** カスタムリソースを使用して、セキュアなクラスターサービスをクラスターにインストールできます。モニターする環境内のすべてのクラスターに、セキュリティーでセキュアなクラスターサービスをインストールする必要があります。

**注意**

セキュアクラスターサービスをインストールすると、コレクターもインストールされます。Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**前提条件**

- OpenShift Container Platform を使用している場合は、バージョン 4.10 以降をインストールする必要があります。
- RHACS Operator をインストールしました。
- init バンドルを生成し、クラスターに適用しました。

**手順**

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** ページに移動します。
2. RHACS Operator をクリックします。
3. **Operator details** ページの central ナビゲーションメニューから **Secured Cluster** をクリックします。
4. **Create SecuredCluster** をクリックします。
5. **Configure via** フィールドで次のいずれかのオプションを選択します。
   - **Form view**: 画面上のフィールドを使用してセキュアなクラスターを設定し、他のフィールドを変更する必要がない場合は、このオプションを使用します。
   - **YAML view**: このビューを使用して、YAML ファイルを使用してセキュアなクラスターをセットアップします。YAML ファイルがウィンドウに表示され、その中のフィールドを編集できます。このオプションを選択した場合、ファイルの編集が終了したら、**Create** をクリックします。
6. **Form view** を使用している場合は、既定の名前を受け入れるか編集して、新しいプロジェクト名を入力します。デフォルト値は **stackrox-secured-cluster-services** です。
7. オプション: クラスターのラベルを追加します。
8. **SecuredCluster** カスタムリソースの一意の名前を入力します。
9. **Central Endpoint** には、Central インスタンスのアドレスとポート番号を入力します。たとえば、Central が **https://central.example.com** で利用できる場合は、central エンドポイントを **central.example.com:443** として指定します。デフォルト値の **central.stackrox.svc:443** は、セキュアなクラスターサービスと Central を同じクラスターにインストールした場合にのみ機能します。複数のクラスターを設定する場合は、デフォルト値を使用しないでください。代わりに、各クラスターの **Central Endpoint** 値を設定するときにホスト名を使用します。
   - セキュアなクラスターサービスと Central を同じクラスターにインストールする場合のみ、**central.stackrox.svc:443** を使用します。
10. デフォルト値を受け入れるか、必要に応じてカスタム値を設定します。たとえば、カスタム証明書または信頼されていない CA を使用している場合は、TLS を設定する必要がある場合があります。
11. **Create** をクリックします。

**次のステップ**

1. オプション: 追加のセキュアなクラスター設定を設定します。
2. インストールの検証

### 2.6.2. Helm チャートを使用したセキュアなクラスターへの RHACS のインストール

Helm チャートをカスタマイズせずに使用するか、デフォルト値を使用するか、設定パラメーターをカスタマイズして、セキュアなクラスターに RHACS をインストールできます。

#### 2.6.2.1. カスタマイズせずに Helm チャートを使用して、セキュアなクラスターに RHACS をインストールする

#### 2.6.2.1.1. Helm チャートリポジトリーの追加

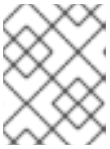
**手順**

- RHACS チャートリポジトリーを追加します。

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

Red Hat Advanced Cluster Security for Kubernetes の Helm リポジトリーには、異なるコンポーネントをインストールするための Helm チャートが含まれています。

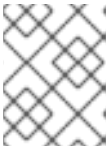
- 集中型コンポーネント (Central および Scanner) をインストールするためのセントラルサービス Helm チャート (**central-services**)。



**注記**

一元化されたコンポーネントを 1 回だけデプロイし、同じインストールを使用して複数の個別のクラスターをモニターできます。

- クラスターごと (Sensor および Admission Controller) およびノードごと (Collector) のコンポーネントをインストールするための Secured Cluster Services Helm チャート (**secured-cluster-services**)。



**注記**

モニターする各クラスターにクラスターごとのコンポーネントをデプロイし、モニターするすべてのノードにノードごとのコンポーネントをデプロイします。

**検証**

- 次のコマンドを実行して、追加されたチャートリポジトリーを確認します。

```
$ helm search repo -l rhacs/
```

#### 2.6.2.1.2. カスタマイズせずに secured-cluster-services Helm チャートをインストールする

次の手順を使用して、**secured-cluster-services** Helm チャートをインストールし、クラスターごとおよびノードごとのコンポーネント (Sensor、アドミッションコントローラー、および Collector) をデプロイします。

**注意**

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**前提条件**

- クラスターの RHACS init バンドルを生成しておく必要があります。
- Central service を公開するアドレスとポート番号が必要です。

**手順**

- Kubernetes ベースのクラスターで次のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <path_to_cluster_init_bundle.yaml> \ 1
  --set clusterName=<name_of_the_secured_cluster> \
  --set centralEndpoint=<endpoint_of_central_service> 2
```

1. **-f** オプションを使用して、init バンドルのパスを指定します。
2. Central のアドレスとポート番号を指定します。例: **acs.domain.com:443**

- OpenShift Container Platform クラスターで以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <path_to_cluster_init_bundle.yaml> \ 1
  --set clusterName=<name_of_the_secured_cluster> \
  --set centralEndpoint=<endpoint_of_central_service> 2
  --set scanner.disable=false
```

1. **-f** オプションを使用して、init バンドルのパスを指定します。
2. Central のアドレスとポート番号を指定します。例: **acs.domain.com:443**

**関連情報**

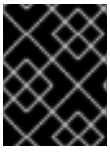
- Red Hat OpenShift での RHACS の init バンドルの生成と適用

### 2.6.2.2. カスタマイズによる secure-cluster-services Helm チャートの設定

このセクションでは、**helm install** および **helm upgrade** コマンドで使用できる Helm チャート設定パラメーターについて説明します。これらのパラメーターは、**--set** オプションを使用するか、YAML 設定ファイルを作成することで指定できます。

以下のファイルを作成して、Red Hat Advanced Cluster Security for Kubernetes をインストールするための Helm チャートを設定します。

- パブリック設定ファイル **values-public.yaml**: このファイルを使用して、機密性の低いすべての設定オプションを保存します。
- プライベート設定ファイル **values-private.yaml**: このファイルを使用して、機密性の高いすべての設定オプションを保存します。このファイルは安全に保管してください。



**重要**

**Download Helm Values File** Helm チャートを使用している間は、チャートの一部である **values.yaml** ファイルを変更しないでください。

#### 2.6.2.2.1. 設定パラメーター

| パラメーター | 説明 |
| --- | --- |
| **clusterName** | クラスターの名前です。 |
| **centralEndpoint** | Central エンドポイントのアドレス (ポート番号を含む)。gRPC に対応していないロードバランサーを使用している場合は、エンドポイントアドレスの前に **wss://** を付けて、WebSocket プロトコルを使用します。複数のクラスターを設定する場合は、アドレスにホスト名を使用します (例: **central.example.com:443**)。 |
| **sensor.endpoint** | ポート番号を含む Sensor エンドポイントのアドレスです。 |
| **sensor.imagePullPolicy** | Sensor コンテナーのイメージプルポリシーです。 |
| **sensor.serviceTLS.cert** | Sensor が使用する内部サービス間の TLS 証明書です。 |
| **sensor.serviceTLS.key** | Sensor が使用する内部サービス間 TLS 証明書キーです。 |
| **sensor.resources.requests.memory** | Sensor コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.requests.cpu** | Sensor コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.limits.memory** | Sensor コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.limits.cpu** | センサーコンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Sensor が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **sensor.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Sensor の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **image.main.name** | **main** イメージの名前です。 |

| パラメーター | 説明 |
| --- | --- |
| **image.collector.name** | Collector イメージの名前です。 |
| **image.main.registry** | main イメージに使用しているレジストリーのアドレスです。 |
| **image.collector.registry** | Collector イメージに使用しているレジストリーのアドレスです。 |
| **image.main.pullPolicy** | **main** イメージのイメージプルポリシーです。 |
| **image.collector.pullPolicy** | Collector イメージのイメージプルポリシーです。 |
| **image.main.tag** | 使用する **main** イメージのタグです。 |
| **image.collector.tag** | 使用する **collector** イメージのタグです。 |
| **collector.collectionMethod** | **EBPF**、**KERNEL_MODULE**、または **NO_COLLECTION** のいずれかです。 |
| **collector.imagePullPolicy** | Collector コンテナーのイメージプルポリシーです。 |
| **collector.complianceImagePullPolicy** | Compliance コンテナーのイメージプルポリシーです。 |
| **collector.disableTaintTolerations** | **false** を指定すると、許容値が Collector に適用され、Collector Pod は taint のあるすべてのノードにスケジュールできます。**true** として指定すると、許容値は適用されず、Collector Pod は taint のあるノードにスケジュールされません。 |
| **collector.resources.requests.memory** | Collector コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.requests.cpu** | Collector コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.limits.memory** | Collector コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.limits.cpu** | Collector コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **collector.complianceResources.requests.memory** | Compliance コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.requests.cpu** | Compliance の CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.limits.memory** | Compliance コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.limits.cpu** | Compliance コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.serviceTLS.cert** | Collector が使用する内部サービス間 TLS 証明書です。 |
| **collector.serviceTLS.key** | Collector が使用する内部サービス間 TLS 証明書キーです。 |
| **admissionControl.listenOnCreates** | この設定は、Kubernetes がワークロード作成イベントの **AdmissionReview** リクエストで Red Hat Advanced Cluster Security for Kubernetes に接続するように設定されているかどうかを制御します。 |
| **admissionControl.listenOnUpdates** | このパラメーターを **false** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、Kubernetes API サーバーがオブジェクト更新イベントを送信しないように **ValidatingWebhookConfiguration** を作成します。オブジェクトの更新ボリュームは通常、オブジェクトが作成するボリュームよりも多いため、これを **false** のままにしておくと、アドミッションコントロールサービスのロードが制限され、アドミッションコントロールサービスが誤動作する可能性が低くなります。 |
| **admissionControl.listenOnEvents** | この設定は、クラスターが Kubernetes **exec** および **portforward** イベントの **AdmissionReview** リクエストで Red Hat Advanced Cluster Security for Kubernetes に接続するように設定されているかどうかを制御します。Red Hat Advanced Cluster Security for Kubernetes は、OpenShift Container Platform 3.11 でこの機能をサポートしていません。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。 |

| パラメーター | 説明 |
| --- | --- |
| **admissionControl.dynamic.enforceOnCreates** | この設定は、Red Hat Advanced Cluster Security for Kubernetes がポリシーを評価するかどうかを制御します。無効にすると、すべての AdmissionReview リクエストが自動的に受け入れられます。 |
| **admissionControl.dynamic.enforceOnUpdates** | この設定は、アドミッションコントロールサービスの動作を制御します。これを機能させるには、**listenOnUpdates** を **true** として指定する必要があります。 |
| **admissionControl.dynamic.scanInline** | このオプションを **true** に設定すると、アドミッションコントロールサービスは、アドミッションデシジョンを行う前にイメージスキャンをリクエストします。イメージスキャンには数秒かかるため、このオプションを有効にするのは、クラスターで使用されるすべてのイメージがデプロイ前にスキャンされることを確認できる場合のみです (たとえば、イメージビルド中の CI 統合によって)。このオプションは、RHACS ポータルの **Contact image scanners** オプションに対応しています。 |
| **admissionControl.dynamic.disableBypass** | アドミッションコントローラーのバイパスを無効にするには、**true** に設定します。 |
| **admissionControl.dynamic.timeout** | アドミッションレビューリクエストを評価する間、Red Hat Advanced Cluster Security for Kubernetes が待機する最大時間 (秒単位) です。これを使用して、イメージスキャンを有効にするときにリクエストのタイムアウトを設定します。イメージスキャンが指定された時間より長く実行される場合、Red Hat Advanced Cluster Security for Kubernetes はリクエストを受け入れます。 |
| **admissionControl.resources.requests.memory** | Admission Control コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.requests.cpu** | Admission Control コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.limits.memory** | Admission Control コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.limits.cpu** | Admission Control コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **admissionControl.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Admission Control をスケジュールするように強制します。 |
| **admissionControl.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、アドミッションコントロールの taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **admissionControl.serviceTLS.cert** | Admission Control が使用する内部サービス間 TLS 証明書です。 |
| **admissionControl.serviceTLS.key** | Admission Control が使用する内部サービス間 TLS 証明書キーです。 |
| **registryOverride** | このパラメーターを使用して、デフォルトの **docker.io** レジストリーをオーバーライドします。他のレジストリーを使用している場合は、レジストリーの名前を指定してください。 |
| **collector.disableTaintTolerations** | **false** を指定すると、許容値が Collector に適用され、Collector Pod は taint のあるすべてのノードにスケジュールできます。**true** として指定した場合、許容値は適用されず、Collector Pod は taint のあるノードにスケジュールされません。 |
| **createUpgraderServiceAccount** | **true** を指定して、**sensor-upgrader** アカウントを作成します。デフォルトでは、Red Hat Advanced Cluster Security for Kubernetes は、セキュアなクラスターごとに **sensor-upgrader** と呼ばれるサービスアカウントを作成します。このアカウントは高い権限を持ちますが、アップグレードの時のみ使用されます。このアカウントを作成しない場合、Sensor に十分な権限がない場合は、将来のアップグレードを手動で完了する必要があります。 |
| **createSecrets** | **false** を指定すると、Sensor、Collector、および、アドミッションコントローラーのオーケストレーターシークレットの作成がスキップされます。 |

| パラメーター | 説明 |
| --- | --- |
| **collector.slimMode** | Collector のデプロイにスリムな Collector イメージを使用する場合は、**true** を指定します。slim Collector イメージを使用するには、一致する eBPF プローブまたはカーネルモジュールを提供する必要があります。Red Hat Advanced Cluster Security for Kubernetes をオフラインモードで実行している場合、スリム Collector が機能するには、stackrox.io からカーネルサポートパッケージをダウンロードして Central にアップロードする必要があります。それ以外の場合は、Central が https://collector-modules.stackrox.io/ でホストされているオンラインプローブリポジトリーにアクセスできることを確認する必要があります。 |
| **sensor.resources** | Sensor のリソース仕様です。 |
| **admissionControl.resources** | アドミッションコントローラーのリソース仕様です。 |
| **collector.resources** | Collector のリソース仕様です。 |
| **collector.complianceResources** | Collector の Compliance コンテナーのリソース仕様です。 |
| **exposeMonitoring** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、Sensor、Collector,、およびアドミッションコントローラーのポート番号 9090 で Prometheus メトリクスエンドポイントを公開します。 |
| **auditLogs.disableCollection** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、設定マップとシークレットへのアクセスと変更を検出するために使用される監査ログ検出機能を無効にします。 |
| **scanner.disable** | このオプションを **false** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、セキュアなクラスターに軽量な Scanner と Scanner DB をデプロイして、OpenShift Container Registry でイメージをスキャンできるようにします。Scanner の有効化は、OpenShift でのみサポートされます。デフォルト値は **true** です。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.replicas** | Collector の Compliance コンテナーのリソース仕様です。 |
| **scanner.logLevel** | このパラメーターを設定すると、Scanner のログレベルを変更できます。このオプションは、トラブルシューティングの目的でのみ使用してください。 |
| **scanner.autoscaling.disable** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は Scanner のデプロイメントでの自動スケーリングを無効にします。 |
| **scanner.autoscaling.minReplicas** | 自動スケーリングのレプリカの最小数です。デフォルトは 2 です。 |
| **scanner.autoscaling.maxReplicas** | 自動スケーリングのレプリカの最大数です。デフォルトは 5 です。 |
| **scanner.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Scanner をスケジュールするように強制します。 |
| **scanner.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner の taint toleration キー、値、および effect を指定します。 |
| **scanner.dbNodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Scanner DB が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。 |
| **scanner.resources.requests.memory** | Scanner コンテナーのメモリーリクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.resources.requests.cpu** | Scanner コンテナーの CPU リクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.resources.limits.memory** | Scanner コンテナーのメモリー制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.resources.limits.cpu** | Scanner コンテナーの CPU 制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.requests.memory** | Scanner DB コンテナーのメモリーリクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.requests.cpu** | Scanner DB コンテナーの CPU リクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.limits.memory** | Scanner DB コンテナーのメモリー制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.limits.cpu** | Scanner DB コンテナーの CPU 制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

##### 2.6.2.2.1.1. 環境変数

Sensor と、アドミッションコントローラーの環境変数は、次の形式で指定できます。

```
customize:
  envVars:
    ENV_VAR1: "value1"
    ENV_VAR2: "value2"
```

**customize** 設定を使用すると、この Helm チャートによって作成されたすべてのオブジェクトのカスタム Kubernetes メタデータ (ラベルとアノテーション) と、ワークロードの追加の Pod ラベル、Pod アノテーション、コンテナー環境変数を指定できます。

より一般的なスコープ (たとえば、すべてのオブジェクト) で定義されたメタデータを、より狭いスコープ (たとえば、Sensor デプロイメントのみ) で定義されたメタデータでオーバーライドできるという意味で、設定は階層的です。

#### 2.6.2.2.2. secure-cluster-services Helm チャートのインストール

**values-public.yaml** ファイルと **values-private.yaml** ファイルを設定した後、**secured-cluster-services** Helm チャートをインストールして、クラスターごと、およびノードごとのコンポーネント (Sensor、アドミッションコントローラー、Collector) をデプロイします。

**注意**

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**前提条件**

- クラスターの RHACS init バンドルを生成しておく必要があります。
- Central service を公開するアドレスとポート番号が必要です。

**手順**

- 以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <name_of_cluster_init_bundle.yaml> \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> (1)
```

(1) **-f** オプションを使用して、YAML 設定ファイルのパスを指定します。

**注記**

継続的インテグレーション (CI) システムを使用して **secured-cluster-services** Helm チャートをデプロイするには、init バンドル YAML ファイルを環境変数として **helm install** コマンドに渡します。

```
$ helm install ... -f <(echo "$INIT_BUNDLE_YAML_SECRET") (1)
```

(1) base64 でエンコードされた変数を使用している場合は、代わりに **helm install ... -f <(echo "$INIT_BUNDLE_YAML_SECRET" | base64 --decode)** コマンドを使用してください。

**関連情報**

- Red Hat OpenShift での RHACS の init バンドルの生成と適用

### 2.6.2.3. secure-cluster-services Helm チャートをデプロイした後の設定オプションの変更

**secure-cluster-services** Helm チャートをデプロイした後、任意の設定オプションに変更を加えることができます。

**手順**

1. **values-public.yaml** および **values-private.yaml** 設定ファイルを新しい値で更新します。
2. **helm upgrade** コマンドを実行し、**-f** オプションを使用して設定ファイルを指定します。

```
$ helm upgrade -n stackrox \
```

```
stackrox-secured-cluster-services rhacs/secured-cluster-services \
--reuse-values \ 1
-f <path_to_values_public.yaml> \
-f <path_to_values_private.yaml>
```

1 **--reuse-values** パラメーターを指定する必要があります。指定しない場合、Helm upgrade コマンドは以前に設定されたすべての設定をリセットします。

**注記**

**--set** または **--set-file** パラメーターを使用して設定値を指定することもできます。ただし、これらのオプションは保存されないため、変更を加えるたびにすべてのオプションを手動で再度指定する必要があります。

### 2.6.3. roxctl CLI を使用したセキュアなクラスターへの RHACS のインストール

CLI を使用してセキュアなクラスターに RHACS をインストールするには、次の手順を実行します。

1. **roxctl** CLI をインストールします。
2. Sensor を取り付けます。

#### 2.6.3.1. roxctl CLI のインストール

最初にバイナリーをダウンロードする必要があります。**roxctl** は、Linux、Windows、または macOS にインストールできます。

##### 2.6.3.1.1. Linux への roxctl CLI のインストール

次の手順を使用して、Linux に **roxctl** CLI バイナリーをインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Linux/roxctl
```

2. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

3. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 2.6.3.1.2. macOS への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを macOS にインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Darwin/roxctl
```

2. バイナリーからすべての拡張属性を削除します。

```
$ xattr -c roxctl
```

3. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

4. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 2.6.3.1.3. Windows への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを Windows にインストールできます。

**手順**

- **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Windows/roxctl.exe
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

### 2.6.3.2. Sensor のインストール

クラスターをモニターするには、Sensor をデプロイする必要があります。モニターする各クラスターに Sensor をデプロイする必要があります。次の手順では、RHACS ポータルを使用して Sensor を追加する方法について説明します。

**前提条件**

- Central サービスをすでにインストールしている必要があります。または、Red Hat Advanced Cluster Security Cloud Service (RHACS Cloud Service) で **ACS インスタンス** を選択して Central サービスにアクセスできます。

**手順**

1. セキュアなクラスターの RHACS ポータルで、**Platform Configuration → Clusters** に移動します。
2. **+NewCluster** を選択します。
3. クラスターの名前を指定します。
4. Sensor をデプロイする場所に基づいて、フィールドに適切な値を入力します。
   - 同じクラスターに Sensor をデプロイする場合は、すべてのフィールドのデフォルト値を受け入れます。
   - 別のクラスターにデプロイする場合は、**central.stackrox.svc:443** を、他のクラスターからアクセス可能なロードバランサー、ノードポート、またはポート番号を含む他のアドレスに置き換えます。
   - HAProxy、AWS Application Load Balancer (ALB)、AWS Elastic Load Balancing (ELB) などの非 gRPC 対応のロードバランサーを使用している場合は、WebSocket Secure (**wss**) プロトコルを使用してください。**wss** を使用するには:
     - アドレスの前に **wss://** を付けます。
     - アドレスの後にポート番号を追加します (例 **wss://stackrox-central.example.com:443**)。
5. **Next** をクリックして、Sensor のセットアップを続行します。
6. **Download YAML File and Keys**をクリックして、クラスターバンドル (zip アーカイブ) をダウンロードします。

**重要**

クラスターバンドルの zip アーカイブには、クラスターごとに固有の設定とキーが含まれています。同じファイルを別のクラスターで再利用しないでください。

7. モニター対象クラスターにアクセスできるシステムから、クラスターバンドルから **sensor** スクリプトを解凍して実行します。

```
$ unzip -d sensor sensor-<cluster_name>.zip
```

```
$ ./sensor/sensor.sh
```

Sensor をデプロイするために必要な権限がないという警告が表示された場合は、画面の指示に従うか、クラスター管理者に連絡して支援を求めてください。

Sensor はデプロイされた後、Central に接続し、クラスター情報を提供します。

**検証**

1. RHACS ポータルに戻り、デプロイメントが成功したかどうかを確認します。成功した場合、**Platform Configuration → Clusters** でクラスターのリストを表示すると、クラスターのステータスに緑色のチェックマークと **Healthy** ステータスが表示されます。緑色のチェックマークが表示されない場合は、次のコマンドを使用して問題を確認してください。
   - OpenShift Container Platform で、次のコマンドを入力します。

     ```
     $ oc get pod -n stackrox -w
     ```

   - Kubernetes で、次のコマンドを入力します。

     ```
     $ kubectl get pod -n stackrox -w
     ```

2. **Finish** をクリックしてウィンドウを閉じます。

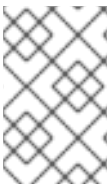
インストール後、Sensor はセキュリティー情報の RHACS へのレポートを開始し、RHACS ポータルダッシュボードは、Sensor をインストールしたクラスターからのデプロイメント、イメージ、およびポリシー違反を表示し始めます。

## 2.7. RED HAT OPENSHIFT での RHACS のインストールの確認

RHACS が正しくインストールされていることを確認する手順を示します。

### 2.7.1. インストールの検証

インストールが完了したら、いくつかの脆弱なアプリケーションを実行し、RHACS ポータルに移動して、セキュリティー評価とポリシー違反の結果を評価します。



**注記**

次のセクションにリストされているサンプルアプリケーションには重大な脆弱性が含まれており、Red Hat Advanced Cluster Security for Kubernetes のビルドおよびデプロイ時の評価機能を検証するように特別に設計されています。

インストールの検証

1. 公開方法に基づいて RHACS ポータルのアドレスを見つけます。
   a. ルートの場合。

      ```
      $ oc get route central -n stackrox
      ```

   b. ロードバランサーの場合。

      ```
      $ oc get service central-loadbalancer -n stackrox
      ```

   c. port forward の場合:
      i. 以下のコマンドを実行します。

         ```
         $ oc port-forward svc/central 18443:443 -n stackrox
         ```

ii. **https://localhost:18443/** に移動します。

2. Red Hat OpenShift CLI を使用して、新しいプロジェクトを作成します。

```
$ oc new-project test
```

3. 重大な脆弱性を持ついくつかのアプリケーションを開始します。

```
$ oc run shell --labels=app=shellshock,team=test-team \
  --image=vulnerables/cve-2014-6271 -n test
$ oc run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes は、これらのデプロイメントがクラスターに送信されるとすぐに、これらのデプロイメントを自動的にスキャンしてセキュリティーリスクとポリシー違反を検出します。RHACS ポータルに移動して、違反を表示します。デフォルトのユーザー名 **admin** と生成されたパスワードを使用して RHACS ポータルにログインできます。

# 第3章 他のプラットフォームへの RHACS のインストール

## 3.1. 他のプラットフォームへの RHACS のインストールの概要

Red Hat Advanced Cluster Security for Kubernetes (RHACS) は、Amazon Elastic Kubernetes Service (Amazon EKS)、Google Kubernetes Engine (Google GKE)、Microsoft Azure Kubernetes Service (Microsoft AKS) などのプラットフォームでセルフマネージド RHACS にセキュリティーサービスを提供します。

インストールする前に:

- インストールのプラットフォームと方法 を理解します。
- Red Hat Advanced Cluster Security for Kubernetes アーキテクチャー を理解します。
- 前提条件 を確認します。

次のリストは、インストール手順の概要を示しています。

1. Helm チャートまたは **roxctl** CLI を使用して、クラスターに Central サービス をインストールします。
2. init バンドル を生成および適用します。
3. セキュアなクラスターのそれぞれに、セキュアなクラスターリソース をインストールします。

## 3.2. 他のプラットフォームでの RHACS の前提条件

Amazon Elastic Kubernetes Service (Amazon EKS)、Google Kubernetes Engine (Google GKE)、Microsoft Azure Kubernetes Service (Microsoft AKS) などの他のプラットフォームに RHACS をインストールする前に、前提条件を満たしていることを確認してください。

### 3.2.1. 一般要件

RHACS には、インストールする前に満たす必要のあるシステム要件がいくつかあります。

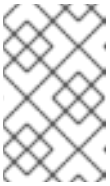
**警告**

次の場所に Red Hat Cluster Security for Kubernetes をインストールしないでください。

- Amazon Elastic File System (Amazon EFS)。代わりに、デフォルトの **gp2** ボリュームタイプで Amazon Elastic Block Store (Amazon EBS) を使用してください。
- Streaming SIMD Extensions (SSE) 4.2 命令セットを備えていない古い CPU。たとえば、**Sandy Bridge** より古い Intel プロセッサー、および **Bulldozer** より古い AMD プロセッサー。(これらのプロセッサーは 2011 年にリリースされました。)

Red Hat Advanced Cluster Security for Kubernetes をインストールするには、次のものが必要です。

- OpenShift Container Platform バージョン 4.10 以降。サポートされているセルフマネージドおよびマネージド OpenShift Container Platform の詳細は、Red Hat Advanced Cluster Security for Kubernetes サポートポリシー を参照してください。
- サポートされているオペレーティングシステムを備えたクラスターノード。
  - Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL).
- サポートされているマネージド Kubernetes プラットフォーム。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。
- サポートされているオペレーティングシステムを備えたクラスターノード。
  - **オペレーティングシステム**: Amazon Linux、CentOS、Google の Container-Optimized OS、Red Hat Enterprise Linux CoreOS (RHCOS)、Debian、Red Hat Enterprise Linux (RHEL)、または Ubuntu。
  - **プロセッサーとメモリー** :2 つの CPU コアと少なくとも 3GiB の RAM。

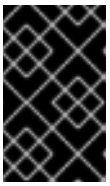
    **注記**

    Central をデプロイするには、4 つ以上のコアを備えたマシンタイプを使用し、スケジューリングポリシーを適用して、そのようなノードで Central を起動します。
  - **アーキテクチャー**: AMD64、ppc64le、または s390x。

    **注記**

    ppc64le または s390x アーキテクチャーの場合、RHACS Secured クラスターサービスは IBM Power、IBM zSystems、および IBM® LinuxONE クラスターにのみインストールできます。現時点では、Central はサポートされていません。
- 永続ボリューム要求 (PVC) を使用した永続ストレージ。

  **重要**

  Red Hat Advanced Cluster Security for Kubernetes で Ceph FS ストレージを使用しないでください。Red Hat は、Red Hat Advanced Cluster Security for Kubernetes に RBD ブロックモード PVC を使用することをお勧めします。
  - 最高のパフォーマンスを得るには、ソリッドステートドライブ (SSD) を使用してください。ただし、SSD を使用できない場合は、別のタイプのストレージを使用できます。

Helm チャートを使用してインストールするには:

- Helm チャートを使用して Red Hat Advanced Cluster Security for Kubernetes をインストールまたは設定する場合は、Helm コマンドラインインターフェイス (CLI)v3.2 以降が必要です。**helm version** コマンドを使用して、インストールした Helm のバージョンを確認する。
- Red Hat OpenShift CLI (**oc**)。

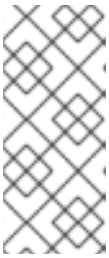
- Red Hat Container Registry へのアクセスがあること。**registry.redhat.io** からイメージをダウンロードする方法は、Red Hat コンテナーレジストリーの認証 を参照してください。

### 3.2.2. Central をインストールするための前提条件

Central と呼ばれるコンテナー化されたサービスは API インタラクションとユーザーインターフェイス (ポータル) アクセスを処理し、Central DB (PostgreSQL 13) と呼ばれるコンテナー化されたサービスはデータの永続性を処理します。

Central と Central DB の両方に永続ストレージが必要です。

- 永続ボリュームクレーム (PVC) を使用してストレージを提供できます。



**注記**

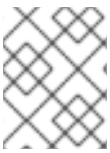
hostPath ボリュームをストレージに使用できるのは、すべてのホスト (またはホストのグループ) が NFS 共有やストレージアプライアンスなどの共有ファイルシステムをマウントしている場合のみです。それ以外の場合、データは単一のノードにのみ保存されます。Red Hat は、hostPath ボリュームの使用を推奨していません。

- 最高のパフォーマンスを得るには、ソリッドステートドライブ (SSD) を使用してください。ただし、SSD を使用できない場合は、別のタイプのストレージを使用できます。

- Web プロキシーまたはファイアウォールを使用する場合は、**definitions.stackrox.io** ドメインと **collector-modules.stackrox.io** ドメインのトラフィックを許可するバイパスルールを設定し、Red Hat Advanced Cluster Security for Kubernetes が Web プロキシーまたはファイアウォールを信頼できるようにする必要があります。そうしないと、脆弱性定義とカーネルサポートパッケージの更新が失敗します。
  Red Hat Advanced Cluster Security for Kubernetes には、以下へのアクセスが必要です。

  - **definitions.stackrox.io では**、更新された脆弱性定義がダウンロードできます。脆弱性定義の更新により、Red Hat Advanced Cluster Security for Kubernetes は、新しい脆弱性が発見されたとき、または追加のデータソースが追加されたときに、最新の脆弱性データを維持できます。

  - 更新されたカーネルサポートパッケージをダウンロードするには、**collector-modules.stackrox.io** を使用します。更新されたカーネルサポートパッケージにより、Red Hat Advanced Cluster Security for Kubernetes は、最新のオペレーティングシステムをモニターし、コンテナー内で実行されているネットワークトラフィックとプロセスに関するデータを収集できます。これらの更新がないと、クラスターに新しいノードを追加したり、ノードのオペレーティングシステムを更新したりすると、Red Hat Advanced Cluster Security for Kubernetes がコンテナーのモニターに失敗する可能性があります。



**注記**

セキュリティー上の理由から、管理アクセスが制限されたクラスターに Central をデプロイする必要があります。

**メモリーとストレージの要件**
次の表に、Central のインストールと実行に必要な最小メモリーとストレージの値を示します。

| Central | CPU | メモリー | ストレージ |
|---|---|---|---|
| **要求** | 1.5 コア | 4 GiB | 100 GiB |
| **制限** | 4 コア | 8 GiB | 100 GiB |

| Central DB | CPU | メモリー | ストレージ |
|---|---|---|---|
| **要求** | 4 コア | 8 GiB | 100 GiB |
| **制限** | 8 コア | 16 GiB | 100 GiB |

**サイジングガイドライン**
クラスター内のノードの数に応じて、次のコンピュートリソースとストレージ値を使用します。

| ノード | デプロイメント | Central CPU | Central Memory | Central Storage |
|---|---|---|---|---|
| 最大 100 | 最大 1000 | 2 コア | 4 GiB | 100 GiB |
| 最大 500 | 最大 2000 | 4 コア | 8 GiB | 100 GiB |
| 500 以上 | 2000 以上 | 8 コア | 12 - 16 GiB | 100 - 200 GiB |

| ノード | デプロイメント | Central DB CPU | Central DB Memory | Central DB Storage |
|---|---|---|---|---|
| 最大 100 | 最大 1000 | 2 コア | 4 GiB | 100 GiB |
| 最大 500 | 最大 2000 | 4 コア | 8 GiB | 100 GiB |
| 500 以上 | 2000 以上 | 8 コア | 12 - 16 GiB | 100 - 200 GiB |

### 3.2.3. Scanner をインストールするための前提条件

Red Hat Advanced Cluster Security for Kubernetes には、Scanner と呼ばれるイメージ脆弱性 Scanner が含まれています。このサービスは、イメージレジストリーに統合されているスキャナーでスキャンされていないイメージをスキャンします。

**メモリーとストレージの要件**

| Scanner | CPU | Memory |
|---|---|---|
| **要求** | 1.2 コア | 2700 MiB |
| **制限** | 5 コア | 8000 MiB |

| Scanner | CPU | Memory |
|---|---|---|

### 3.2.4. Sensor をインストールするための前提条件

Sensor は、Kubernetes および OpenShift Container Platform クラスターをモニターします。これらのサービスは現在、単一のデプロイメントでデプロイされ、Kubernetes API とのインタラクションを処理し、Collector と連携しています。

**メモリーとストレージの要件**

| Sensor | CPU | Memory |
|---|---|---|
| **要求** | 2 コア | 4 GiB |
| **制限** | 4 コア | 8 GiB |

### 3.2.5. Admission コントローラーをインストールするための前提条件

Admission Controller は、ユーザーが設定したポリシーに違反するワークロードを作成するのを防ぎます。

**メモリーとストレージの要件**
デフォルトでは、アドミッションコントロールサービスは 3 つのレプリカを実行します。次の表に、各レプリカのリクエストと制限を示します。

| 受付コントローラー | CPU | Memory |
|---|---|---|
| **要求** | .05 コア | 100 MiB |
| **制限** | .5 コア | 500 MiB |

### 3.2.6. Collector をインストールするための前提条件

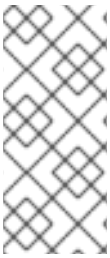
Collector は、セキュアなクラスター内の各ノードのランタイムアクティビティーを監視します。Sensor に接続してこの情報をレポートします。

#### 注意

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**メモリーとストレージの要件**

| Collector | CPU | Memory |
|---|---|---|
| **要求** | .05 コア | 320 MiB |
| **制限** | .75 コア | 1 GiB |



**注記**

Collector は変更可能なイメージタグ (**<version>-latest**) を使用するため、新しい Linux カーネルバージョンのサポートをより簡単に取得できます。コード、既存のカーネルモジュール、またはイメージ更新用の eBPF プログラムに変更はありません。更新では、最初のリリース後に公開された新しいカーネルバージョンをサポートする単一のイメージレイヤーのみが追加されます。

## 3.3. 他のプラットフォームでの RHACS のセントラルサービスのインストール

Central は、RHACS アプリケーション管理インターフェイスとサービスを含むリソースです。データの永続性、API インタラクション、および RHACS ポータルアクセスを処理します。同じ Central インスタンスを使用して、複数の OpenShift Container Platform または Kubernetes クラスターをセキュリティー保護できます。

次のいずれかの方法を使用して、Central をインストールできます。

- Helm チャートを使用してインストールする
- **roxctl** CLI を使用してインストールします (この方法を使用する必要がある特定のインストールが必要でない限り、この方法は使用しないでください)。

### 3.3.1. Helm チャートを使用して Central をインストールする

カスタマイズせずに Helm チャートを使用するか、デフォルト値を使用するか、設定パラメーターをさらにカスタマイズして Helm チャートを使用することにより、Central をインストールできます。

#### 3.3.1.1. カスタマイズせずに Helm チャートを使用して Central をインストールする

RHACS は、カスタマイズなしで Red Hat OpenShift クラスターにインストールできます。Helm チャートリポジトリーを追加し、**Central-Services** Helm チャートをインストールして、Central と Scanner の一元化されたコンポーネントをインストールする必要があります。

##### 3.3.1.1.1. Helm チャートリポジトリーの追加

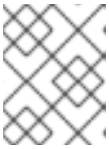
**手順**

- RHACS チャートリポジトリーを追加します。

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

Red Hat Advanced Cluster Security for Kubernetes の Helm リポジトリーには、異なるコンポーネントをインストールするための Helm チャートが含まれています。

- 集中型コンポーネント (Central および Scanner) をインストールするためのセントラルサービス Helm チャート (**central-services**)。



**注記**

一元化されたコンポーネントを 1 回だけデプロイし、同じインストールを使用して複数の個別のクラスターをモニターできます。

- クラスターごと (Sensor および Admission Controller) およびノードごと (Collector) のコンポーネントをインストールするための Secured Cluster Services Helm チャート (**secured-cluster-services**)。

**注記**

モニターする各クラスターにクラスターごとのコンポーネントをデプロイし、モニターするすべてのノードにノードごとのコンポーネントをデプロイします。

**検証**

- 次のコマンドを実行して、追加されたチャートリポジトリーを確認します。

```
$ helm search repo -l rhacs/
```

#### 3.3.1.1.2. カスタマイズせずにセントラルサービス Helm チャートをインストールする

次の手順を使用して、**Central-Services** Helm チャートをインストールし、集中型コンポーネント (Central および Scanner) をデプロイします。

**前提条件**

- Red Hat Container Registry へのアクセスがあること。**registry.redhat.io** からイメージをダウンロードする方法は、Red Hat コンテナーレジストリーの認証 を参照してください。

**手順**

- 次のコマンドを実行して Central services をインストールし、ルートを使用して Central を公開します。

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password> \
  --set central.exposure.route.enabled=true
```

- または、次のコマンドを実行して Central services をインストールし、ロードバランサーを使用して Central を公開します。

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password> \
  --set central.exposure.loadBalancer.enabled=true
```

- または、次のコマンドを実行して Central services をインストールし、port forward を使用して Central を公開します。

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
    --set imagePullSecrets.username=<username> \
  --set imagePullSecrets.password=<password>
```

**重要**

- 外部サービスに接続するためにプロキシーが必要なクラスターに Red Hat Cluster Security for Kubernetes をインストールする場合は、**proxyConfig** パラメーターを使用してプロキシー設定を指定する必要があります。以下に例を示します。

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

- インストール先の namespace に 1 つ以上のイメージプルシークレットをすでに作成している場合は、ユーザー名とパスワードを使用する代わりに、**--set imagePullSecrets.useExisting="<pull-secret-1;pull-secret-2>"** を使用できます。

- イメージプルシークレットは使用しないでください。

  - **quay.io/stackrox-io** または認証を必要としないプライベートネットワークのレジストリーからイメージを取得する場合。ユーザー名とパスワードを指定する代わりに、**--set imagePullSecrets.allowNone=true** を使用します。

  - インストールする namespace のデフォルトサービスアカウントでイメージプルシークレットをすでに設定している場合。ユーザー名とパスワードを指定する代わりに、**--set imagePullSecrets.useFromDefaultServiceAccount=true** を使用します。

インストールコマンドの出力は次のとおりです。

- 自動的に生成された管理者パスワード。
- すべての設定値を保存するための手順。
- Helm が生成する警告。

#### 3.3.1.2. カスタマイズされた Helm チャートを使用して Central をインストールする

helm **install** および **helm upgrade** コマンドで Helm チャート設定パラメーターを使用することにより、Red Hat OpenShift クラスターに RHACS をカスタマイズしてインストールできます。これらのパラメーターは、**--set** オプションを使用するか、YAML 設定ファイルを作成することで指定できます。

以下のファイルを作成して、Red Hat Advanced Cluster Security for Kubernetes をインストールするための Helm チャートを設定します。

- パブリック設定ファイル **values-public.yaml**: このファイルを使用して、機密性の低いすべての設定オプションを保存します。
- プライベート設定ファイル **values-private.yaml**: このファイルを使用して、機密性の高いすべての設定オプションを保存します。このファイルを安全に保管してください。

#### 3.3.1.2.1. プライベート設定ファイル

このセクションでは、**values-private.yaml** ファイルの設定可能なパラメーターをリストします。これらのパラメーターのデフォルト値はありません。

##### 3.3.1.2.1.1. イメージプルのシークレット

レジストリーからイメージをプルするために必要な認証情報は、以下の要素によって異なります。

- カスタムレジストリーを使用している場合、以下のパラメーターを指定する必要があります。
  - **imagePullSecrets.username**
  - **imagePullSecrets.password**
  - **image.registry**
- カスタムレジストリーへのログインにユーザー名とパスワードを使用しない場合は、以下のいずれかのパラメーターを指定する必要があります。
  - **imagePullSecrets.allowNone**
  - **imagePullSecrets.useExisting**
  - **imagePullSecrets.useFromDefaultServiceAccount**

| パラメーター | 説明 |
| --- | --- |
| **imagePullSecrets.username** | レジストリーへのログインに使用されるアカウントのユーザー名。 |
| **imagePullSecrets.password** | レジストリーへのログインに使用されるアカウントのパスワード |
| **imagePullSecrets.allowNone** | カスタムレジストリーを使用していて、クレデンシャルなしでイメージをプルできる場合は、**true** を使用します。 |
| **imagePullSecrets.useExisting** | 値としてのシークレットのコンマ区切りリスト。たとえば、**secret1, secret2, secretN** です。ターゲット namespace に指定された名前で既存のイメージプルシークレットを既に作成している場合は、このオプションを使用します。 |
| **imagePullSecrets.useFromDefaultServiceAccount** | 十分なスコープのイメージプルシークレットを使用してターゲット namespace にデフォルトのサービスアカウントをすでに設定している場合は、**true** を使用します。 |

#### 3.3.1.2.1.2. プロキシー設定

外部サービスに接続するためにプロキシーが必要なクラスターに Red Hat Cluster Security for Kubernetes をインストールする場合は、**proxyConfig** パラメーターを使用してプロキシー設定を指定する必要があります。以下に例を示します。

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

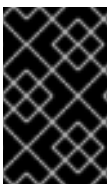
| パラメーター | 説明 |
| --- | --- |
| **env.proxyConfig** | プロキシー設定。 |

#### 3.3.1.2.1.3. Central

Central の設定可能なパラメーター。

新規インストールの場合、次のパラメーターをスキップできます。

- **central.jwtSigner.key**
- **central.serviceTLS.cert**
- **central.serviceTLS.key**
- **central.adminPassword.value**
- **central.adminPassword.htpasswd**
- **central.db.serviceTLS.cert**
- **central.db.serviceTLS.key**
- **central.db.password.value**
- これらのパラメーターの値を指定しない場合、Helm チャートはそれらの値を自動生成します。
- これらの値を変更する場合は、**helm upgrade** コマンドを使用し、**--set** オプションを使用して値を指定できます。



**重要**

管理者パスワードの設定には、**central.adminPassword.value** または **central.adminPassword.htpasswd** のいずれかのみを使用できますが、両方を使用することはできません。

| パラメーター | 説明 |
| --- | --- |
| **central.jwtSigner.key** | Red Hat Advanced Cluster Security for Kubernetes が認証用の JSON Web トークン (JWT) に署名するために使用する必要がある秘密鍵。 |
| **central.serviceTLS.cert** | セントラルサービスが Central をデプロイするために使用する必要がある内部証明書。 |
| **central.serviceTLS.key** | セントラルサービスが使用する必要がある内部証明書の秘密鍵。 |
| **central.defaultTLS.cert** | Central が使用する必要のあるユーザー向けの証明書。Red Hat Advanced Cluster Security for Kubernetes は、RHACS ポータルにこの証明書を使用します。<br>● 新規インストールの場合は、証明書を提供する必要があります。提供しない場合、Red Hat Advanced Cluster Security for Kubernetes は自己署名証明書を使用して Central をインストールします。<br>● アップグレードする場合、Red Hat Advanced Cluster Security for Kubernetes は既存の証明書とそのキーを使用します。 |
| **central.defaultTLS.key** | Central が使用する必要のあるユーザー向け証明書の秘密鍵。<br>● 新規インストールの場合は、秘密鍵を指定する必要があります。指定しない場合、Red Hat Advanced Cluster Security for Kubernetes は自己署名証明書を使用して Central をインストールします。<br>● アップグレードする場合、Red Hat Advanced Cluster Security for Kubernetes は既存の証明書とそのキーを使用します。 |
| **central.db.password.value** | Central DB の接続パスワード。 |
| **central.adminPassword.value** | Red Hat Advanced Cluster Security for Kubernetes にログインするための管理者パスワード。 |
| **central.adminPassword.htpasswd** | Red Hat Advanced Cluster Security for Kubernetes にログインするための管理者パスワード。このパスワードは、bcrypt を使用してハッシュ形式で保存されます。 |
| **central.db.serviceTLS.cert** | Central DB サービスが Central DB をデプロイするために使用する内部証明書。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.serviceTLS.key** | Central DB サービスが使用する内部証明書の秘密キー。 |
| **central.db.password.value** | Central DB への接続に使用されるパスワード。 |

注記

**Central.adminPassword.htpasswd** パラメーターを使用している場合は、bcrypt でエンコードされたパスワードハッシュを使用する必要があります。コマンド **htpasswd -nB admin** を実行して、パスワードハッシュを生成できます。以下に例を示します。

```
htpasswd: |
  admin:<bcrypt-hash>
```

3.3.1.2.1.4. Scanner

Scanner の設定可能なパラメーター。

新規インストールの場合、次のパラメーターをスキップでき、Helm チャートがそれらの値を自動生成します。それ以外の場合、新しいバージョンにアップグレードする場合は、以下のパラメーターの値を指定してください。

- **scanner.dbPassword.value**
- **scanner.serviceTLS.cert**
- **scanner.serviceTLS.key**
- **scanner.dbServiceTLS.cert**
- **scanner.dbServiceTLS.key**

| パラメーター | 説明 |
| --- | --- |
| **scanner.dbPassword.value** | Scanner データベースでの認証に使用するパスワード。Red Hat Advanced Cluster Security for Kubernetes はその値を内部で自動的に作成して使用するため、このパラメーターは変更しないでください。 |
| **scanner.serviceTLS.cert** | Scanner サービスが Scanner のデプロイに使用する必要がある内部証明書。 |
| **scanner.serviceTLS.key** | Scanner サービスが使用する必要がある内部証明書の秘密鍵。 |
| **scanner.dbServiceTLS.cert** | Scanner-db サービスが Scanner データベースをデプロイするために使用する必要がある内部証明書。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.dbServiceTLS.key** | Scanner-db サービスが使用する必要がある内部証明書の秘密鍵。 |

#### 3.3.1.2.2. パブリック設定ファイル

このセクションでは、**values-public.yaml** ファイルの設定可能なパラメーターをリストします。

##### 3.3.1.2.2.1. イメージプルのシークレット

イメージプルシークレットは、レジストリーからイメージをプルするために必要なクレデンシャルです。

| パラメーター | 説明 |
| --- | --- |
| **imagePullSecrets.allowNone** | カスタムレジストリーを使用していて、クレデンシャルなしでイメージをプルできる場合は、**true** を使用します。 |
| **imagePullSecrets.useExisting** | 値としてのシークレットのコンマ区切りリスト。たとえば、**secret1, secret2**。ターゲット namespace に指定された名前で既存のイメージプルシークレットを既に作成している場合は、このオプションを使用します。 |
| **imagePullSecrets.useFromDefaultServiceAccount** | 十分なスコープのイメージプルシークレットを使用してターゲット namespace にデフォルトのサービスアカウントをすでに設定している場合は、**true** を使用します。 |

##### 3.3.1.2.2.2. Image

Image は、Helm チャートが **central.image**、**scanner.image**、および **scanner.dbImage** パラメーターのイメージを解決するために使用するメインレジストリーをセットアップするための設定を宣言します。

| パラメーター | 説明 |
| --- | --- |
| **image.registry** | イメージレジストリーのアドレス。**Registry.redhat.io** などのホスト名、または **us.gcr.io/stackrox-mirror** などのリモートレジストリーホスト名のいずれかを使用します。 |

##### 3.3.1.2.2.3. 環境変数

Red Hat Advanced Cluster Security for Kubernetes は、クラスター環境を自動的に検出し、**env.openshift**、**env.istio**、および **env.platform** の値を設定します。クラスター環境の自動検出をオーバーライドするには、これらの値のみを設定してください。

| パラメーター | 説明 |
| --- | --- |
| **env.openshift** | OpenShift Container Platform クラスターにインストールし、クラスター環境の自動検出をオーバーライドする場合は、**true** を使用します。 |
| **env.istio** | **true** を使用して、Istio が有効化されたクラスターにインストールし、クラスター環境の自動検出をオーバーライドします。 |
| **env.platform** | Red Hat Advanced Cluster Security for Kubernetes をインストールするプラットフォーム。その値を **default** または **gke** に設定して、クラスタープラットフォームを指定し、クラスター環境の自動検出をオーバーライドします。 |
| **env.offlineMode** | オフラインモードで Red Hat Advanced Cluster Security for Kubernetes を使用するには、**true** を使用します。 |

#### 3.3.1.2.2.4. 追加の信頼された認証局

Red Hat Advanced Cluster Security for Kubernetes は、信頼するシステムルート証明書を自動的に参照します。Central または Scanner が、組織内の機関またはグローバルに信頼されているパートナー組織によって発行された証明書を使用するサービスに到達する必要がある場合、次のパラメーターを使用して信頼するルート認証局を指定することにより、これらのサービスの信頼を追加できます。

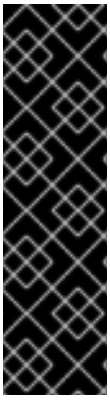
| パラメーター | 説明 |
| --- | --- |
| **additionalCAs.<certificate_name>** | 信頼するルート認証局の PEM エンコード証明書を指定します。 |

#### 3.3.1.2.2.5. Central

Central の設定可能なパラメーター。

- **hostPath** または **PersistentVolumeClaim** のいずれかとして永続ストレージオプションを指定する必要があります。
- 外部アクセス用の Central のデプロイメントを公開するため。1 つのパラメーター、**central.exposure.loadBalancer**、**central.exposure.nodePort**、または **central.exposure.route** のいずれかを指定する必要があります。これらのパラメーターに値を指定しない場合は、手動で Central を公開するか、ポート転送を使用して Central にアクセスする必要があります。

次の表には、外部 PostgreSQL データベース (テクノロジープレビュー) の設定が含まれています。



**重要**

外部 PostgreSQL サポートはテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

| パラメーター | 説明 |
| --- | --- |
| **central.endpointsConfig** | Central のエンドポイント設定オプションです。 |
| **central.nodeSelector** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Central の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Central の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **central.exposeMonitoring** | ポート番号 **9090** で Central の Prometheus メトリックエンドポイントを公開するには、**true** を指定します。 |
| **central.image.registry** | Central イメージのグローバル **image.registry** パラメーターをオーバーライドするカスタムレジストリーです。 |
| **central.image.name** | デフォルトの Central イメージ名 (**main**) をオーバーライドするカスタムイメージ名。 |
| **central.image.tag** | Central イメージのデフォルトタグをオーバーライドするカスタムイメージタグです。新規インストール時に独自のイメージタグを指定した場合は、**helm upgrade** コマンドを実行して新しいバージョンにアップグレードするときに、このタグを手動でインクリメントする必要があります。独自のレジストリーで Central イメージをミラーリングする場合は、元のイメージタグを変更しないでください。 |

| パラメーター | 説明 |
| --- | --- |
| **central.image.fullRef** | Central イメージのレジストリーアドレス、イメージ名、およびイメージタグを含む完全なリファレンスです。このパラメーターの値を設定すると、**central.image.registry**、**central.image.name**、および **central.image.tag** パラメーターがオーバーライドされます。 |
| **central.resources.requests.memory** | Central がデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **central.resources.requests.cpu** | Central がデフォルト値をオーバーライドするための CPU リクエストです。 |
| **central.resources.limits.memory** | Central がデフォルト値をオーバーライドためのメモリー制限です。 |
| **central.resources.limits.cpu** | Central がデフォルト値をオーバーライドための CPU 制限です。 |
| **central.persistence.hostPath** | RHACS がデータベースボリュームを作成するノード上のパス。Red Hat はこのオプションの使用を推奨していません。 |
| **central.persistence.persistentVolumeClaim.claimName** | 使用している永続ボリューム要求 (PVC) の名前です。 |
| **central.persistence.persistentVolumeClaim.createClaim** | 新しい PVC を作成するには **true** を使用し、既存のクレームを使用するには **false** を使用します。 |
| **central.persistence.persistentVolumeClaim.size** | 指定された要求による管理対象の永続ボリュームのサイズ (GiB 単位) です。 |
| **central.exposure.loadBalancer.enabled** | ロードバランサーを使用して Central を公開するには、**true** を使用します。 |
| **central.exposure.loadBalancer.port** | Central を公開するポート番号です。デフォルトのポート番号は 443 です。 |
| **central.exposure.nodePort.enabled** | **true** を使用して、ノードポートサービスを使用して Central を公開します。 |

| パラメーター | 説明 |
| --- | --- |
| **central.exposure.nodePort.port** | Central を公開するポート番号です。このパラメーターをスキップすると、OpenShift Container Platform は自動的にポート番号を割り当てます。Red Hat では、ノードポートを使用して Red Hat Advanced Cluster Security for Kubernetes を公開する場合、ポート番号を指定しないことを推奨しています。 |
| **central.exposure.route.enabled** | ルートを使用して Central を公開するには、**true** を使用します。このパラメーターは、OpenShift Container Platform クラスターでのみ使用できます。 |
| **central.db.external** | (テクノロジープレビュー) Central DB をデプロイメントせず、外部データベースを使用することを指定するには、**true** を使用します。 |
| **central.db.source.connectionString** | (テクノロジープレビュー) Central がデータベースへの接続に使用する接続文字列。これは、**central.db.external** が true に設定されている場合にのみ使用されます。接続文字列は、PostgreSQL ドキュメントの追加リソースで説明されているように、キーワード/値の形式である必要があります。<br>● PostgreSQL 13 のみがサポートされています。<br>● PgBouncer を介した接続はサポートされていません。<br>● ユーザーは、データベースを作成および削除できるスーパーユーザーである必要があります。 |
| **central.db.source.minConns** | 確立されるデータベースへの接続の最小数。 |
| **central.db.source.maxConns** | 確立されるデータベースへの接続の最大数。 |
| **central.db.source.statementTimeoutMs** | 単一のクエリーまたはトランザクションがデータベースに対してアクティブにできるミリ秒数。 |
| **central.db.postgresConfig** | PostgreSQL ドキュメントの「追加リソース」で説明されているように、Central DB に使用される postgresql.conf。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.hbaConfig** | PostgreSQL ドキュメントの「追加リソース」で説明されているように、Central DB に使用される pg_hba.conf。 |
| **central.db.nodeSelector** | ノードセレクターのラベルを **label-key: label-value** として指定して、Central DB が指定されたラベルを持つノードのみをスケジュールするように強制します。 |
| **central.db.image.registry** | Central DB イメージのグローバル **image.registry** パラメーターをオーバーライドするカスタムレジストリー。 |
| **central.db.image.name** | デフォルトの Central DB イメージ名 (**central-db**) をオーバーライドするカスタムイメージ名。 |
| **central.db.image.tag** | Central DB イメージのデフォルトのタグをオーバーライドするカスタムイメージタグ。新規インストール時に独自のイメージタグを指定した場合は、**helm upgrade** コマンドを実行して新しいバージョンにアップグレードするときに、このタグを手動でインクリメントする必要があります。Central DB イメージを独自のレジストリーにミラーリングする場合は、元のイメージタグを変更しないでください。 |
| **central.db.image.fullRef** | Central DB イメージのレジストリーアドレス、イメージ名、イメージタグを含む完全なリファレンス。このパラメーターの値を設定すると、**central.db.image.registry**、**central.db.image.name**、および **central.db.image.tag** パラメーターがオーバーライドされます。 |
| **central.db.resources.requests.memory** | Central DB がデフォルト値をオーバーライドするためのメモリー要求。 |
| **central.db.resources.requests.cpu** | Central DB がデフォルト値をオーバーライドするための CPU 要求。 |
| **central.db.resources.limits.memory** | Central DB がデフォルト値をオーバーライドするためのメモリー制限。 |
| **central.db.resources.limits.cpu** | Central DB がデフォルト値をオーバーライドするための CPU 制限。 |
| **central.db.persistence.hostPath** | RHACS がデータベースボリュームを作成するノード上のパス。Red Hat はこのオプションの使用を推奨していません。 |

| パラメーター | 説明 |
| --- | --- |
| **central.db.persistence.persistentVolumeClaim.claimName** | 使用している永続ボリューム要求 (PVC) の名前です。 |
| **central.db.persistence.persistentVolumeClaim.createClaim** | **true** を使用して新しい永続ボリューム要求を作成するか、**false** を使用して既存の要求を使用します。 |
| **central.db.persistence.persistentVolumeClaim.size** | 指定された要求による管理対象の永続ボリュームのサイズ (GiB 単位) です。 |

3.3.1.2.2.6. Scanner

Scanner の設定可能なパラメーター。

| パラメーター | 説明 |
| --- | --- |
| **scanner.disable** | Scanner を使用せずに Red Hat Advanced Cluster Security for Kubernetes をインストールする場合は **true** を使用します。**helm upgrade** コマンドで使用すると、Helm は既存の Scanner のデプロイメントを削除します。 |
| **scanner.exposeMonitoring** | **true** を指定すると、ポート番号 **9090** でスキャナーの Prometheus メトリックエンドポイントが公開されます。 |
| **scanner.replicas** | Scanner のデプロイメント用に作成するレプリカの数です。**Scanner.autoscaling** パラメーターと一緒に使用する場合、この値はレプリカの初期数を設定します。 |
| **scanner.logLevel** | Scanner のログレベルを設定します。Red Hat では、ログレベルのデフォルト値 (**INFO**) を変更しないことをお勧めしています。 |
| **scanner.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Scanner をスケジュールするように強制します。 |
| **scanner.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.autoscaling.disable** | **true** を使用した Scanner のデプロイメントの自動スケーリングを無効にします。自動スケーリングを無効にすると、**minReplicas** パラメーターと **maxReplicas** パラメーターは効果がありません。 |
| **scanner.autoscaling.minReplicas** | 自動スケーリングのレプリカの最小数です。 |
| **scanner.autoscaling.maxReplicas** | 自動スケーリングのレプリカの最大数です。 |
| **scanner.resources.requests.memory** | Scanner がデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **scanner.resources.requests.cpu** | Scanner がデフォルト値をオーバーライドするための CPU リクエストです。 |
| **scanner.resources.limits.memory** | Scanner がデフォルト値をオーバーライドするためのメモリー制限です。 |
| **scanner.resources.limits.cpu** | Scanner がデフォルト値をオーバーライドするための CPU 制限です。 |
| **scanner.dbResources.requests.memory** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするためのメモリーリクエストです。 |
| **scanner.dbResources.requests.cpu** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするための CPU リクエストです。 |
| **scanner.dbResources.limits.memory** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするためのメモリー制限です。 |
| **scanner.dbResources.limits.cpu** | Scanner データベースのデプロイメントがデフォルト値をオーバーライドするための CPU 制限です。 |
| **scanner.image.registry** | Scanner イメージのカスタムレジストリーです。 |
| **scanner.image.name** | デフォルトの Scanner イメージ名 (**scanner**) をオーバーライドするカスタムイメージ名です。 |
| **scanner.dbImage.registry** | Scanner DB イメージのカスタムレジストリーです。 |
| **scanner.dbImage.name** | デフォルトの Scanner DB イメージ名 (**scanner-db**) をオーバーライドするカスタムイメージ名です。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.dbNodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Scanner DB が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |

#### 3.3.1.2.2.7. カスタマイズ

これらのパラメーターを使用して、Red Hat Advanced Cluster Security for Kubernetes が作成するすべてのオブジェクトの追加の属性を指定します。

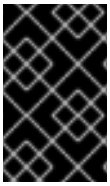
| パラメーター | 説明 |
| --- | --- |
| **customize.labels** | すべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.annotations** | すべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.podLabels** | すべてのデプロイメントにアタッチするカスタムラベルです。 |
| **customize.podAnnotations** | すべてのデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.envVars** | すべてのオブジェクトのすべてのコンテナーのカスタム環境変数です。 |
| **customize.central.labels** | Central が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.central.annotations** | Central が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.central.podLabels** | すべての Central のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.central.podAnnotations** | すべての Central のデプロイメントにアタッチするカスタムアノテーションです。 |

| パラメーター | 説明 |
| --- | --- |
| **customize.central.envVars** | すべての Central コンテナーのカスタム環境変数です。 |
| **customize.scanner.labels** | Scanner が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.scanner.annotations** | Scanner が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.scanner.podLabels** | すべての Scanner のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.scanner.podAnnotations** | すべての Scanner のデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.scanner.envVars** | すべての Scanner コンテナーのカスタム環境変数です。 |
| **customize.scanner-db.labels** | Scanner DB が作成するすべてのオブジェクトにアタッチするカスタムラベルです。 |
| **customize.scanner-db.annotations** | Scanner DB が作成するすべてのオブジェクトにアタッチするカスタムアノテーションです。 |
| **customize.scanner-db.podLabels** | すべての Scanner DB のデプロイメントにアタッチするカスタムラベルです。 |
| **customize.scanner-db.podAnnotations** | すべての Scanner DB のデプロイメントにアタッチするカスタムアノテーションです。 |
| **customize.scanner-db.envVars** | すべての Scanner DB コンテナーのカスタム環境変数です。 |

以下のように使用することもできます。

- すべてのオブジェクトのラベルとアノテーションを指定するための **customize.other.service/*.labels** および **customize.other.service/*.annotations** パラメーターです。
- または、特定のサービス名を指定します。たとえば、**customize.other.service/central-loadbalancer.labels** と **customize.other.service/central-loadbalancer.annotations** をパラメーターとして指定し、それらの値を設定します。

#### 3.3.1.2.2.8. 高度なカスタマイズ



**重要**

このセクションで指定されているパラメーターは、情報提供のみを目的としています。Red Hat は、namespace とリリース名が変更された Red Hat Advanced Cluster Security for Kubernetes インスタンスをサポートしていません。

| パラメーター | 説明 |
| --- | --- |
| **allowNonstandardNamespace** | **true** を使用して、Red Hat Advanced Cluster Security for Kubernetes をデフォルトの namespace **stackrox** 以外の namespace にデプロイします。 |
| **allowNonstandardReleaseName** | **true** を使用して、Red Hat Advanced Cluster Security for Kubernetes をデフォルトの **stackrox-central-services** 以外のリリース名でデプロイします。 |

**関連情報**

- 接続文字列 - PostgreSQL ドキュメント
- 設定ファイルを介したパラメーターのやりとり - PostgreSQL ドキュメント
- pg_hba.conf ファイル - PostgreSQL ドキュメント

#### 3.3.1.2.3. セントラルサービス Helm チャートのインストール

**values-public.yaml** ファイルと **values-private.yaml** ファイルを設定した後、**central-services** Helm チャートをインストールして、集中型コンポーネント (Central と Scanner) をデプロイします。

**手順**

- 以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> 1
```

1 **-f** オプションを使用して、YAML 設定ファイルのパスを指定します。

### 3.3.1.3. central-services Helm チャートをデプロイした後の設定オプションの変更

**central-services** Helm チャートをデプロイした後、任意の設定オプションに変更を加えることができます。

**手順**

1. **values-public.yaml** および **values-private.yaml** 設定ファイルを新しい値で更新します。
2. **helm upgrade** コマンドを実行し、**-f** オプションを使用して設定ファイルを指定します。

```
$ helm upgrade -n stackrox \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> \
  -f <path_to_values_private.yaml>
```

**注記**

**--set** または **--set-file** パラメーターを使用して設定値を指定することもできます。ただし、これらのオプションは保存されないため、変更を加えるたびにすべてのオプションを手動で再度指定する必要があります。

### 3.3.2. roxctl CLI を使用して Central をインストールする

**警告**

実稼働環境では、Red Hat は Operator または Helm チャートを使用して RHACS をインストールすることを推奨しています。この方法を使用する必要がある特定のインストールがない限り、**roxctl** のインストール手法を使用しないでください。

#### 3.3.2.1. roxctl CLI のインストール

Red Hat Advanced Cluster Security for Kubernetes をインストールするには、バイナリーをダウンロードして **roxctl** CLI をインストールする必要があります。**roxctl** は、Linux、Windows、または macOS にインストールできます。

##### 3.3.2.1.1. Linux への roxctl CLI のインストール

次の手順を使用して、Linux に **roxctl** CLI バイナリーをインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Linux/roxctl
```

2. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

3. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 3.3.2.1.2. macOS への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを macOS にインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Darwin/roxctl
```

2. バイナリーからすべての拡張属性を削除します。

```
$ xattr -c roxctl
```

3. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

4. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 3.3.2.1.3. Windows への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを Windows にインストールできます。

**手順**

- **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Windows/roxctl.exe
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

### 3.3.2.2. 対話型インストーラーの使用

対話型インストーラーを使用して、お使いの環境に必要なシークレット、デプロイメント設定、およびデプロイメントスクリプトを生成します。

**手順**

1. 対話型インストールコマンドを実行します。

```
$ roxctl central generate interactive
```

**重要**

**roxctl** CLI を使用して Red Hat Advanced Cluster Security for Kubernetes をインストールすると、下位互換性のためにデフォルトで PodSecurityPolicy (PSP) オブジェクトが作成されます。RHACS を Kubernetes バージョン 1.25 以降または OpenShift Container Platform バージョン 4.12 以降にインストールする場合、PSP オブジェクトの作成を無効にする必要があります。これを行うには、**roxctl central generate** コマンドと **roxctl sensor generate** コマンドで **--enable-pod-security-policies** オプションを **false** に指定します。

2. **Enter** を押してプロンプトのデフォルト値を受け入れるか、必要に応じてカスタム値を入力します。

```
Enter path to the backup bundle from which to restore keys and certificates (optional):
Enter PEM cert bundle file (optional): 1
Enter administrator password (default: autogenerated):
Enter orchestrator (k8s, openshift): openshift
Enter the directory to output the deployment bundle to (default: "central-bundle"):
Enter the OpenShift major version (3 or 4) to deploy on (default: "0"): 4
Enter Istio version when deploying into an Istio-enabled cluster (leave empty when not
running Istio) (optional):
Enter the method of exposing Central (route, lb, np, none) (default: "none"): route 2
Enter main image to use (default: "stackrox.io/main:3.0.61.1"):
Enter whether to run StackRox in offline mode, which avoids reaching out to the Internet
(default: "false"):
Enter whether to enable telemetry (default: "true"):
Enter the deployment tool to use (kubectl, helm, helm-values) (default: "kubectl"):
Enter Scanner DB image to use (default: "stackrox.io/scanner-db:2.15.2"):
Enter Scanner image to use (default: "stackrox.io/scanner:2.15.2"):
Enter Central volume type (hostpath, pvc): pvc 3
Enter external volume name (default: "stackrox-db"):
Enter external volume size in Gi (default: "100"):
Enter storage class name (optional if you have a default StorageClass configured):
```

1. カスタム TLS 証明書を追加する場合は、PEM でエンコードされた証明書のファイルパスを指定します。カスタム証明書を指定すると、対話型インストーラーは、使用しているカスタム証明書の PEM 秘密鍵を提供するように要求します。

2. RHACS ポータルを使用するには、ルート、ロードバランサー、またはノードポートを使用して Central を公開する必要があります。

3. hostPath ボリュームを使用して OpenShift Container Platform に Red Hat Cluster Security for Kubernetes をインストールする場合は、SELinux ポリシーを変更する必要があります。

**警告**

OpenShift Container Platform で、hostPath ボリュームを使用するには、SELinux ポリシーを変更して、ホストとコンテナーが共有するディレクトリーへのアクセスを許可する必要があります。これは、SELinux がデフォルトでディレクトリー共有をブロックしているためです。SELinux ポリシーを変更するには、次のコマンドを実行します。

```
$ sudo chcon -Rt svirt_sandbox_file_t <full_volume_path>
```

ただし、Red Hat は SELinux ポリシーの変更を推奨していません。代わりに、OpenShift Container Platform にインストールするときに PVC を使用してください。

完了すると、インストーラーは central-bundle という名前のフォルダーを作成します。このフォルダーには、Central をデプロイするために必要な YAML マニフェストとスクリプトが含まれています。さらに、信頼できる認証局である Central と Scanner をデプロイするために実行する必要があるスクリプトの画面上の説明と、RHACS ポータルにログインするための認証手順、プロンプトに答える際にパスワードを入力しなかった場合は自動生成されたパスワードも表示されます。

#### 3.3.2.3. Central インストールスクリプトの実行

対話型インストーラーを実行したら、**setup.sh** スクリプトを実行して Central をインストールできます。

**手順**

1. **setup.sh** スクリプトを実行して、イメージレジストリーアクセスを設定します。

```
$ ./central-bundle/central/scripts/setup.sh
```

2. 必要なリソースを作成します。

```
$ oc create -R -f central-bundle/central
```

3. デプロイメントの進行状況を確認します。

```
$ oc get pod -n stackrox -w
```

4. Central の実行後、RHACS ポータルの IP アドレスを見つけて、ブラウザーで開きます。プロンプトに応答するときに選択した公開方法に応じて、次のいずれかの方法を使用して IP アドレスを取得します。

| 公開方法 | コマンド | アドレス | 例 |
|---|---|---|---|
| ルート | **oc -n stackrox get route central** | 出力の **HOST/PORT** 列の下のアドレス | **https://central-stackrox.example.route** |

| 公開方法 | コマンド | アドレス | 例 |
|---|---|---|---|
| ノードポート | **oc get node -owide && oc -n stackrox get svc central-loadbalancer** | サービス用に表示されたポート上の任意のノードの IP またはホスト名 | **https://198.51.100.0:31489** |
| ロードバランサー | **oc -n stackrox get svc central-loadbalancer** | EXTERNAL-IP、またはポート 443 でサービスに表示されるホスト名 | **https://192.0.2.0** |
| なし | **central-bundle/central/scripts/port-forward.sh 8443** | **https://localhost:8443** | **https://localhost:8443** |

**注記**

対話型インストール中に自動生成されたパスワードを選択した場合は、次のコマンドを実行して、Central にログインするためのパスワードを確認できます。

```
$ cat central-bundle/password
```

## 3.4. 他のプラットフォームでの RHACS の INIT バンドルの生成と適用

**SecuredCluster** リソースをクラスターにインストールする前に、init バンドルを作成する必要があります。**SecuredCluster** がインストールおよび設定されているクラスターは、このバンドルを使用して Central で認証します。RHACS ポータルまたは **roxctl** CLI を使用して、init バンドルを作成できます。次に、それを使用してリソースを作成することにより、init バンドルを適用します。

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

### 3.4.1. init バンドルの生成

#### 3.4.1.1. RHACS ポータルを使用した init バンドルの生成

RHACS ポータルを使用して、シークレットを含む init バンドルを作成できます。

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

**手順**

1. 公開方法に基づいて RHACS ポータルのアドレスを見つけます。

a. ルートの場合。

```
$ oc get route central -n stackrox
```

b. ロードバランサーの場合。

```
$ oc get service central-loadbalancer -n stackrox
```

c. port forward の場合:

i. 以下のコマンドを実行します。

```
$ oc port-forward svc/central 18443:443 -n stackrox
```

ii. **https://localhost:18443/** に移動します。

2. RHACS ポータルで、**Platform Configuration → Integrations** に移動します。
3. **Authentication Tokens** セクションに移動し、**Cluster Init Bundle** をクリックする。
4. **Generate bundle** をクリックする。
5. クラスター初期化バンドルの名前を入力し、**Generate** をクリックする。
   a. Helm チャートを使用してインストールする場合は、**Download Helm Values File**をクリックして、生成されたバンドルをダウンロードします。
   b. Operator を使用してインストールする場合は、**Download Kubernetes Secret File** をクリックして、生成されたバンドルをダウンロードします。

**重要**

このバンドルにはシークレットが含まれているため、セキュアに保管してください。同じバンドルを使用して、複数のセキュアなクラスターを作成できます。

**次のステップ**

1. セキュアなクラスターでリソースを作成して、init バンドルを適用します。
2. 各クラスターにセキュアなクラスターサービスをインストールします。

#### 3.4.1.2. roxctl CLI を使用した init バンドルの生成

**roxctl** CLI を使用して、シークレットを含む init バンドルを作成できます。

**注記**

init バンドルを作成するには、**Admin** ユーザーロールが必要です。

**前提条件**

**ROX_API_TOKEN** および **ROX_CENTRAL_ADDRESS** 環境変数が設定されている。

- **ROX_API_TOKEN** および **ROX_CENTRAL_ADDRESS** 環境変数を設定します。

```
$ export ROX_API_TOKEN=<api_token>
```

```
$ export ROX_CENTRAL_ADDRESS=<address>:<port_number>
```

**手順**

- 次のコマンドを実行して、シークレットを含むクラスター初期化バンドルを生成します。
  Helm のインストールの場合:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output cluster_init_bundle.yaml
```

  Operator のインストールの場合:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output-secrets cluster_init_bundle.yaml
```

**重要**

このバンドルにはシークレットが含まれているため、安全に保管してください。同じバンドルを使用して、複数のセキュアなクラスターを設定できます。

**次の手順**

- Red Hat OpenShift CLI を使用して、init バンドルを使用してリソースを作成します。

#### 3.4.1.3. init バンドルを使用したリソースの作成

セキュアなクラスターをインストールする前に、init バンドルを使用して、セキュアなクラスター上のサービスが Central と通信できるようにする必要なリソースをクラスター上に作成する必要があります。

**注記**

Helm チャートを使用してインストールする場合は、この手順を実行しないでください。Helm を使用してインストールを完了します。関連情報セクションの 「Helm チャートを使用した保護されたクラスターへの RHACS のインストール」を参照してください。

**前提条件**

- シークレットを含む init バンドルを生成している必要があります。

**手順**

リソースを作成するには、次のいずれかの手順を実行します。

- OpenShift Container Platform Web コンソールのトップメニューで + をクリックし、**Import YAML** ページを開きます。init バンドルファイルをドラッグするか、その内容をコピーしてエディターに貼り付け、**Create** をクリックします。

- Red Hat OpenShift CLI を使用して、次のコマンドを実行してリソースを作成します。

```
$ oc create -f <init_bundle>.yaml \ 1
  -n <stackrox> 2
```

1 シークレットを含む init バンドルのファイル名を指定します。

2 Central サービスがインストールされているプロジェクトの名前を指定します。

- **kubectl** CLI を使用して、次のコマンドを実行してリソースを作成します。

```
$ kubectl create namespace stackrox 1
$ kubectl create -f <init_bundle>.yaml \ 2
  -n <stackrox> 3
```

1 セキュアなクラスターリソースがインストールされるプロジェクトを作成します。この例では **stackrox** を使用します。

2 シークレットを含む init バンドルのファイル名を指定します。

3 作成したプロジェクト名を指定します。この例では **stackrox** を使用します。

**次の手順**

- モニターするすべてのクラスターに RHACS セキュアクラスターサービスをインストールします。

## 3.5. 他のプラットフォームでの RHACS 用のセキュアなクラスターサービスのインストール

RHACS は、Amazon Elastic Kubernetes Service (Amazon EKS)、Google Kubernetes Engine (Google GKE)、Microsoft Azure Kubernetes Service (Microsoft AKS) などのプラットフォームの安全なクラスターにインストールできます。

### 3.5.1. Helm チャートを使用したセキュアなクラスターへの RHACS のインストール

Helm チャートをカスタマイズせずに使用するか、デフォルト値を使用するか、設定パラメーターをカスタマイズして、セキュアなクラスターに RHACS をインストールできます。

#### 3.5.1.1. カスタマイズせずに Helm チャートを使用して、セキュアなクラスターに RHACS をインストールする

##### 3.5.1.1.1. Helm チャートリポジトリーの追加

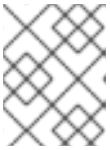
**手順**

- RHACS チャートリポジトリーを追加します。

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

Red Hat Advanced Cluster Security for Kubernetes の Helm リポジトリーには、異なるコンポーネントをインストールするための Helm チャートが含まれています。

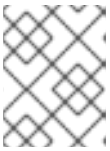
- 集中型コンポーネント (Central および Scanner) をインストールするためのセントラルサービス Helm チャート (**central-services**)。



**注記**

一元化されたコンポーネントを 1 回だけデプロイし、同じインストールを使用して複数の個別のクラスターをモニターできます。

- クラスターごと (Sensor および Admission Controller) およびノードごと (Collector) のコンポーネントをインストールするための Secured Cluster Services Helm チャート (**secured-cluster-services**)。



**注記**

モニターする各クラスターにクラスターごとのコンポーネントをデプロイし、モニターするすべてのノードにノードごとのコンポーネントをデプロイします。

**検証**

- 次のコマンドを実行して、追加されたチャートリポジトリーを確認します。

```
$ helm search repo -l rhacs/
```

#### 3.5.1.1.2. カスタマイズせずに secured-cluster-services Helm チャートをインストールする

次の手順を使用して、**secured-cluster-services** Helm チャートをインストールし、クラスターごとおよびノードごとのコンポーネント (Sensor、アドミッションコントローラー、および Collector) をデプロイします。

**注意**

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**前提条件**

- クラスターの RHACS init バンドルを生成しておく必要があります。
- Central service を公開するアドレスとポート番号が必要です。

**手順**

- Kubernetes ベースのクラスターで次のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
    stackrox-secured-cluster-services rhacs/secured-cluster-services \
    -f <path_to_cluster_init_bundle.yaml> \ 1
    --set clusterName=<name_of_the_secured_cluster> \
    --set centralEndpoint=<endpoint_of_central_service> 2
```

1. **-f** オプションを使用して、init バンドルのパスを指定します。
2. Central のアドレスとポート番号を指定します。例: **acs.domain.com:443**

- OpenShift Container Platform クラスターで以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <path_to_cluster_init_bundle.yaml> \ 1
  --set clusterName=<name_of_the_secured_cluster> \
  --set centralEndpoint=<endpoint_of_central_service> 2
  --set scanner.disable=false
```

1. **-f** オプションを使用して、init バンドルのパスを指定します。
2. Central のアドレスとポート番号を指定します。例: **acs.domain.com:443**

**関連情報**

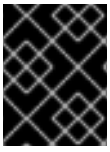
- 他のプラットフォームでの RHACS の init バンドルの生成と適用

#### 3.5.1.2. カスタマイズによる secure-cluster-services Helm チャートの設定

このセクションでは、**helm install** および **helm upgrade** コマンドで使用できる Helm チャート設定パラメーターについて説明します。これらのパラメーターは、**--set** オプションを使用するか、YAML 設定ファイルを作成することで指定できます。

以下のファイルを作成して、Red Hat Advanced Cluster Security for Kubernetes をインストールするための Helm チャートを設定します。

- パブリック設定ファイル **values-public.yaml**: このファイルを使用して、機密性の低いすべての設定オプションを保存します。
- プライベート設定ファイル **values-private.yaml**: このファイルを使用して、機密性の高いすべての設定オプションを保存します。このファイルは安全に保管してください。



**重要**

**Download Helm Values File** Helm チャートを使用している間は、チャートの一部である **values.yaml** ファイルを変更しないでください。

##### 3.5.1.2.1. 設定パラメーター

| パラメーター | 説明 |
| --- | --- |
| **clusterName** | クラスターの名前です。 |

| パラメーター | 説明 |
| --- | --- |
| **centralEndpoint** | Central エンドポイントのアドレス (ポート番号を含む)。gRPC に対応していないロードバランサーを使用している場合は、エンドポイントアドレスの前に **wss://** を付けて、WebSocket プロトコルを使用します。複数のクラスターを設定する場合は、アドレスにホスト名を使用します (例: **central.example.com:443**)。 |
| **sensor.endpoint** | ポート番号を含む Sensor エンドポイントのアドレスです。 |
| **sensor.imagePullPolicy** | Sensor コンテナーのイメージプルポリシーです。 |
| **sensor.serviceTLS.cert** | Sensor が使用する内部サービス間の TLS 証明書です。 |
| **sensor.serviceTLS.key** | Sensor が使用する内部サービス間 TLS 証明書キーです。 |
| **sensor.resources.requests.memory** | Sensor コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.requests.cpu** | Sensor コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.limits.memory** | Sensor コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.resources.limits.cpu** | センサーコンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **sensor.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Sensor が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **sensor.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Sensor の taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **image.main.name** | **main** イメージの名前です。 |

| パラメーター | 説明 |
| --- | --- |
| **image.collector.name** | Collector イメージの名前です。 |
| **image.main.registry** | main イメージに使用しているレジストリーのアドレスです。 |
| **image.collector.registry** | Collector イメージに使用しているレジストリーのアドレスです。 |
| **image.main.pullPolicy** | **main** イメージのイメージプルポリシーです。 |
| **image.collector.pullPolicy** | Collector イメージのイメージプルポリシーです。 |
| **image.main.tag** | 使用する **main** イメージのタグです。 |
| **image.collector.tag** | 使用する **collector** イメージのタグです。 |
| **collector.collectionMethod** | **EBPF**、**KERNEL_MODULE**、または **NO_COLLECTION** のいずれかです。 |
| **collector.imagePullPolicy** | Collector コンテナーのイメージプルポリシーです。 |
| **collector.complianceImagePullPolicy** | Compliance コンテナーのイメージプルポリシーです。 |
| **collector.disableTaintTolerations** | **false** を指定すると、許容値が Collector に適用され、Collector Pod は taint のあるすべてのノードにスケジュールできます。**true** として指定すると、許容値は適用されず、Collector Pod は taint のあるノードにスケジュールされません。 |
| **collector.resources.requests.memory** | Collector コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.requests.cpu** | Collector コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.limits.memory** | Collector コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.resources.limits.cpu** | Collector コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **collector.complianceResources.requests.memory** | Compliance コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.requests.cpu** | Compliance の CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.limits.memory** | Compliance コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.complianceResources.limits.cpu** | Compliance コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **collector.serviceTLS.cert** | Collector が使用する内部サービス間 TLS 証明書です。 |
| **collector.serviceTLS.key** | Collector が使用する内部サービス間 TLS 証明書キーです。 |
| **admissionControl.listenOnCreates** | この設定は、Kubernetes がワークロード作成イベントの **AdmissionReview** リクエストで Red Hat Advanced Cluster Security for Kubernetes に接続するように設定されているかどうかを制御します。 |
| **admissionControl.listenOnUpdates** | このパラメーターを **false** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、Kubernetes API サーバーがオブジェクト更新イベントを送信しないように **ValidatingWebhookConfiguration** を作成します。オブジェクトの更新ボリュームは通常、オブジェクトが作成するボリュームよりも多いため、これを **false** のままにしておくと、アドミッションコントロールサービスのロードが制限され、アドミッションコントロールサービスが誤動作する可能性が低くなります。 |
| **admissionControl.listenOnEvents** | この設定は、クラスターが Kubernetes **exec** および **portforward** イベントの **AdmissionReview** リクエストで Red Hat Advanced Cluster Security for Kubernetes に接続するように設定されているかどうかを制御します。Red Hat Advanced Cluster Security for Kubernetes は、OpenShift Container Platform 3.11 でこの機能をサポートしていません。詳細は、Red Hat Advanced Cluster Security for Kubernetes Support Policy を参照してください。 |

| パラメーター | 説明 |
| --- | --- |
| **admissionControl.dynamic.enforceOnCreates** | この設定は、Red Hat Advanced Cluster Security for Kubernetes がポリシーを評価するかどうかを制御します。無効にすると、すべての AdmissionReview リクエストが自動的に受け入れられます。 |
| **admissionControl.dynamic.enforceOnUpdates** | この設定は、アドミッションコントロールサービスの動作を制御します。これを機能させるには、**listenOnUpdates** を **true** として指定する必要があります。 |
| **admissionControl.dynamic.scanInline** | このオプションを **true** に設定すると、アドミッションコントロールサービスは、アドミッションデシジョンを行う前にイメージスキャンをリクエストします。イメージスキャンには数秒かかるため、このオプションを有効にするのは、クラスターで使用されるすべてのイメージがデプロイ前にスキャンされることを確認できる場合のみです (たとえば、イメージビルド中の CI 統合によって)。このオプションは、RHACS ポータルの **Contact image scanners** オプションに対応しています。 |
| **admissionControl.dynamic.disableBypass** | アドミッションコントローラーのバイパスを無効にするには、**true** に設定します。 |
| **admissionControl.dynamic.timeout** | アドミッションレビューリクエストを評価する間、Red Hat Advanced Cluster Security for Kubernetes が待機する最大時間 (秒単位) です。これを使用して、イメージスキャンを有効にするときにリクエストのタイムアウトを設定します。イメージスキャンが指定された時間より長く実行される場合、Red Hat Advanced Cluster Security for Kubernetes はリクエストを受け入れます。 |
| **admissionControl.resources.requests.memory** | Admission Control コンテナーのメモリーリクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.requests.cpu** | Admission Control コンテナーの CPU リクエストです。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.limits.memory** | Admission Control コンテナーのメモリー制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **admissionControl.resources.limits.cpu** | Admission Control コンテナーの CPU 制限です。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **admissionControl.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Admission Control をスケジュールするように強制します。 |
| **admissionControl.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、アドミッションコントロールの taint toleration キー、値、および effect を指定します。このパラメーターは、主にインフラストラクチャーノードに使用されます。 |
| **admissionControl.serviceTLS.cert** | Admission Control が使用する内部サービス間 TLS 証明書です。 |
| **admissionControl.serviceTLS.key** | Admission Control が使用する内部サービス間 TLS 証明書キーです。 |
| **registryOverride** | このパラメーターを使用して、デフォルトの **docker.io** レジストリーをオーバーライドします。他のレジストリーを使用している場合は、レジストリーの名前を指定してください。 |
| **collector.disableTaintTolerations** | **false** を指定すると、許容値が Collector に適用され、Collector Pod は taint のあるすべてのノードにスケジュールできます。**true** として指定した場合、許容値は適用されず、Collector Pod は taint のあるノードにスケジュールされません。 |
| **createUpgraderServiceAccount** | **true** を指定して、**sensor-upgrader** アカウントを作成します。デフォルトでは、Red Hat Advanced Cluster Security for Kubernetes は、セキュアなクラスターごとに **sensor-upgrader** と呼ばれるサービスアカウントを作成します。このアカウントは高い権限を持ちますが、アップグレードの時のみ使用されます。このアカウントを作成しない場合、Sensor に十分な権限がない場合は、将来のアップグレードを手動で完了する必要があります。 |
| **createSecrets** | **false** を指定すると、Sensor、Collector、および、アドミッションコントローラーのオーケストレーターシークレットの作成がスキップされます。 |

| パラメーター | 説明 |
| --- | --- |
| **collector.slimMode** | Collector のデプロイにスリムな Collector イメージを使用する場合は、**true** を指定します。slim Collector イメージを使用するには、一致する eBPF プローブまたはカーネルモジュールを提供する必要があります。Red Hat Advanced Cluster Security for Kubernetes をオフラインモードで実行している場合、スリム Collector が機能するには、stackrox.io からカーネルサポートパッケージをダウンロードして Central にアップロードする必要があります。それ以外の場合は、Central が https://collector-modules.stackrox.io/ でホストされているオンラインプローブリポジトリーにアクセスできることを確認する必要があります。 |
| **sensor.resources** | Sensor のリソース仕様です。 |
| **admissionControl.resources** | アドミッションコントローラーのリソース仕様です。 |
| **collector.resources** | Collector のリソース仕様です。 |
| **collector.complianceResources** | Collector の Compliance コンテナーのリソース仕様です。 |
| **exposeMonitoring** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、Sensor、Collector,、およびアドミッションコントローラーのポート番号 9090 で Prometheus メトリクスエンドポイントを公開します。 |
| **auditLogs.disableCollection** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、設定マップとシークレットへのアクセスと変更を検出するために使用される監査ログ検出機能を無効にします。 |
| **scanner.disable** | このオプションを **false** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は、セキュアなクラスターに軽量な Scanner と Scanner DB をデプロイして、OpenShift Container Registry でイメージをスキャンできるようにします。Scanner の有効化は、OpenShift でのみサポートされます。デフォルト値は **true** です。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.replicas** | Collector の Compliance コンテナーのリソース仕様です。 |
| **scanner.logLevel** | このパラメーターを設定すると、Scanner のログレベルを変更できます。このオプションは、トラブルシューティングの目的でのみ使用してください。 |
| **scanner.autoscaling.disable** | このオプションを **true** に設定すると、Red Hat Advanced Cluster Security for Kubernetes は Scanner のデプロイメントでの自動スケーリングを無効にします。 |
| **scanner.autoscaling.minReplicas** | 自動スケーリングのレプリカの最小数です。デフォルトは 2 です。 |
| **scanner.autoscaling.maxReplicas** | 自動スケーリングのレプリカの最大数です。デフォルトは 5 です。 |
| **scanner.nodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、指定されたラベルを持つノードでのみ Scanner をスケジュールするように強制します。 |
| **scanner.tolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner の taint toleration キー、値、および effect を指定します。 |
| **scanner.dbNodeSelector** | ノードセレクターラベルを **label-key:label-value** として指定して、Scanner DB が指定されたラベルを持つノードでのみスケジュールするように強制します。 |
| **scanner.dbTolerations** | ノードセレクターが taint されたノードを選択する場合は、このパラメーターを使用して、Scanner DB の taint toleration キー、値、および effect を指定します。 |
| **scanner.resources.requests.memory** | Scanner コンテナーのメモリーリクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.resources.requests.cpu** | Scanner コンテナーの CPU リクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

| パラメーター | 説明 |
| --- | --- |
| **scanner.resources.limits.memory** | Scanner コンテナーのメモリー制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.resources.limits.cpu** | Scanner コンテナーの CPU 制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.requests.memory** | Scanner DB コンテナーのメモリーリクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.requests.cpu** | Scanner DB コンテナーの CPU リクエスト。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.limits.memory** | Scanner DB コンテナーのメモリー制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |
| **scanner.dbResources.limits.cpu** | Scanner DB コンテナーの CPU 制限。このパラメーターを使用して、デフォルト値をオーバーライドします。 |

##### 3.5.1.2.1.1. 環境変数

Sensor と、アドミッションコントローラーの環境変数は、次の形式で指定できます。

```
customize:
  envVars:
    ENV_VAR1: "value1"
    ENV_VAR2: "value2"
```

**customize** 設定を使用すると、この Helm チャートによって作成されたすべてのオブジェクトのカスタム Kubernetes メタデータ (ラベルとアノテーション) と、ワークロードの追加の Pod ラベル、Pod アノテーション、コンテナー環境変数を指定できます。

より一般的なスコープ (たとえば、すべてのオブジェクト) で定義されたメタデータを、より狭いスコープ (たとえば、Sensor デプロイメントのみ) で定義されたメタデータでオーバーライドできるという意味で、設定は階層的です。

#### 3.5.1.2.2. secure-cluster-services Helm チャートのインストール

**values-public.yaml** ファイルと **values-private.yaml** ファイルを設定した後、**secured-cluster-services** Helm チャートをインストールして、クラスターごと、およびノードごとのコンポーネント (Sensor、アドミッションコントローラー、Collector) をデプロイします。

**注意**

Unified Extensible Firmware Interface (UEFI) があり、Secure Boot が有効になっているシステムに Collector をインストールするには、カーネルモジュールが署名されておらず、UEFI ファームウェアが署名されていないパッケージをロードできないため、eBPF プローブを使用する必要があります。Collector は、開始時に Secure Boot ステータスを識別し、必要に応じて eBPF プローブに切り替えます。

**前提条件**

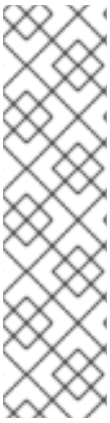
- クラスターの RHACS init バンドルを生成しておく必要があります。
- Central service を公開するアドレスとポート番号が必要です。

**手順**

- 以下のコマンドを実行します。

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <name_of_cluster_init_bundle.yaml> \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> (1)
```

(1) **-f** オプションを使用して、YAML 設定ファイルのパスを指定します。



**注記**

継続的インテグレーション (CI) システムを使用して **secured-cluster-services** Helm チャートをデプロイするには、init バンドル YAML ファイルを環境変数として **helm install** コマンドに渡します。

```
$ helm install ... -f <(echo "$INIT_BUNDLE_YAML_SECRET") (1)
```

(1) base64 でエンコードされた変数を使用している場合は、代わりに **helm install ... -f <(echo "$INIT_BUNDLE_YAML_SECRET" | base64 --decode)** コマンドを使用してください。

**関連情報**

- 他のプラットフォームでの RHACS の init バンドルの生成と適用

### 3.5.1.3. secure-cluster-services Helm チャートをデプロイした後の設定オプションの変更

**secure-cluster-services** Helm チャートをデプロイした後、任意の設定オプションに変更を加えることができます。

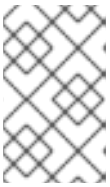
**手順**

1. **values-public.yaml** および **values-private.yaml** 設定ファイルを新しい値で更新します。
2. **helm upgrade** コマンドを実行し、**-f** オプションを使用して設定ファイルを指定します。

```
$ helm upgrade -n stackrox \
```

```
stackrox-secured-cluster-services rhacs/secured-cluster-services \
--reuse-values \ 1
-f <path_to_values_public.yaml> \
-f <path_to_values_private.yaml>
```

1 **--reuse-values** パラメーターを指定する必要があります。指定しない場合、Helm upgrade コマンドは以前に設定されたすべての設定をリセットします。



**注記**

**--set** または **--set-file** パラメーターを使用して設定値を指定することもできます。ただし、これらのオプションは保存されないため、変更を加えるたびにすべてのオプションを手動で再度指定する必要があります。

### 3.5.2. roxctl CLI を使用したセキュアなクラスターへの RHACS のインストール

CLI を使用してセキュアなクラスターに RHACS をインストールするには、次の手順を実行します。

1. **roxctl** CLI をインストールします。
2. Sensor を取り付けます。

#### 3.5.2.1. roxctl CLI のインストール

最初にバイナリーをダウンロードする必要があります。**roxctl** は、Linux、Windows、または macOS にインストールできます。

##### 3.5.2.1.1. Linux への roxctl CLI のインストール

次の手順を使用して、Linux に **roxctl** CLI バイナリーをインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Linux/roxctl
```

2. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

3. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 3.5.2.1.2. macOS への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを macOS にインストールできます。

**手順**

1. **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Darwin/roxctl
```

2. バイナリーからすべての拡張属性を削除します。

```
$ xattr -c roxctl
```

3. **roxctl** バイナリーを実行可能にします。

```
$ chmod +x roxctl
```

4. **PATH** 上にあるディレクトリーに **roxctl** バイナリーを配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

#### 3.5.2.1.3. Windows への roxctl CLI のインストール

次の手順を使用して、**roxctl** CLI バイナリーを Windows にインストールできます。

**手順**

- **roxctl** CLI の最新バージョンをダウンロードします。

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.0.5/bin/Windows/roxctl.exe
```

**検証**

- インストールした **roxctl** のバージョンを確認します。

```
$ roxctl version
```

### 3.5.2.2. Sensor のインストール

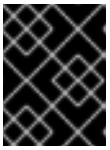
クラスターをモニターするには、Sensor をデプロイする必要があります。モニターする各クラスターに Sensor をデプロイする必要があります。次の手順では、RHACS ポータルを使用して Sensor を追加する方法について説明します。

**前提条件**

- Central サービスをすでにインストールしている必要があります。または、Red Hat Advanced Cluster Security Cloud Service (RHACS Cloud Service) で **ACS インスタンス** を選択して Central サービスにアクセスできます。

**手順**

1. セキュアなクラスターの RHACS ポータルで、**Platform Configuration → Clusters** に移動します。
2. **+NewCluster** を選択します。
3. クラスターの名前を指定します。
4. Sensor をデプロイする場所に基づいて、フィールドに適切な値を入力します。
   - 同じクラスターに Sensor をデプロイする場合は、すべてのフィールドのデフォルト値を受け入れます。
   - 別のクラスターにデプロイする場合は、**central.stackrox.svc:443** を、他のクラスターからアクセス可能なロードバランサー、ノードポート、またはポート番号を含む他のアドレスに置き換えます。
   - HAProxy、AWS Application Load Balancer (ALB)、AWS Elastic Load Balancing (ELB) などの非 gRPC 対応のロードバランサーを使用している場合は、WebSocket Secure (**wss**) プロトコルを使用してください。**wss** を使用するには:
     - アドレスの前に **wss://** を付けます。
     - アドレスの後にポート番号を追加します (例 **wss://stackrox-central.example.com:443**)。
5. **Next** をクリックして、Sensor のセットアップを続行します。
6. **Download YAML File and Keys**をクリックして、クラスターバンドル (zip アーカイブ) をダウンロードします。



**重要**

クラスターバンドルの zip アーカイブには、クラスターごとに固有の設定とキーが含まれています。同じファイルを別のクラスターで再利用しないでください。

7. モニター対象クラスターにアクセスできるシステムから、クラスターバンドルから **sensor** スクリプトを解凍して実行します。

```
$ unzip -d sensor sensor-<cluster_name>.zip
```

```
$ ./sensor/sensor.sh
```

Sensor をデプロイするために必要な権限がないという警告が表示された場合は、画面の指示に従うか、クラスター管理者に連絡して支援を求めてください。

Sensor はデプロイされた後、Central に接続し、クラスター情報を提供します。

**検証**

1. RHACS ポータルに戻り、デプロイメントが成功したかどうかを確認します。成功した場合、**Platform Configuration → Clusters** でクラスターのリストを表示すると、クラスターのステータスに緑色のチェックマークと **Healthy** ステータスが表示されます。緑色のチェックマークが表示されない場合は、次のコマンドを使用して問題を確認してください。
   - Kubernetes で、次のコマンドを入力します。

     ```
     $ kubectl get pod -n stackrox -w
     ```

2. **Finish** をクリックしてウィンドウを閉じます。

インストール後、Sensor はセキュリティー情報の RHACS へのレポートを開始し、RHACS ポータルダッシュボードは、Sensor をインストールしたクラスターからのデプロイメント、イメージ、およびポリシー違反を表示し始めます。

## 3.6. 他のプラットフォームでの RHACS のインストールの確認

RHACS が正しくインストールされていることを確認する手順を示します。

### 3.6.1. インストールの検証

インストールが完了したら、いくつかの脆弱なアプリケーションを実行し、RHACS ポータルに移動して、セキュリティー評価とポリシー違反の結果を評価します。

**注記**

次のセクションにリストされているサンプルアプリケーションには重大な脆弱性が含まれており、Red Hat Advanced Cluster Security for Kubernetes のビルドおよびデプロイ時の評価機能を検証するように特別に設計されています。

インストールの検証

1. 公開方法に基づいて RHACS ポータルのアドレスを見つけます。
   a. ロードバランサーの場合。

      ```
      $ kubectl get service central-loadbalancer -n stackrox
      ```

   b. port forward の場合:
      i. 以下のコマンドを実行します。

         ```
         $ kubectl port-forward svc/central 18443:443 -n stackrox
         ```

      ii. **https://localhost:18443/** に移動します。
2. 新規の namespace を作成します。

   ```
   $ kubectl create namespace test
   ```

3. 重大な脆弱性を持ついくつかのアプリケーションを開始します。

```
$ kubectl run shell --labels=app=shellshock,team=test-team \
  --image=vulnerables/cve-2014-6271 -n test
$ kubectl run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes は、これらのデプロイメントがクラスターに送信されるとすぐに、これらのデプロイメントを自動的にスキャンしてセキュリティーリスクとポリシー違反を検出します。RHACS ポータルに移動して、違反を表示します。デフォルトのユーザー名 **admin** と生成されたパスワードを使用して RHACS ポータルにログインできます。

# 第4章 RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES のアンインストール

Red Hat Advanced Cluster Security for Kubernetes をインストールすると、以下が作成されます。

- Operator のインストール方法を選択した場合は、Operator がインストールされる **rhacs-operator** という namespace
- **stackrox** と呼ばれる namespace 、または Central および SecuredCluster カスタムリソースを作成した別の namespace
- すべてのコンポーネントの **PodSecurityPolicy** および Kubernetes ロールベースアクセス制御 (RBAC) オブジェクト
- 生成されたネットワークポリシーで使用するための namespace の追加ラベル
- アプリケーションカスタムリソース定義 (CRD) (存在しない場合)

Red Hat Advanced Cluster Security for Kubernetes をアンインストールするには、これらのアイテムをすべて削除する必要があります。

## 4.1. NAMESPACE の削除

OpenShift Container Platform または Kubernetes コマンドラインインターフェイスを使用して、Red Hat Advanced Cluster Security for Kubernetes が作成する namespace を削除できます。

**手順**

- **stackrox** namespace を削除します。
  - OpenShift Container Platform

```
$ oc delete namespace stackrox
```

  - Kubernetes の場合:

```
$ kubectl delete namespace stackrox
```

**注記**

別の namespace に RHACS をインストールした場合は、**delete** コマンドでその namespace の名前を使用してください。

## 4.2. グローバルリソースの削除

OpenShift Container Platform または Kubernetes コマンドラインインターフェイスを使用して、Red Hat Advanced Cluster Security for Kubernetes が作成するグローバルリソースを削除できます。

**手順**

- グローバルリソースを削除します。
  - OpenShift Container Platform

```
$ oc get clusterrole,clusterrolebinding,role,rolebinding,psp -o name | grep stackrox |
xargs oc delete --wait
```

```
$ oc delete scc -l "app.kubernetes.io/name=stackrox"
```

```
$ oc delete ValidatingWebhookConfiguration stackrox
```

- Kubernetes の場合:

```
$ kubectl get clusterrole,clusterrolebinding,role,rolebinding,psp -o name | grep stackrox |
xargs kubectl delete --wait
```

```
$ kubectl delete ValidatingWebhookConfiguration stackrox
```

## 4.3. ラベルとアノテーションの削除

OpenShift Container Platform または Kubernetes コマンドラインインターフェイスを使用して、Red Hat Advanced Cluster Security for Kubernetes が作成するラベルとアノテーションを削除できます。

**手順**

- ラベルとアノテーションを削除します。
  - OpenShift Container Platform

```
$ for namespace in $(oc get ns | tail -n +2 | awk '{print $1}'); do     oc label namespace
$namespace namespace.metadata.stackrox.io/id-;     oc label namespace $namespace
namespace.metadata.stackrox.io/name-;     oc annotate namespace $namespace
modified-by.stackrox.io/namespace-label-patcher-;   done
```

  - Kubernetes の場合:

```
$ for namespace in $(kubectl get ns | tail -n +2 | awk '{print $1}'); do     kubectl label
namespace $namespace namespace.metadata.stackrox.io/id-;     kubectl label
namespace $namespace namespace.metadata.stackrox.io/name-;     kubectl annotate
namespace $namespace modified-by.stackrox.io/namespace-label-patcher-;   done
```