ポリシーイベント

本文の内容は、Sysdig Secure ポリシーのドキュメント(https://docs.sysdig.com/en/policies.html）を元に日本語に翻訳・再構成した内容となっております。

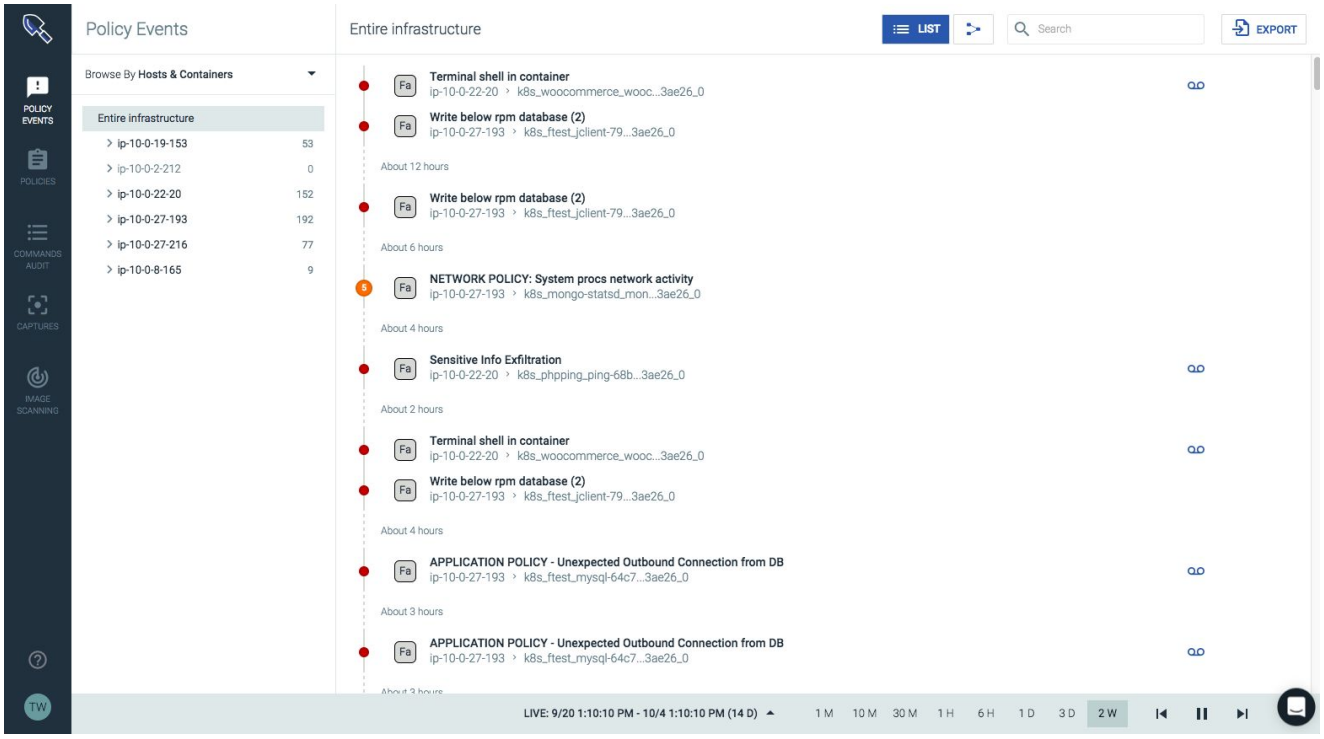
ポリシーイベント

ポリシーイベントモジュールは、定義されたタイムライン中にインフラストラクチャ内で発生したすべてのイベントの完全なリストを表示します。 このモジュールは、ユーザーにインフラストラクチャ全体の概要と、特定のコンポーネントを深く掘り下げ、誤検知を特定し、パフォーマンスを最適化するポリシーを構成する機能を提供します。

ポリシーイベントモジュールをナビゲートする

リストビュー

リストビューには、グループ化/タイムライン内のすべてのイベントの包括的なリストが時系列で表示されます。



注意

複数のイベントが同時に発生する場合、ドットにはイベントの数が含まれます。

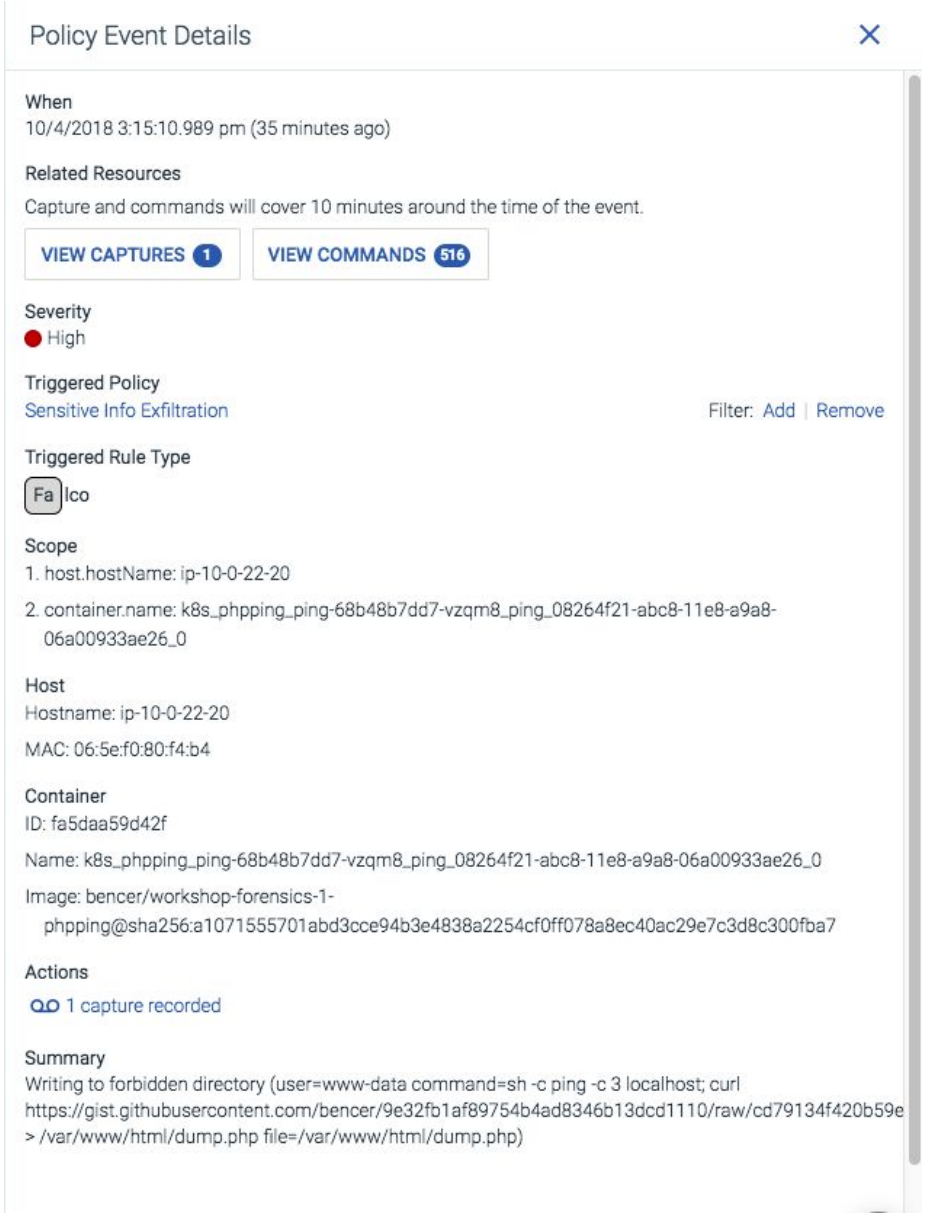
このビューでは、イベントが新しい順に表示され、最新のイベントが上部にリストされます。 次の情報が表示されます。

名前	説明
Severity	トリガーされたポリシーに基づくイベントの重大度。 • 黄色の点は、重大度の低いイベントを示します。 • オレンジ色の点は、重大度が中程度のイベントを示します。 • 赤い点は、重大度の高いイベントを示します。 注意 Sysdig APIを使用してポリシーイベントを処理する場合、重大度レベルの数値に注意してください。低= 1中= 2高= 3

Rule Type	イベントによって違反されたルールのタイプ。各ルールタイプは、周期表スタイルの識別子で表されます。 • Pr: プロセス • Co: コンテナ • Ne: ネットワーク • Fi: ファイルシステム • Sy: Syscall • Fa: Falco
Policy List	イベントによってトリガーされたポリシー。各ポリシーは太字でリストされています。
Entity	イベントの発生元のエンティティ。エンティティには、現在の[`Browse By`]メニューの選択、およびドリルダウンメニューで選択したエントリが反映されます。 **注意** 複数のエンティティが影響を受ける場合、X個のエンティティが関与していることを示す表記が表示されます。Xは影響を受けるエンティティの数を表します。
Action(s) taken	イベントへの対応として実行されたアクション。各アクションはアイコンで表されます： • 一時停止記号は、コンテナが一時停止されたことを示します。ユーザーがdocker unpause操作を実行するまで、コンテナは一時停止したままです。 • 停止記号は、コンテナが停止し、操作を再開しなかったことを示します。 • テープシンボルは、イベントのキャプチャが記録されたことを示します。

イベントの詳細

イベントを選択すると、[Policy Event Details]パネルが開き、イベントの詳細な概要、発生した場所、および違反したポリシーが表示されます。



次の情報が表示されます。

名前	説明
When	イベントが発生した日付と時刻。
Related Resources	以下を含むイベントに関する追加情報： • [`View Captures`]ボタンをクリックすると、[`Captures`]タブが開き、イベントに対して記録されたキャプチャにアクセスできます。 • [`View Commands`]ボタンをクリックすると、[`Commands History`]タブが開き、イベントをトリガーしたコマンドへのアクセスが提供されます。
Severity	トリガーされたポリシーに基づくイベントの重大度。 • 黄色のバーは、重大度が低いことを示します。 • オレンジ色のバーは中程度の重大度を示します。 • 赤いバーは、重大度が高いことを示します。 **注意** Sysdig APIを使用してポリシーイベントを処理する場合、重大度レベルの数値に注意してください。低= 1中= 2高= 3
Triggered Policy	イベントをトリガーしたポリシー。 リンクをクリックすると、[`Policies`]タブが開き、選択したポリシーがデプロイされます。 **注意** 各ポリシーの横にあるフィルターリンクを追加/削除すると、そのポリシーが検索バーに追加/削除されます。
Triggered Rule Type	イベントによって違反されたルールのタイプ。 各ルールタイプは、周期表スタイルの識別子で表されます： • Pr: プロセス • Co: コンテナ • Ne: ネットワーク • Fi: ファイルシステム • Sy: Syscall • Fa: Falco
Scope	インフラストラクチャ内のイベントの範囲 **注意** リストされたエンティティと表示される順序は、[`Browse By`]メニューで選択したグループ化に基づいて異なります。 詳細については、Sysdig Secure

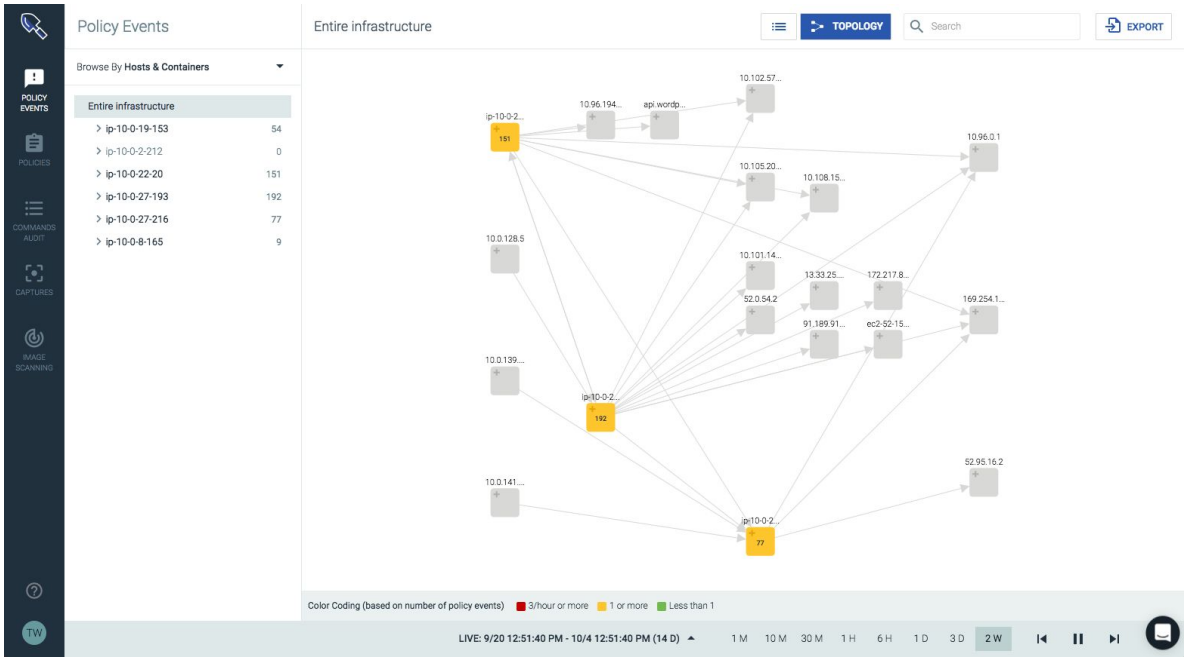
	のドキュメントの「インフラストラクチャ」セクションを参照してください。
Host	イベントが発生したホストのホスト名とMACアドレス。
Container	イベントが発生したコンテナのID、名前、およびイメージ。
Actions	イベントへの対応として実行されたアクション。 各アクションはアイコンで表されます： • 一時停止記号は、コンテナが一時停止されたことを示します。 一時停止すると、ユーザーが「docker unpause」操作を行うまで一時停止したままになります。 • 停止記号は、コンテナが停止し、操作を再開しなかったことを示します。 • テープシンは、イベントのキャプチャが記録されたことを示します。
Summary	イベントに関する詳細情報

トポロジービュー

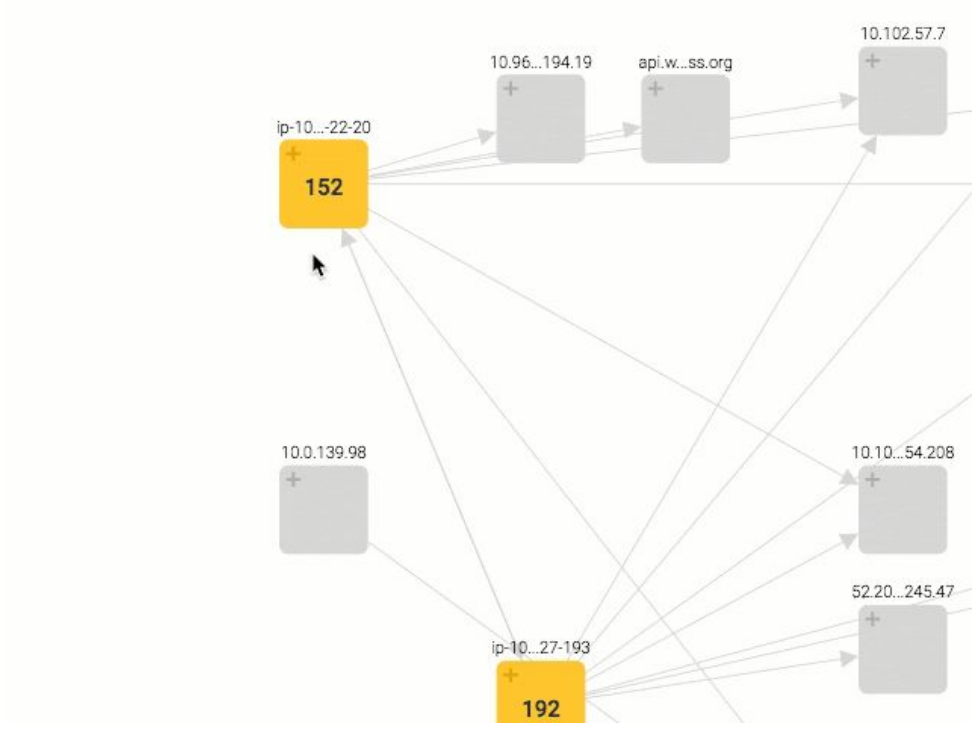
トポロジビューは、構成されたグループ化/タイムラインに基づいて、さまざまなホスト、コンテナ、およびサービス全体のネットワーク依存関係を視覚的に示す、すべてのイベントの概要を提供します。

注意

グループ化と時間間隔の構成の詳細については、「ポリシーイベントのフィルター」セクションを参照してください。



ズームインして、ノードの左上隅にある展開（プラス）アイコンを選択することにより、各ノードをドリルダウンして、レビューを必要とする正確なイベントを見つけることができます。



ポリシーイベントのフィルター処理

グルーピング

グループ化はラベルの階層的な組織であり、ユーザーは論理的な階層でインフラストラクチャビューを整理できます。 ユーザーは、[Browse By]メニューを使用して事前に構成されたグループを切り替えるか、カスタムグループを構成してから、インフラストラクチャの詳細を確認できます。 グループ化の詳細については、Sysdig Secureのグループ化の構成ドキュメントを参照してください。

タイムナビゲーション

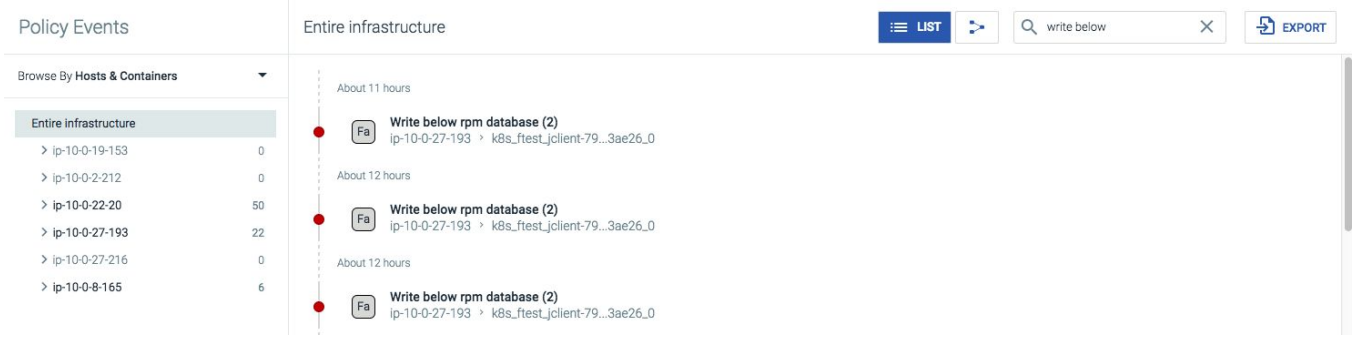
タイムウィンドウナビゲーションバーは、一般的なタイムウィンドウへのクイックリンクをユーザーに提供し、テーブルをフィルタリングして、そのウィンドウ内で実行されたコマンドのみを表示します。 タイムウィンドウの詳細については、タイムウィンドウのドキュメントを参照してください。

注意

Sysdig Secureは現在、カスタム時間枠を設定する機能を提供していません。

検索フィルター

検索フィルターは、検索バーを使用して適用できます。 [Browse By]メニューのグループと一緒にイベント番号が更新され、検索条件を満たすイベントの数が反映されます。 以下の検索バーの例では、`Write below rpm database`のみが表示されます。



注意

トポロジビューは、検索機能の影響を受けません。

Sysdig Secureでグループ化を構成する

グループ化はラベルの階層的な組織であり、ユーザーは論理的な階層でインフラストラクチャビューを整理できます。 ユーザーは、[Browse By]メニューを使用して事前に構成されたグループを切り替えるか、カスタムグループを構成してから、インフラストラクチャの詳細を確認できます。

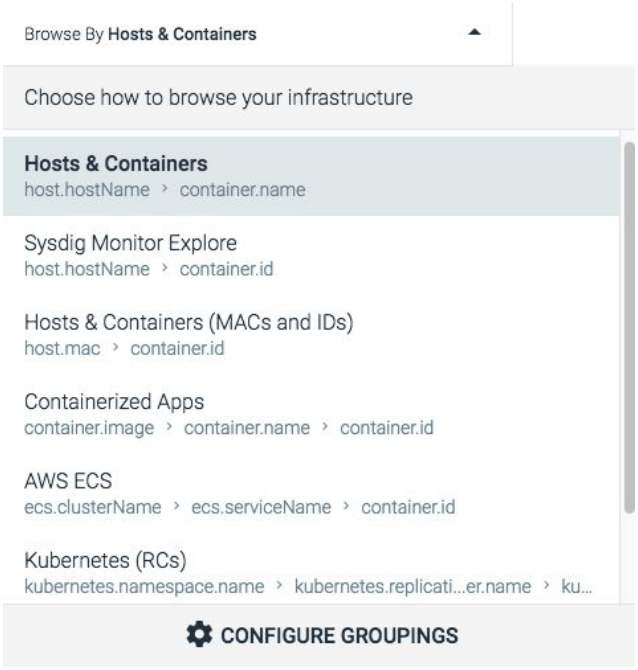
注意

グループ化に関するより一般的な情報については、グループ化、スコーピング、およびセグメント化メトリクスのドキュメントを参照してください。

スイッチグループ

新しいグループに切り替えるには：

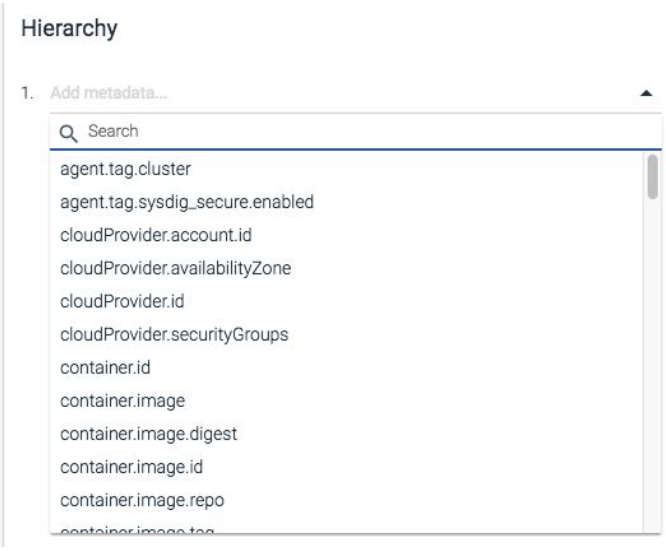
1. Commands Auditモジュールから、[Browse By]メニューを開き、リストからグループを選択します。



新しいグループを作成する

新しいグループを作成するには：

1. [`Commands Audit`]モジュールから[`Browse By`]メニューを開き、[`Configure Groupings`]をクリックして`Groupings Editor`を開きます。
2. [`Create New Grouping`]リンクをクリックします。
3. 第1レベルのグループ化ドロップダウンメニューを開きます



4. 目的の最上位ラベルを選択するか、スクロールまたは検索バーを使用して検索し、選択します。

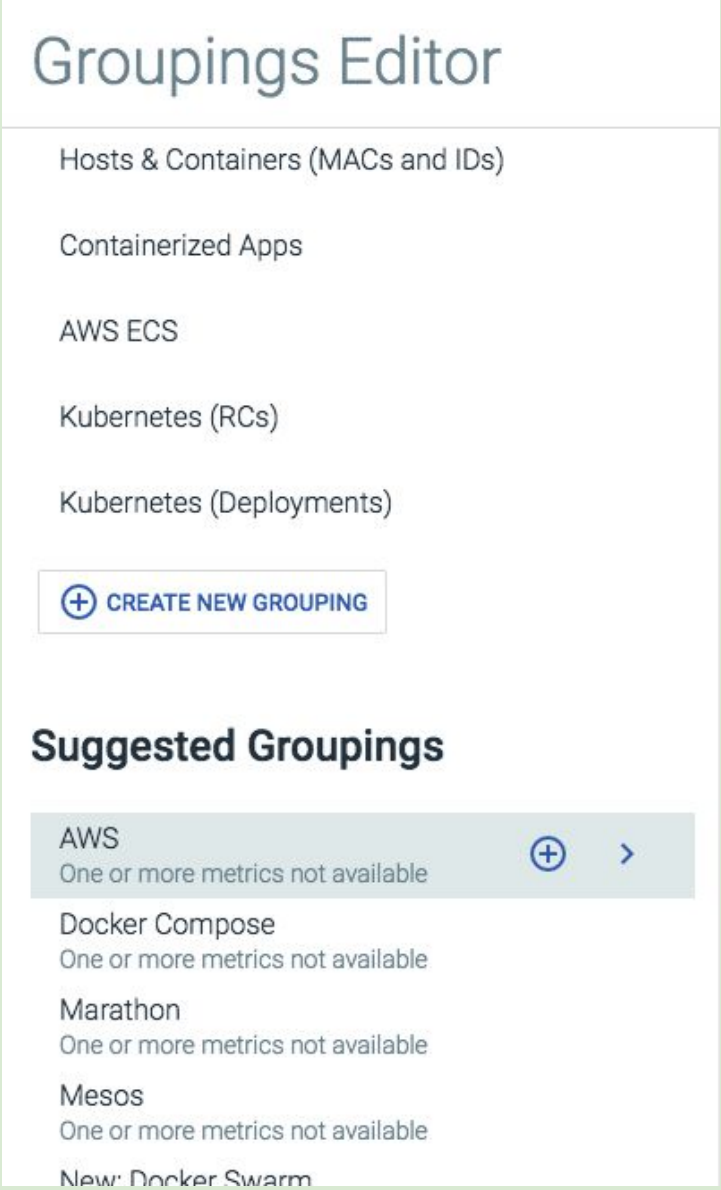
5. オプション：グループレベルを追加するには、新しい空白のドロップダウンメニューを開き、手順3を繰り返します。

> **注意**
>
> インフラストラクチャラベル階層で使用可能なレイヤーがなくなるまで、手順5を繰り返します。

6. [`Save`]ボタンをクリックして、グループを保存します。

注意

Sysdig Secureにはいくつかの推奨グループがあり、それらは事前に構成されていますが、デフォルトリストには含まれていません。 これらは、グループ名の横にある追加（プラス）ボタンをクリックして追加できます。



既存のグループの名前を変更する

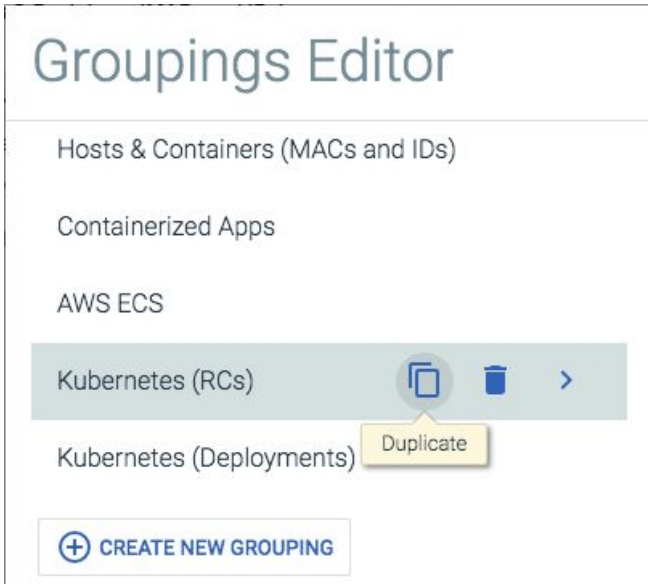
グループの名前を変更するには：

1. [Commands Audit]モジュールから[rowse By]メニューを開き、[Configure Groupings]をクリックしてグループ化エディターを開きます。
2. リストからグループを選択します。
3. [Edit]パネルでグループの名前を編集し、[Save]ボタンをクリックします。

既存のグループ化を複製する

グループ化の複製を作成するには：

1. [Commands Audit]モジュールから[Browse By]メニューを開き、[Configure Groupings]をクリックしてGroupings Editorを開きます。
2. 関連するグループの横にあるDuplicate（ページ）アイコンをクリックして複製します。



3. 新しいグループ化を構成し、[Save]ボタンをクリックします。

グループを削除する

グループ化を削除するには：

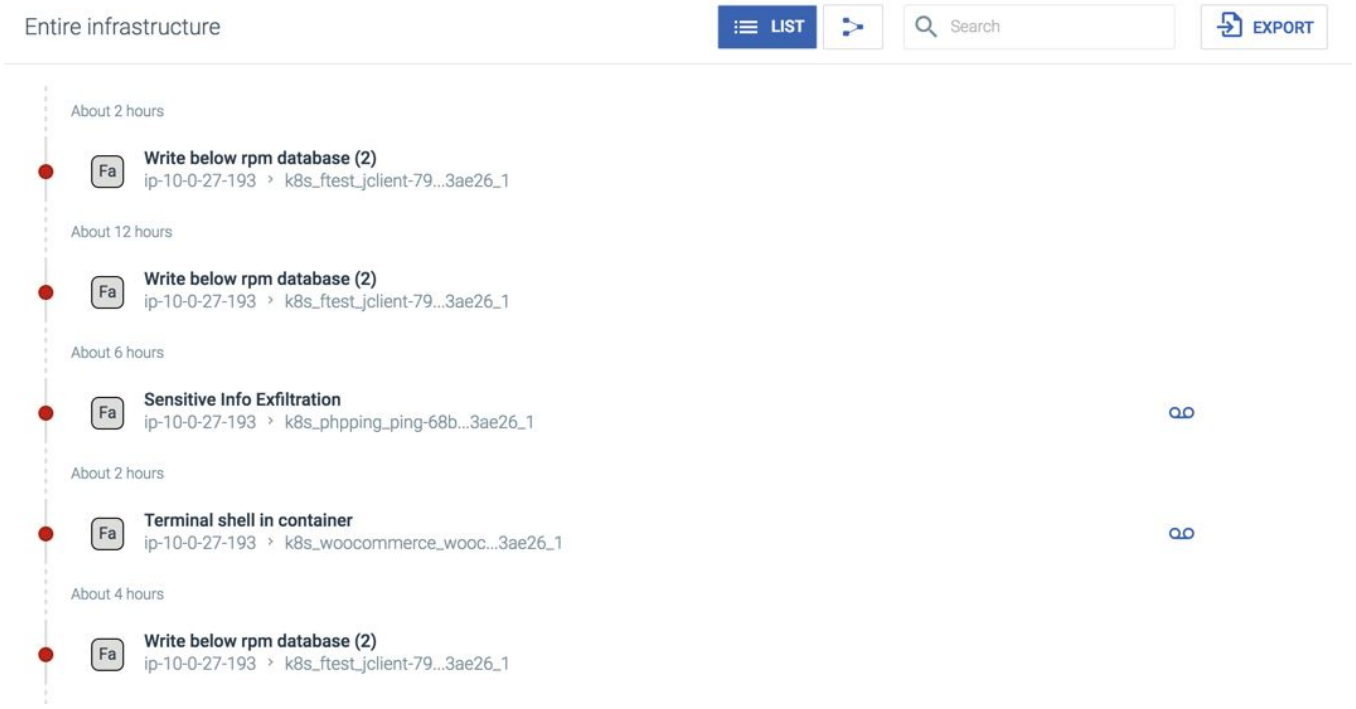
1. [ommands Audit]モジュールから[Browse By]メニューを開き、[Configure Groupings]をクリックしてグループ化エディターを開きます。

2. [`Delete`]（ゴミ箱）アイコンをクリックして、グループを削除します。

3. [Save]ボタンをクリックして、変更を確認します。

一般的なポリシーイベントの特定

最も頻繁に発生するイベントに最初に対処する必要があります。 これは、最も一般的な発生のリストを確認することにより、ポリシーイベントモジュールから実行できます。



より詳細なビューについては、Sysdigは、直接ポリシーイベントストリームにアクセスするためのpython-sdc-client(https://github.com/draios/python-sdc-client)およびSysdig Secure APIを提供します。サンプルプログラムget_secure_policy_events.pyは、-summarize引数を使用して、頻度順に並べられたポリシーイベントの要約ビューを提供します。 --summarizeは、同様のイベントをより適切に集約できるように、出力文字列からコンテナ情報を削除します。

注意

Sysdigはトラブルシューティングのために UIに直接アクセスできないため、Sysdigは、python-sdc-clientの使用も推奨しています。特にget_secure_policy_events.pyは、オンプレミスのお客様のポリシーイベントを収集するために、プログラムをファイルに書き込み、フィードバックとして渡すことができます。 python-sdc-clientの詳細については、Sysdig Cloud Python Script Libraryのドキュメントを参照してください。

出力例を以下に示します。

```
user@host:~$ python examples/get_secure_policy_events.py --help
usage: examples/get_secure_policy_events.py [-s|--summarize] [-l|--limit <limit>] <sysdig-token> [<duration sec>|<from sec> <to sec>]
-s|--summarize: group policy events by sanitized output and print by frequency
-l|--limit: with -s, only print the first <limit> outputs
You can find your token at https://secure.sysdig.com/#/settings/user

user@host:~$ python examples/get_secure_policy_events.py --summarize 5b83272d-6e3f-44b3-b3b8-9dd8671f98b7 604800
56 Database-related program spawned process other than itself (user=root program=sh -c ls > /dev/null parent=mysqld)
24 Sensitive file opened for reading by non-trusted program (user=root name=ftest command=ftest -i 25200 -a exfiltration file=/etc/shadow
parent=docker-containe gparent=docker-containe ggparent=dockerd gggparent=systemd)
14 Rpm database opened for writing by a non-rpm program (command=ftest -i 43200 -a write_rpm_database
file=/var/lib/rpm/created-by-event-generator-sh)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34870)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34871)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34872)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34869)
1 A shell was spawned in a container with an attached terminal (user=root  shell=bash parent=exe cmdline=bash  terminal=34816)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34867)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34868)
1 A shell was spawned in a container with an attached terminal (user=root  shell=sh parent=exe cmdline=sh  terminal=34866)
```

ポリシーの一意性を決定する

多くの場合、ポリシーイベントは多くの環境で使用される一般的なソフトウェアの使用に関連しているため、複数の顧客が同じイベントを見ることになります。ポリシーイベントが他のお客様に発生する可能性がある場合は、デフォルトのFalcoルールを更新できるように、Sysdigサポートにお問い合わせください。

誤検知のためにFalcoルールを設定する

環境にローカルなポリシーイベントの場合、それらに対処する最初の手順は、`Policies`モジュールの[`Rules Editor`]タブでカスタムFalcoルールを追加することです。これらの追加は、誤検知に対処するために、デフォルトルールセクションのリスト、マクロ、またはルールの動作を拡張または上書きします。

多くのルールにはプレフィックスuser_のマクロがあり、ルールの動作を変更するためにカスタムルールセクションでオーバーライドできます。以下に例を示します。

- user_known_write_etc_conditions
- user_read_sensitive_file_conditions
- user_known_change_thread_namespace_binaries
- user_shell_container_exclusions

- user_trusted_containers
- user_sensitive_mount_containers

user_マクロが使用できない場合、Sysdigは、既存のリスト、マクロ、またはルールに上書きするのではなく、変更を追加することをお勧めします。

警告

上書きされたリスト、マクロ、およびルールは、デフォルトのルールが更新されても静的のままです。 これにより、新しいデフォルトルールがシャドウされ、ユーザーがアクセスできなくなる可能性があります。

Falcoルールに必要な特定の変更は、誤検知をトリガーしたポリシーに大きく依存します。いくつかの例を以下に示します。

- `Write below etc`

  Write Below etcポリシーの場合、/ etcの下に特定のファイルを書き込む追加プログラムがある場合があります（この例では、プログラムcatsdとファイル/etc/catfood.cfgを使用します）。マクロは、次のように定義できます。

  ```
  - macro: catsd_writing_catfood_cfg
    condition: (proc.name=catsd and fd.name=/etc/catfood.cfg)
  ```

  さらに、マクロuser_known_write_etc_conditionsを上書きして、新しい例外を追加する必要があります。

  ```
  - macro: user_known_write_etc_conditions
    condition: catsd_writing_catfood_cfg
  ```

- `Launch Privileged Container`

  `Launch Privileged Container`ポリシーには、特権で実行する必要がある特定のコンテナが存在する場合があります。以下のサンプルマクロは、registry.customer.com / mydatastoreで始まるイメージを使用し、user_trusted_containersマクロをオーバーライドします。

  ```
  - macro: user_trusted_containers
    condition: (container.image startswith registry.customer.com/mydatastore)
  ```

- `Run shelluntrusted`

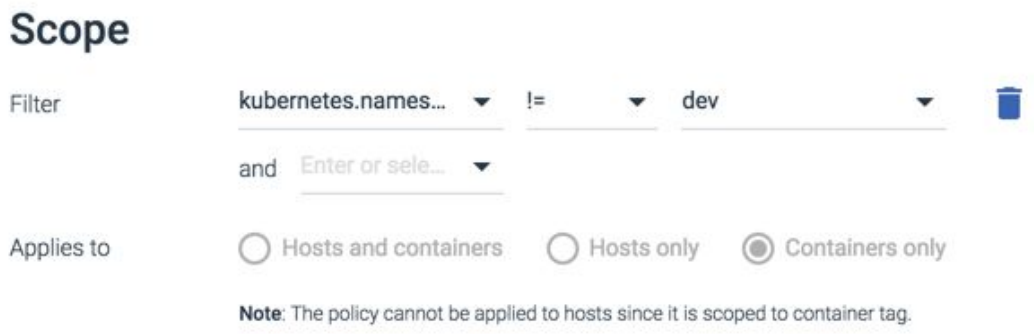
  `Run shelluntrusted`ポリシーの場合、正当な目的で環境内にシェルを生成するプログラムが存在する場合があります。以下の例では、user_known_shell_spawn_binariesリストを使用して、既知のシェルスポーンバイナリのリストにphpを追加します。

  ```
  - list: user_known_shell_spawn_binaries
    append: true
    items: [php]
  ```

  Falcoルールの作成の詳細については、Falcoルールのドキュメントを参照してください。

誤検知のためのポリシースコープを設定する

Falcoルールの変更が誤検知に対処できない場合、ポリシーのスコープを変更して、発生する/発生しない環境の部分に焦点を合わせることができます。たとえば、多くのシェルを生成するdev NamespaceのKubernetesの下で実行されるdev環境があります。この場合、スコープをEntire Infrastructureからkubernetes.namespace.name！= "dev"に変更することにより、ポリシーを変更してスコープからdev Namespaceを除外できます。

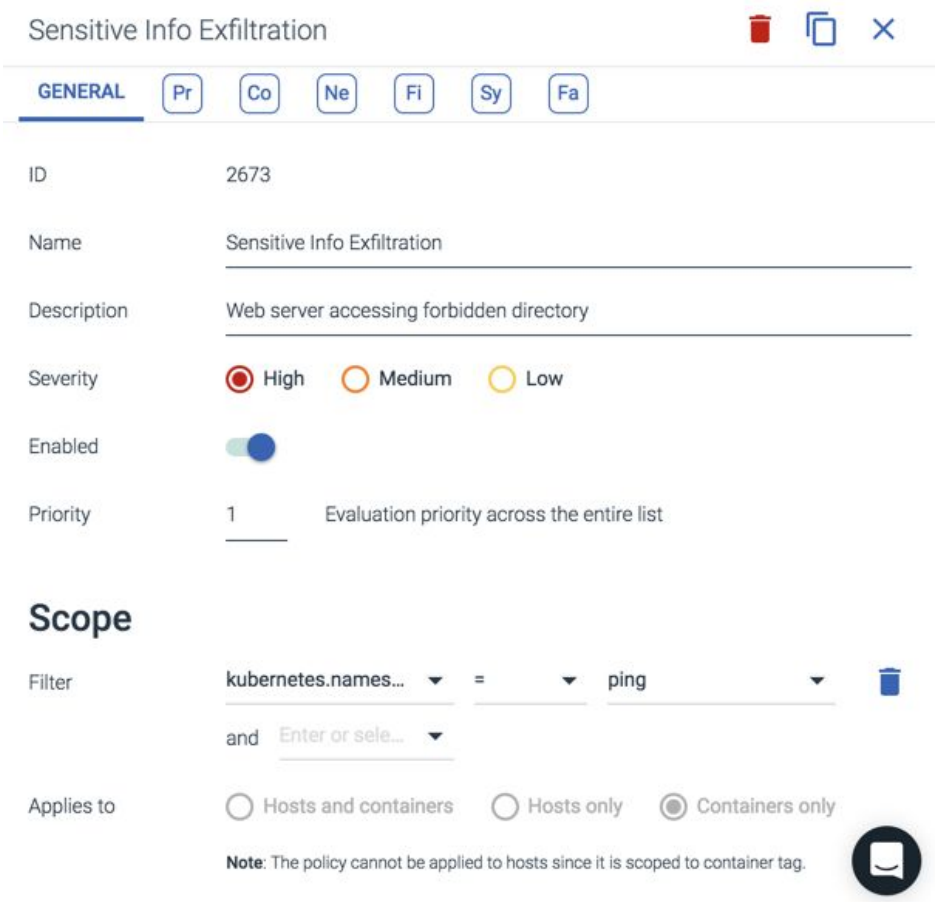


ポリシーを無効にする

上記のアクションのいずれも機能しない場合、ポリシーを無効にすることができます。 ポリシーを無効にするには：

1. [`Policies`]モジュールから、関連するポリシーを選択します。

2. `Enabled`スイッチを切り替えて、ポリシーを無効にします。



3. [`Apply Changes`]ボタンをクリックします。

ウォークスルー例

以下のウォークスルーの例では、ポリシーと基になるルールを変更してコンテナをホワイトリストに登録し、誤検知の結果を最小限に抑えます。

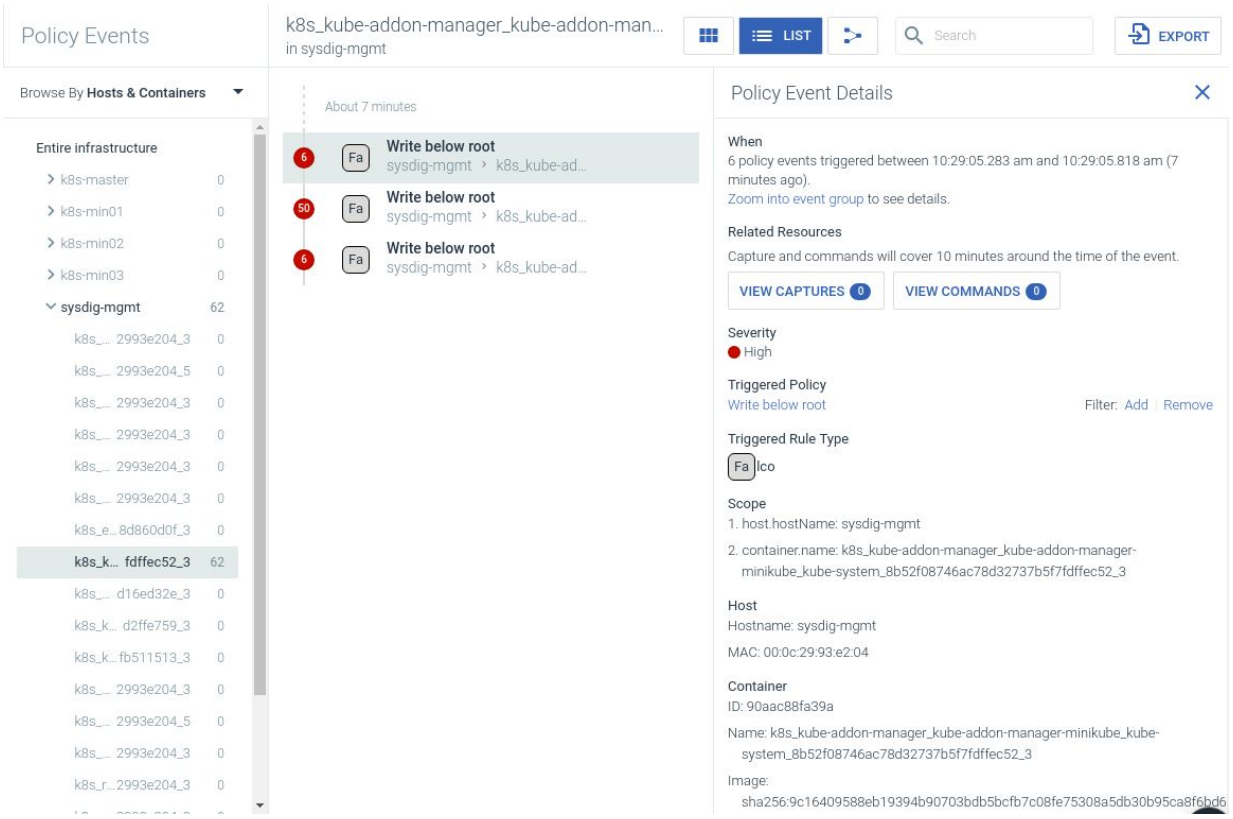
ホワイトリストに入れるコンテナ/イメージを特定する

コンテナ/イメージは、Sysdig Secure UI（イベントが既にトリガーされている場合）から、またはホストから直接識別できます。

Sysdig Secure UIから

コンテナ/イメージを見つけるには：

1. [`Policy Events`]モジュールから、左側のドリルダウンメニューを使用して、ホワイトリストに登録するオブジェクトを特定します。 この例では、目的のオブジェクトは組み込みKubernetesコンポーネント（k8s_kube-addon-manager）の1つです。

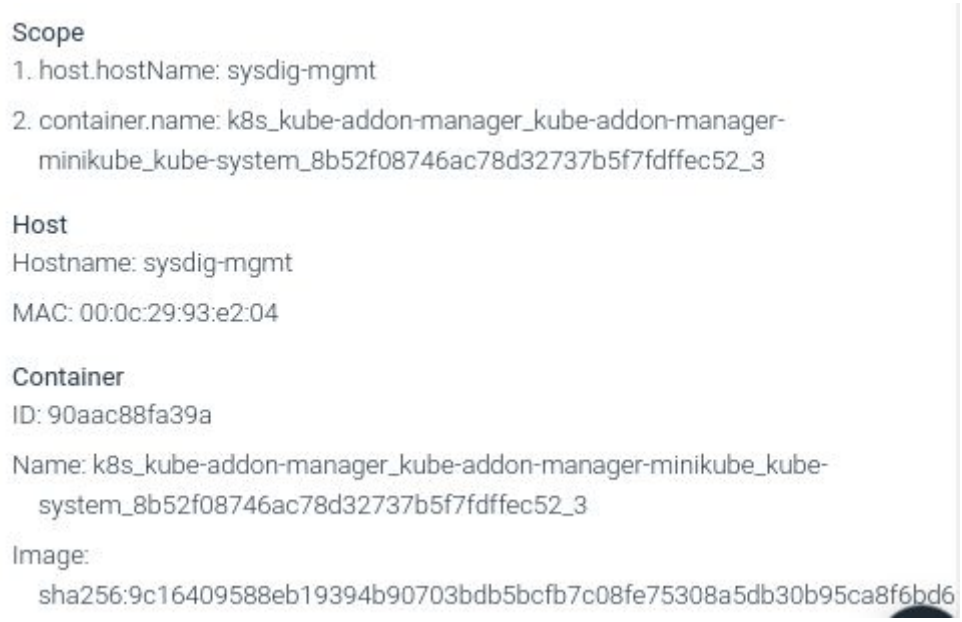


注意

すべての情報は[`Policy Event Details`]ウィンドウに含まれているため、ドリルダウンメニューを使用する必要はありません。 ただし、ドリルダウンメニューを使用すると、ホワイトリストに登録する正しいオブジェクトを確認できます。

2. 関連するイベントをクリックして[`Policy Event Details`]ウィンドウを開き、イベントの詳細を確認し、ホワイトリストに登録するオブジェクトを特定します。 この例では、イメージは

Kubernetesの内部コンポーネントであるため、`sha256 diges`です。したがって、コンポーネントが後日アップグレードされる場合は、代わりにコンテナ名を使用する必要があります。



注意

ホワイトリストにはさまざまなメタデータを使用できます。コンテナの場合、コンテナ名、イメージ名、またはダイジェストを使用できます。コンテナ名が最も一般的なアプローチですが、このオプションはスプーフィングされる可能性があります。イメージ名（sysdig / agentなど）はより安全なオプションです。ソースは保証されているため、ダイジェストの使用が最も安全なオプションですが、コンポーネントの更新後もプラットフォームがオブジェクトを正しくホワイトリストに登録し続けることを保証する自動化ソリューションが必要になる場合があります。

3. オブジェクトによってトリガーされるポリシールールの名前を特定します。

ホストCLIから

コンテナ/イメージを見つけるには：

1. ホスト上のターミナルで、次のコマンドを実行します。

> **注意**
>
> 以下のコマンド例では、docker psの出力を制限して、関連する情報のみを提供しています。これは、コンテナ名またはソースイメージのみを検索しているためです。

```
user@host:~$ docker ps --format "table {{.Names}}\t{{.Image}}"
```

> **注意**
>
> リストが大きい場合は、more | lessまたはgrepを使用して特定のオブジェクトを見つけます。 以下の出力例では、目的のイメージがk8s_kube-addon-managerであるため、grepを使用してアドオンを検索しています。 コマンドには、表の見出しも含まれます。

```
user@host:~$ docker ps --format "table {{.Names}}\t{{.Image}}" | grep 'addon\|NAME' |
more
NAMES
IMAGE
k8s_kube-addon-manager_kube-addon-manager-minikube_kube-system_8b52f08746ac78d32737b5f7f
dffec52_3               9c16409588eb
k8s_POD_kube-addon-manager-minikube_kube-system_8b52f08746ac78d32737b5f7fdffec52_3
k8s.gcr.io/pause-amd64:3.1
```

2. 違反したポリシールールを特定します。

> **注意**
>
> これは、Sysdig Secure UIを介してのみ実行できます。

ホワイトリストルールを追加する

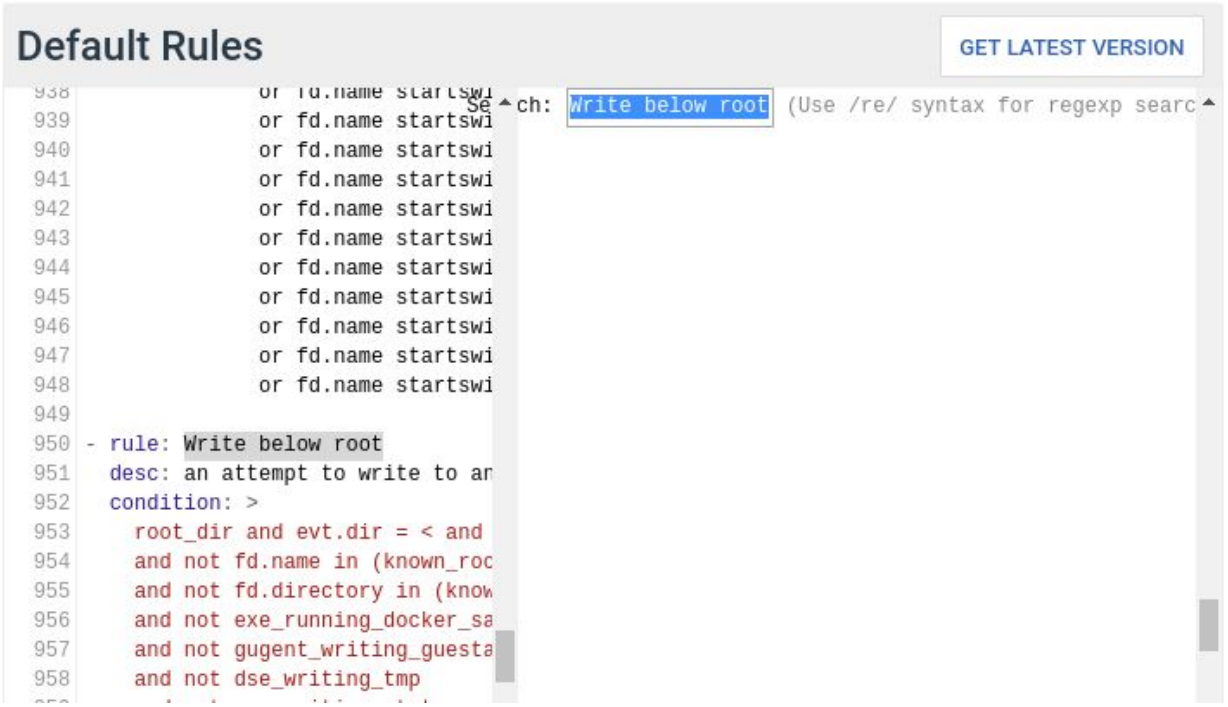
Sysdig Secure内のデフォルトのFalcoルールは読み取り専用です。 ルールを変更するには、カスタムルールとして書き換えます。

1. [Policies]モジュールから、[Rules Editor]タブに移動します。
2. 新しいマクロを作成するか、既存のマクロを編集して、マクロを定義します。

> **注意**
>
> この例では、既存のマクロを編集します。プロセスは新しいマクロを作成する場合と同じですが、一意の名前が必要です。

A. [Default Rules]セクションで既存のマクロを見つけて、[Custom Rules]セクションにコピーします
B. ホワイトリストに登録するオブジェクトを定義する条件をマクロに追加します。

3. 既存のデフォルトルールを見つけて、user_trustedコンテナマクロを使用しているかどうかを判断します。



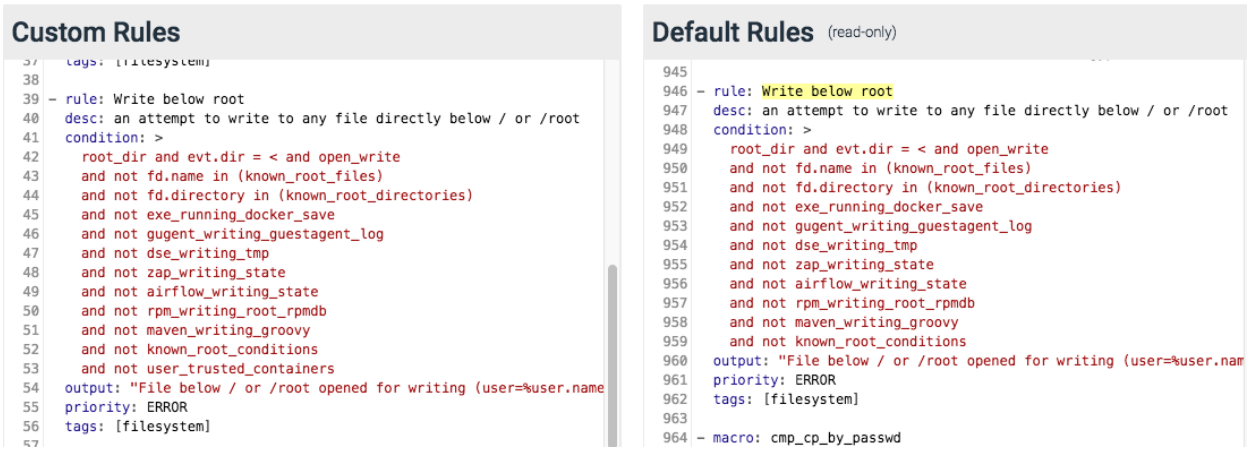

```
- rule: Write below root
  desc: an attempt to write to any file directly below / or /root
  condition: >
    root_dir and evt.dir = < and open_write
    and not fd.name in (known_root_files)
    and not fd.directory in (known_root_directories)
    and not exe_running_docker_save
    and not gugent_writing_guestagent_log
    and not dse_writing_tmp
    and not zap_writing_state
    and not airflow_writing_state
    and not rpm_writing_root_rpmdb
    and not maven_writing_groovy
    and not known_root_conditions
```

注意

この例では、「Write below root」はuser_trusted_containersマクロを使用しません。

4. ルールを[Custom Rules]セクションにコピーします。

5. 構成されたマクロをルールに追加します。

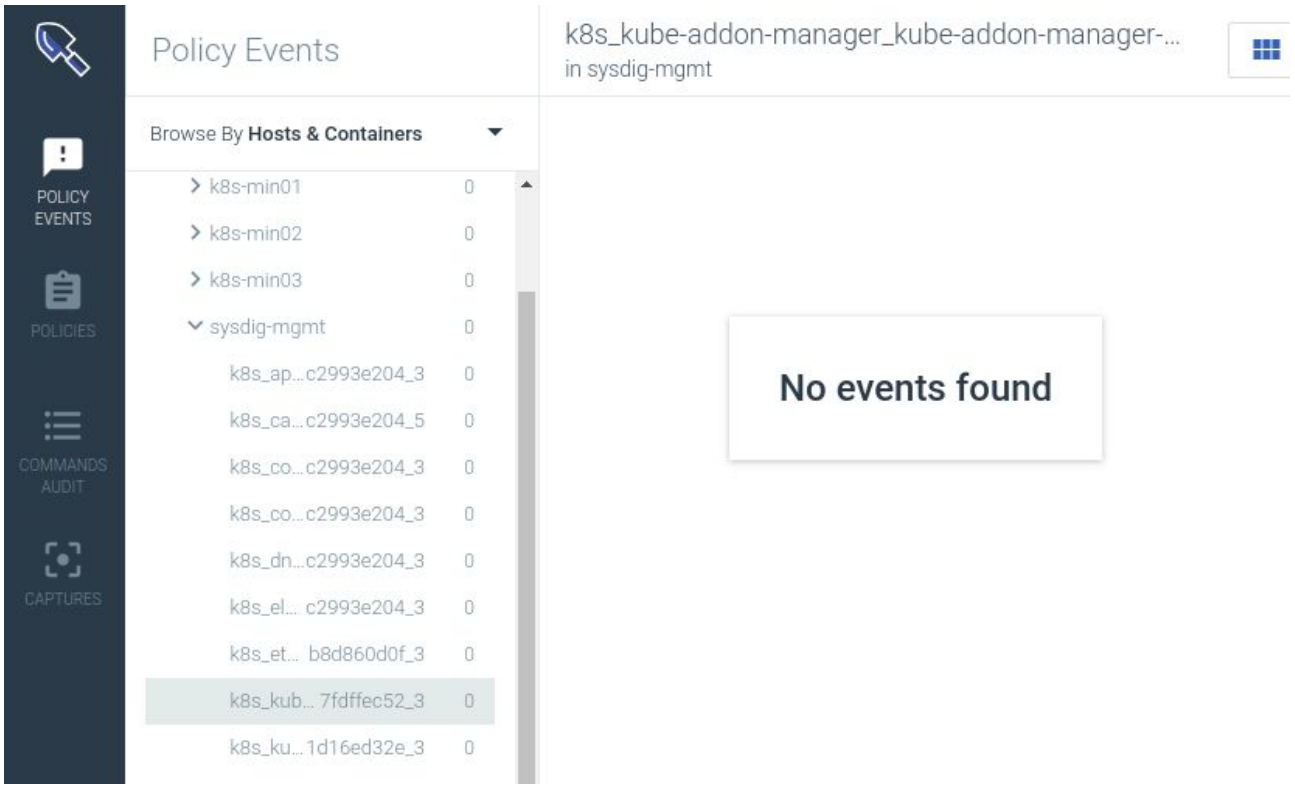


6. [Save]ボタンをクリックして、変更を保存します。

更新されたルールは、次にSysdig Agentがバックエンドにチェックインするときに使用可能になります（通常は10秒ごとに）。

ポリシーイベントがクリアされていることを確認する

ポリシーが適用され、イベントの通知が消去されると、ホワイトリストに登録されたオブジェクトには、イベントが見つからないか、このポリシーがトリガーされなくなったことが表示されます。



イベント転送

Sysdigは、Splunk、Elastic Stack、Qradar、Arcsight、LogDNAなどのサードパーティSIEM（セキュリティ情報およびイベント管理）プラットフォームおよびロギングツールへのポリシーイベントの送信をサポートしています。Sysdig Secureのイベント転送機能を使用して、環境内のツールにイベントを転送できます。そうすることで、セキュリティイベントを表示し、Sysdigの結果を、既にイベント分析に使用しているツールと関連付けることができます。

イベント転送の種類

- スプランク
- syslog

Syslogとの統合

Syslogは、システムロギングプロトコルを指します。これは、主にネットワークデバイスで使用され、特定の形式でイベントとログを中央システムに送信して保存および分析するために使用される標準です。 Syslogイベントには、重大度レベル、ホストIP、タイムスタンプ、診断情報などが含まれます。

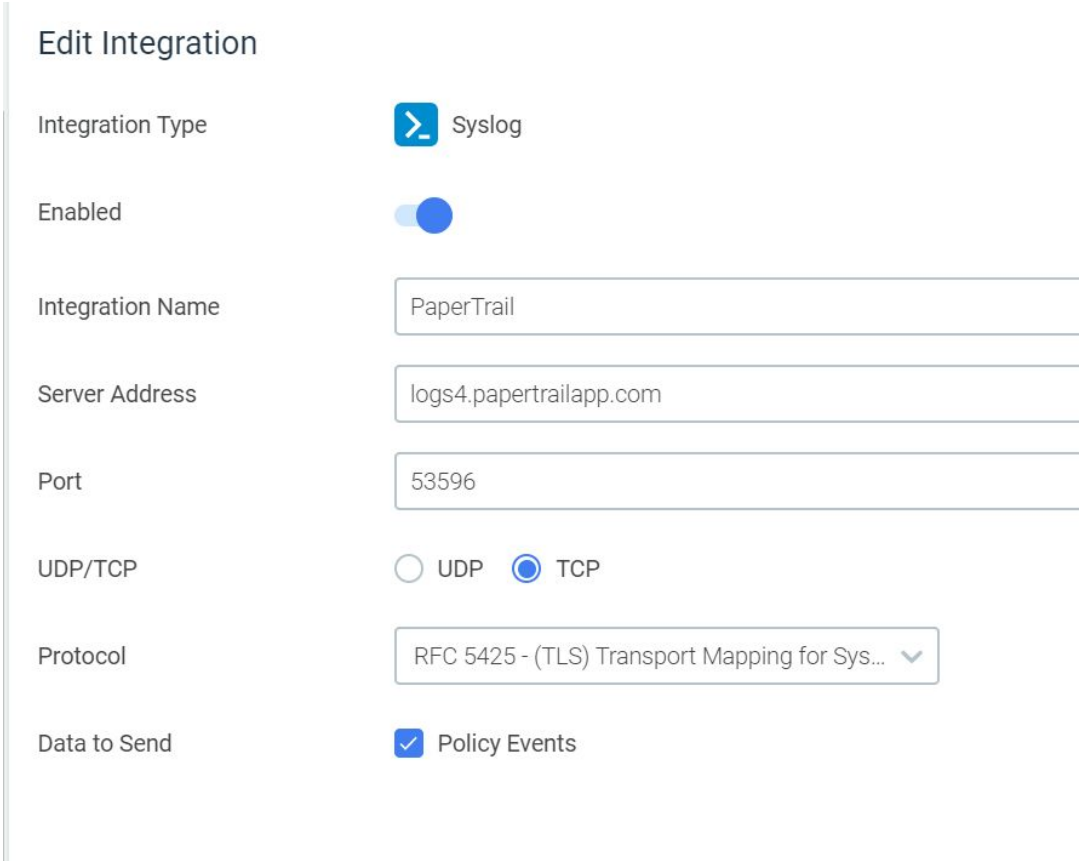
Sysdig Event Forwardingを使用すると、Sysdig Secureによって収集されたポリシーイベントをSyslogサーバーに送信できます。

Syslogイベント転送を設定する

イベントデータをSyslogサーバーに転送するには：

1. Sysdig Secure UIの[`Settings`]モジュールから、[`Events Forwarding`]タブに移動します。
2. [`Add Integration`]ボタンをクリックします。
3. ドロップダウンメニューから[Syslog]を選択します。
4. 必要に応じて、Enabledスイッチを切り替えます。デフォルトでは、新しい統合は有効になっています。

5. 必要なオプションを設定します。



a. Integration Name：統合名を定義します。
b. **Server Address**：イベントの転送先のSyslogサーバーを指定します。ドメイン名またはIPアドレスを入力します。ドメイン名が複数のIPアドレスに解決される場合、最初に解決されたアドレスが使用されます。
c. **Port**：ポート番号を指定します。
d. プロトコルを指定します。

   UDP：SyslogリスナーがSyslogデータを受信するために使用するプロトコル。デフォルトのポートは514です。ただし、Syslogサーバーは任意のポートを使用するように構成できます。

   TCP：ネットワークの輻輳の処理とパケット損失の防止の点でUDPよりもはるかに信頼性が高いため、セキュリティインシデントにはTCPを使用します。 TCP接続にもポート514を使用します。ポート514のTCPは、データ転送の信頼性を高めるために使用されます。 RFC 5425仕様のデフォルトポートは6514です。

e. 送信するデータのタイプを指定します。

RFC 3164はプロトコルの古いバージョンですが、現在のプロトコルはRFC 5424です。RFC5425は、暗号化されたチャネルを使用するためのRFC 5424の拡張です。ログを送信するサーバーに応じてプロトコルを選択します。

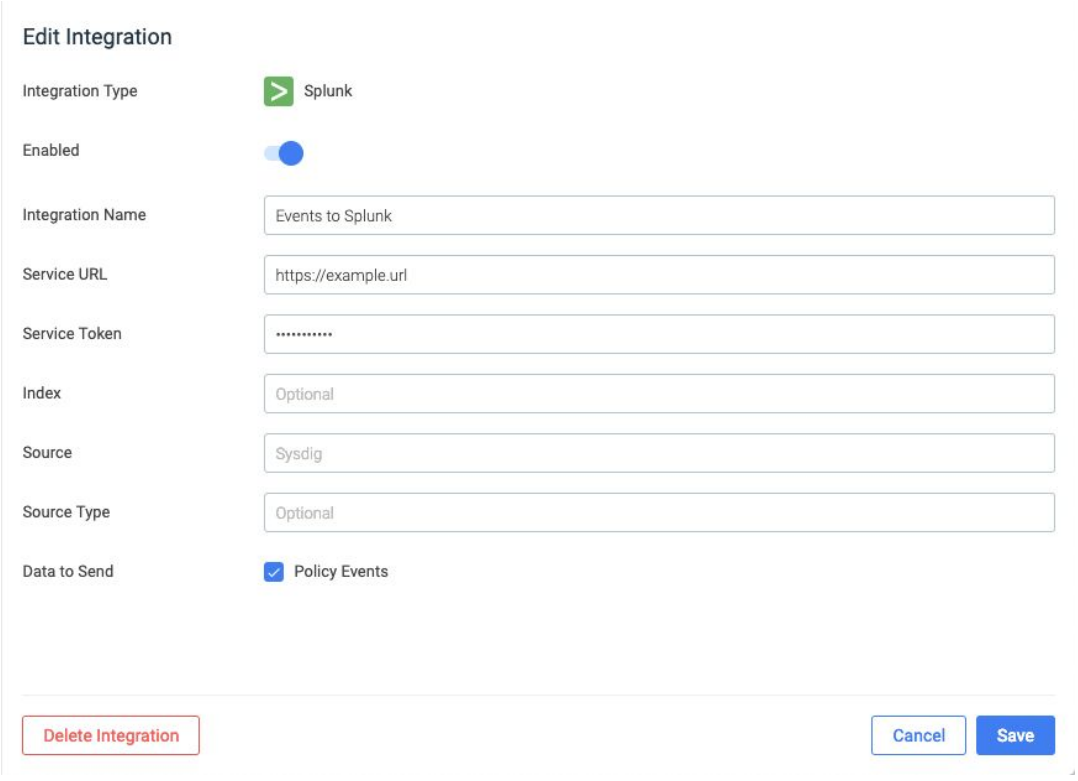
6. [`Save`]ボタンをクリックして、統合を保存します。

Splunkとの統合

Splunkイベント転送を構成する

イベントデータをSplunkに転送するには：

1. Sysdig Secure UIの[`Settings`]モジュールから、[`Events Forwarding`]タブに移動します。
2. [`Add Integration`]ボタンをクリックします。
3. ドロップダウンメニューから[`Splunk`]を選択します。
4. 必要に応じて、Enabledスイッチを切り替えます。 デフォルトでは、新しい統合は有効になっています。
5. 必要なオプションを設定します。

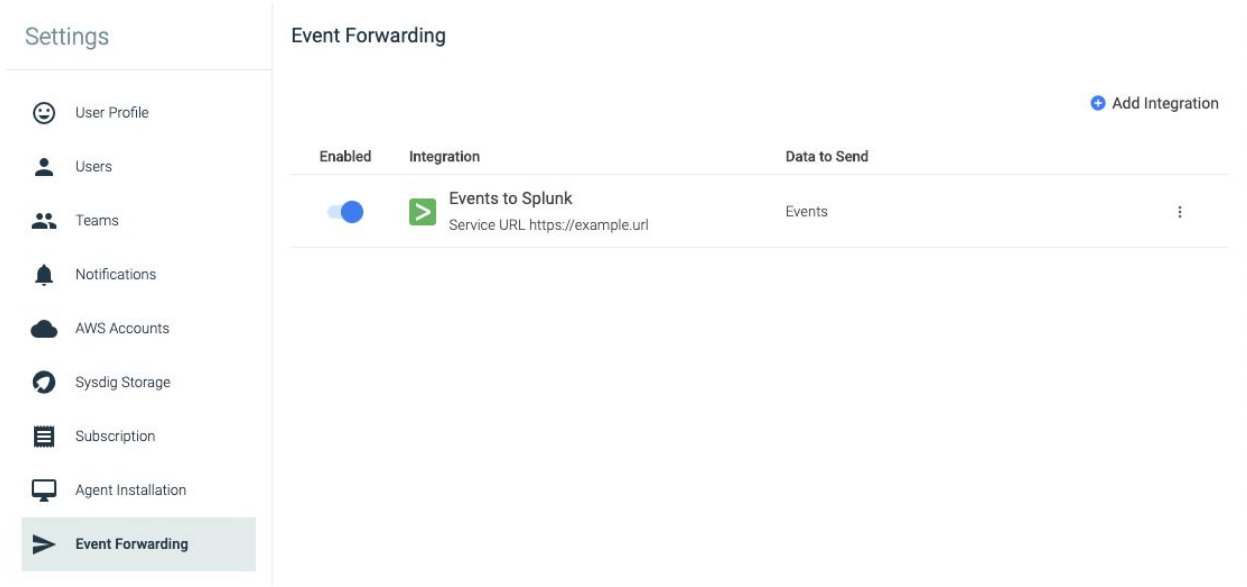


a. 統合名を定義します。

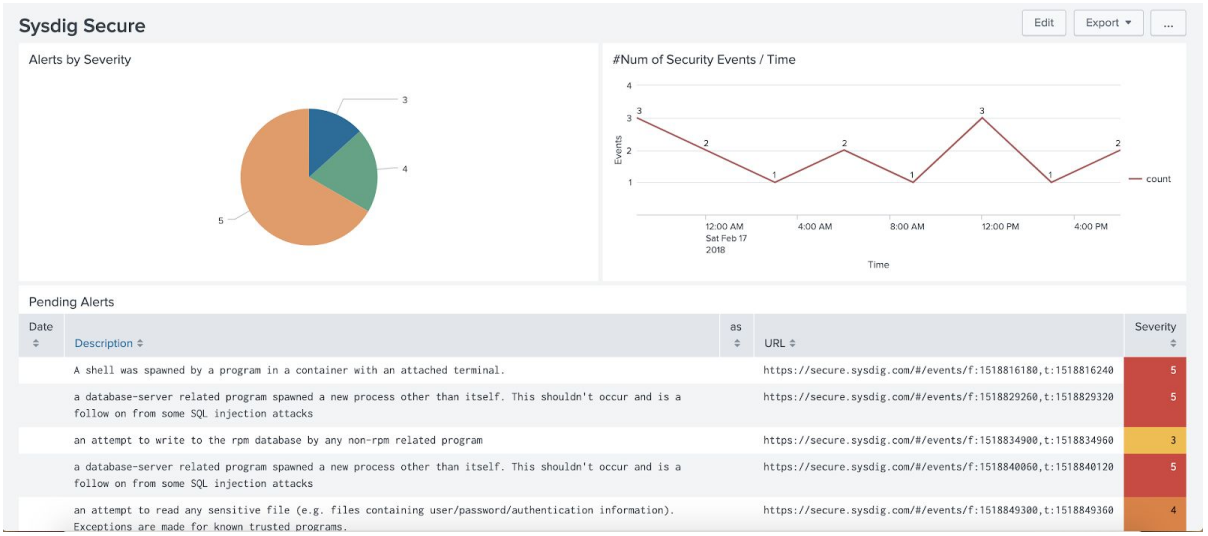
b. SplunkサービスのURLを定義します。 これは、イベントをSplunkデプロイメントに転送するHTTPイベントコレクターです。 ポートとともにサービスURLを入力してください。 形式は、scheme:// host:portです。
c. Splunkサービストークンを定義します。 これは、SysdigがHTTP Event Collectorへの接続を認証するために使用するトークンです。
d. **Optional**：必要に応じて追加のSplunkパラメーター（インデックス、ソース、ソースタイプ）を構成します。
e. **Index**：イベントが保存されるインデックス。 HTTP Event Collectorの構成中にインデックスを選択した場合は、インデックスを指定します。
f. **Source**：ソース、つまりSysdig Monitorからのイベントのソースキー/フィールドを設定します。
g. **Source Type**：イベントのデータ構造を識別します。 詳細については、ソースタイプを参照してください。

これらのパラメーターの詳細については、Splunkのドキュメントを参照してください。

6. Splunkに送信するデータを選択します。 現在、Sysdigはポリシーイベントの送信のみをサポートしています。
7. [`Save`]ボタンをクリックして、統合を保存します。



Sysdig Secureから転送されたポリシーイベントがSplunkに表示される方法の例を次に示します。



エージェントラベルを使用したイベントの強化

デフォルトのラベル

イベントラベルを有効にすると、エージェントはデフォルトでこれらのラベルを含めます

```
event_labels:
  include:
    - process.name
    - host.hostName
    - agent.tag
    - container.name
    - kubernetes.cluster.name
    - kubernetes.namespace.name
    - kubernetes.deployment.name
    - kubernetes.pod.name
    - kubernetes.node.name
```

カスタムラベルの追加

イベントラベルには、イベントラベルを含めることも除外することもできます。

```
event_labels:
  exclude:
    - custom.label.to.exclude
```

```
event_labels:
  include:
    - custom.label.to.include
```

強化されたイベントがsplunkに送信される例

```
{ [-]
baselineId: null
containerId: e4d32e56d9d2
description: A shell was used as the entrypoint/exec point into a container with an
attached terminal.
eventLabels: [ [-]
{ [-]
key: kubernetes.node.name
value: ip-172-31-72-246
}
{ [-]
key: container.name
value:
k8s_elasticsearch_sysdigcloud-elasticsearch-0_sysdigcloud_c824e1f8-aa1f-11e9-aff4-027768606
bae_0
}
{ [-]
key: kubernetes.cluster.name
value: SysdigBackend
}
{ [-]
key: kubernetes.pod.name
value: sysdigcloud-elasticsearch-0
}
{ [-]
key: kubernetes.namespace.name
value: sysdigcloud
}
{ [-]
key: agent.tag.timezone
value: UTC
}
{ [-]
```

```
key: agent.tag.location
value: europe
}
{ [-]
key: process.name
value: bash
}
{ [-]
key: host.hostName
value: ip-172-31-72-246
}
]
falsePositive: false
fields: [ [+]
]
hostMac: 02:77:68:60:6b:ae
id: 702701271278202880
isAggregated: false
matchedOnDefault: false
name: Terminal shell in container
output: A shell was spawned in a container with an attached terminal (user=root
k8s_elasticsearch_sysdigcloud-elasticsearch-0_sysdigcloud_c824e1f8-aa1f-11e9-aff4-027768606
bae_0 (id=e4d32e56d9d2) shell=bash parent=docker-runc cmdline=bash terminal=34816)
policyId: 18
ruleSubtype: null
ruleType: RULE_TYPE_FALCO
severity: 5
timestamp: 1564065391633554
version: 1
}
```


イベント転送統合を削除する

既存の統合を削除するには：

1. Sysdig Secure UIの[`Settings`]モジュールから、[`Events Forwarding`]タブに移動します。
2. [`More Options`]（3つのドット）アイコンをクリックします。
3. [`Delete Integration`]ボタンをクリックします。

4. [`Yes, delete`]ボタンをクリックして、変更を確認します。

Kubernetes監査ログ

Kubernetesログ統合により、Sysdig Secureは、Falcoルール、アクティビティ監査にKubernetes Auditログデータを使用し、Podセキュリティポリシーの影響をテストできます。以下にリストされているディストリビューションとプラットフォームの例を提供します。

統合により、次の監査が可能になります。

- ポッド、サービス、デプロイメント、デーモンセットなどの作成と破棄
- Config mapマップまたはシークレットの作成/更新/削除
- エンドポイントへの変更をサブスクライブする

前提条件

Sysdig Agentをインストールし、Agent Serviceを適用します

これらの手順は、Sysdigエージェントがすでにkubernetesクラスターにデプロイされていることを前提としています。詳細については、エージェントのインストールを参照してください。エージェントをインストールしたら、Sysdigエージェントのサービスアカウント、シークレット、configmap、daemonsetの情報を手元に用意してください。

エージェントのインストール中にsysdig-agent-service.yamlが明示的にデプロイされなかった場合、ここで適用する必要があります。

```
kubectl apply -f
https://raw.githubusercontent.com/draios/sysdig-cloud-scripts/master/agent_deploy/kubernetes/
sysdig-agent-service.yaml -n sysdig-agent
```

有効化ステップを選択する

Sysdigは、さまざまなプラットフォームとディストリビューションでKubernetes auditログ統合をテストしました。以下のセクションで説明するように、それぞれに異なる手順が必要です。

Kubernetes監査イベントのルーティングは、Kubernetesバージョン間で急速に変更されました。詳細については、Kubernetesのドキュメントをご覧ください。

ルーティングは次のいずれかを介して実行されます。

- Webhookバックエンド：Kubernetesバージョン> = 1.11、または
- Audit Sinkを使用した動的バックエンド：Kubernetesバージョン> = 1.13

ディストリビューションが1.11 / 1.12と1.13+の両方をサポートする場合、動的バックエンドバージョンが優先されます。

次の表は、テスト済みのオプションをまとめたものです。

Distro/Platform	Version	Uses Webhook	Uses Dynamic	Uses Other
OpenShift	3.11	X		
OpenShift	4.2, 4.3		X	
MiniShift	3.11	X		
Kops	1.15	X		
GKE (Google)	1.13			X (bridge)
EKS (Amazon)	eks.5/ Kubernetes 1.14			X CloudWatch
RKE (Rancher)	RKE v1.0.0/Kubernetes 1.13+	X		
IKS (IBM)	1.15	X		
Minikube	1.11/1.12	X		
Minikube	1.13+		X	

Kubernetes監査ログを有効にする

これらの手順は、Kubernetesクラスターに監査構成またはログ記録がないことを前提としています。この手順では、監査ログメッセージをSysdigエージェントにルーティングするためにのみ構成を追加します。

これらの手順の多くを自動化するベータスクリプトがあり、検証/非実稼働環境に適しています。いずれの場合でも、続行する前にステップバイステップの手順を注意深く読むことをお勧めします。

OpenShift 3.11

> **注意**
>
> Openshift 3.11は、webhookバックエンドのみをサポートします（Openshiftドキュメントで「高度な監査」として説明されています）。

Kubernetes APIマスターノードで次の手順を実行します。

1. 提供されたaudit-policy.yamlファイルを/etc/origin/masterディレクトリのKubernetes APIマスターノードにコピーします。

   （このディレクトリは/etc/origin/masterのKube APIサーバーコンテナにマウントされているため、ファイルはコンテナで実行されているOpenShiftサービスによって取得されます。）

2. Webhook構成ファイルを作成し、/etc/origin/masterディレクトリのKubernetes APIマスターノードにコピーします。
3. /etc/origin/master/master-config.yamlファイルに次を追加して、既存のauditConfig：エントリを置き換えて、マスター構成を変更します。

```
auditConfig:
  enabled: true
  maximumFileSizeMegabytes: 10
  maximumRetainedFiles: 1
  auditFilePath: "/etc/origin/master/k8s_audit_events.log"
  logFormat: json
  webHookMode: "batch"
  webHookKubeConfig: /etc/origin/master/webhook-config.yaml
```

```
  policyFile: /etc/origin/master/audit-policy.yaml
```

これを行う1つの方法は、oc ex configパッチを使用することです。

上記の内容がファイルaudit-patch.yamlにあり、/etc/origin/master/master-config.yamlを/tmp/master-config.yaml.originalにコピーしたと仮定すると、次を実行できます。

```
oc ex config patch /tmp/master-config.yaml.original -p "$(cat audit-patch.yaml)"
&gt; /etc/origin/master/master-config.yaml
```

4. 以下を実行して、APIサーバーを再起動します。

```
# sudo /usr/local/bin/master-restart api
# sudo /usr/local/bin/master-restart controllers
```

再起動すると、サーバーはKubernetes auditイベントをSysdigエージェントサービスにルーティングします。

MiniShift 3.11

OpenShift 3.11と同様に、Minishift 3.11はwebhookバックエンドをサポートしていますが、MinishiftがKubernetes APIサーバーを起動する方法は異なります。したがって、コマンドライン引数は上記の手順とは多少異なります。

1. 提供されたaudit-policy.yamlファイルをMinishift VMのディレクトリ/var/lib/minishift/base/kube-apiserver/にコピーします。

   （このディレクトリは/etc/origin/masterのkube APIサーバーコンテナーにマウントされるため、コンテナーで実行されているMinishiftサービスによってファイルが取得されます。）

2. Webhook構成ファイルを作成し、Minishift VMのディレクトリ/var/ lib / minishift / base / kube-apiserver /にコピーします。
3. 以下をMinishift VMの/var/lib/minishift/base/kube-apiserver/master-config.yamlに追加して、マスター構成を変更し、必要に応じてマージ/更新します。

注：master-config.yamlは、/var/lib/minishift/base/openshift-apiserverや/var/lib/minishift/base/openshift-controller-manager/などの他のディレクトリにも存在します。

kube-apiserverにあるものを変更する必要があります。

```
kubernetesMasterConfig:
  apiServerArguments:
  audit-log-maxbackup:
  - "1"
  audit-log-maxsize:
  - "10"
  audit-log-path:
  - /etc/origin/master/k8s_audit_events.log
  audit-policy-file:
  - /etc/origin/master/audit-policy.yaml
  audit-webhook-batch-max-wait:
  - 5s
  audit-webhook-config-file:
  - /etc/origin/master/webhook-config.yaml
  audit-webhook-mode:
  - batch
```

4. 次を実行してAPIサーバーを再起動します:

```
(minishiftの場合)
# minishift openshift restart
```

再起動すると、サーバーはKubernetes auditイベントをSysdigエージェントサービスにルーティングします。

OpenShift 4.2, 4.3

デフォルトでは、Openshift 4.2/4.3は、Kubernetes APIサーバーログを有効にし、パス/var/log/kube-apiserver/audit.logで各マスターノードで利用可能にします。ただし、APIサーバーはデフォルトでは動的バックエンドを作成する機能で構成されていません。

最初に、APIサーバー構成を変更して、動的バックエンドの作成を有効にする必要があります。 次に、Audit sinks を作成して、監査イベントをSysdigエージェントにルーティングします。

1. 以下を実行して、APIサーバー構成を更新します:

```
oc patch kubeapiserver cluster --type=merge -p
'{"spec":{"unsupportedConfigOverrides":{"apiServerArguments":{"audit-dynamic-configura
tion":["true"],"feature-gates":["DynamicAuditing=true"],"runtime-config":["auditregist
ration.k8s.io/v1alpha1=true"]}}}}'
```

2. APIサーバーが更新された構成で再起動するのを待ちます。
3. ダイナミック audit sinkを作成します。
4. ダイナミック audit sinkが作成されると、Kubernetes Audit イベントがSysdigエージェントサービスにルーティングされます

Kops

最新リリースのKops（1.15）はまだ動的バックエンドの構成をサポートしていないため、これらの手順ではWebhookを使用します。kops setを使用してクラスター構成を変更し、kops updateを使用して構成を更新してから、kops rolling-updateを使用してローリングアップデートを実行します。

注：Kopsの修正はhttps://github.com/kubernetes/kops/pull/7424にマージされていますが、まだKopsリリースの一部ではありません。

1. Webhook構成ファイルを作成し、ローカルに保存します。
2. 現在のクラスター構成を取得し、ファイルに保存します。

   kops get cluster <your cluster name> -o yaml> cluster-current.yaml

3. webhook-config.yamlが/var/lib/k8s_auditの各マスターノードで利用可能であり、webhookバックエンドを有効にするために必要な引数でkube-apiserverプロセスが実行されるようにするには、cluster.yamlを編集して追加/fileAssetsおよびkubeAPIServerセクションを次のように変更します。

```
apiVersion: kops.k8s.io/v1alpha2
kind: Cluster
spec:
  ...
  fileAssets:
```

```
    - name: webhook-config
      path: /var/lib/k8s_audit/webhook-config.yaml
      roles: [Master]
      content: |
        <contents of webhook-config.yaml go here>
    - name: audit-policy
      path: /var/lib/k8s_audit/audit-policy.yaml
      roles: [Master]
      content: |
        <contents of audit-policy.yaml go here>
  ...
  kubeAPIServer:
    auditLogPath: /var/lib/k8s_audit/audit.log
    auditLogMaxBackups: 1
    auditLogMaxSize: 10
    auditWebhookBatchMaxWait: 5s
    auditPolicyFile: /var/lib/k8s_audit/audit-policy.yaml
    auditWebhookConfigFile: /var/lib/k8s_audit/webhook-config.yaml
…
```

yqを使用してこれを行う簡単な方法は、次のスクリプトを使用することです:

```
cat <<EOF > merge.yaml
spec:
  fileAssets:
    - name: webhook-config
      path: /var/lib/k8s_audit/webhook-config.yaml
      roles: [Master]
      content: |
$(cat webhook-config.yaml | sed -e 's/^/        /')
    - name: audit-policy
      path: /var/lib/k8s_audit/audit-policy.yaml
      roles: [Master]
      content: |
$(cat audit-policy.yaml | sed -e 's/^/        /')
  kubeAPIServer:
    auditLogPath: /var/lib/k8s_audit/audit.log
    auditLogMaxBackups: 1
    auditLogMaxSize: 10
```

```
    auditWebhookBatchMaxWait: 5s
    auditPolicyFile: /var/lib/k8s_audit/audit-policy.yaml
    auditWebhookConfigFile: /var/lib/k8s_audit/webhook-config.yaml
EOF

yq m -a cluster-current.yaml merge.yaml > cluster.yaml
```

4. 新しいクラスター構成でKopsを構成します。

```
kops replace -f cluster.yaml
```

5. クラスター構成を更新して、クラスターへの変更を準備します。

```
kops update cluster <your cluster name> --yes
```

6. ローリング更新を実行して、新しいファイルとAPIサーバー構成でマスターノードを再デプロイします。

```
kops rolling-update cluster --yes
```

GKE (Google)

注意

これらの手順は、クラスターを作成し、gcloudおよびkubectlコマンドラインプログラムがクラスターと対話するように構成済みであることを前提としています。

現在、kubectl exec trace 機能はGKEクラスターではサポートされていませんが、他のすべてのアクティビティ監査機能は正常に機能するでしょう。

GKEはすでにKubernetes監査ログを提供していますが、ログはStackdriverを使用して公開され、Kubernetesが使用するネイティブ形式とは異なる形式です。

物事を単純化するために、Stackdriverから監査ログを読み取り、Kubernetesネイティブ形式に一致するように再フォーマットし、構成可能なWebhookおよびSysdigエージェントサービスにログを送信するブリッジプログラム (https://github.com/sysdiglabs/stackdriver-webhook-bridge/blob/master/stackdriver-webhook-bridge.yaml) を作成しました。

1. ログを読み取ることができるGoogle Cloud（Kubernetesではない）サービスアカウントとキーを作成します。

```
$ gcloud iam service-accounts create swb-logs-reader --description "Service
account used by stackdriver-webhook-bridge" --display-name
"stackdriver-webhook-bridge logs reader"

$ gcloud projects add-iam-policy-binding <your gce project id> --member
serviceAccount:swb-logs-reader@<your gce project id>.iam.gserviceaccount.com
--role 'roles/logging.viewer'

$ gcloud iam service-accounts keys create $PWD/swb-logs-reader-key.json
--iam-account swb-logs-reader@<your gce project id>.iam.gserviceaccount.com
```

1. サービスアカウントキーを含むKubernetesシークレットを作成します。

```
kubectl create secret generic stackdriver-webhook-bridge
--from-file=key.json=$PWD/swb-logs-reader-key.json -n sysdig-agent
```

2. 提供されているものを使用して、ブリッジプログラムをクラスタにデプロイします。

stackdriver-webhook-bridge.yaml file:

```
kubectl apply -f stackdriver-webhook-bridge.yaml -n sysdig-agent
```

ブリッジプログラムは、監査イベントをドメイン名sysdig-agent.sysdig-agent.svc.cluster.localにルーティングします。これは、エージェントのデプロイ時または前提条件ステップとして作成したsysdig-agentサービスに対応します。

EKS (Amazon)

これらの手順は、Kubernetes v1.14のeks.5で検証されました。

Amazon EKSは監査ログ用のウェブフックを提供しませんが、監査ログをCloudWatchに転送できます。 SysdigエージェントからCloudWatchログにアクセスするには、次の手順に従います。

1. EKSクラスターのCloudWatchログを有効にします。

2. ワーカーノードからCloudWatchへのアクセスを許可します。
3. CloudWatchをポーリングし、イベントをSysdigエージェントに転送する新しいデプロイメントを追加します。

AWS UIを使用して実装できる設定例(https://github.com/sysdiglabs/ekscloudwatch)と、監査ログフォワーダの例のコードおよびイメージを見つけることができます。（本番システムでは、これはIaCスクリプトとして実装されます。）

CloudWatchは追加のAWS有料サービスであることに注意してください。 さらに、このソリューションでは、ワーカーノードで実行されているすべてのポッドがAWS APIを介してCloudWatchログを読み取ることが許可されます

RKE (Rancher) with Kubernetes 1.13+

これらの手順は、RKE v1.0.0およびKubernetes v1.16.3で検証されました。 Kubernetes v1.13以降のバージョンで動作するでしょう。

監査サポートはデフォルトですでに有効になっていますが、さらに細かくするために監査ポリシーを更新する必要があります。これらの手順により、エージェントのサービスを指すwebhookバックエンドが有効になります。監査機能フラグを有効にする方法がないため、動的監査バックエンドはサポートされていません。

1. 各Kubernetes APIマスターノードで、ディレクトリ/var/lib/k8s_auditを作成します。
2. 各Kubernetes APIマスターノードで、提供されたaudit-policy.yamlファイルをマスターノードの/var/lib/k8s_auditディレクトリにコピーします。（このディレクトリはAPIサーバーにマウントされ、監査/ウェブフックファイルへのアクセスを許可します。）
3. Webhook構成ファイルを作成し、各Kubernetes APIマスターノードのディレクトリ/var/lib/k8s_auditにコピーします。
4. RKEクラスター設定cluster.ymlを変更して、extra_argsセクションとextra_bindsセクションをkube-apiセクションに追加します。 次に例を示します。

```
kube-api:
...
    extra_args:
      audit-policy-file: /var/lib/k8s_audit/audit-policy.yaml
      audit-webhook-config-file: /var/lib/k8s_audit/webhook-config.yaml
```

```
      audit-webhook-batch-max-wait: 5s
    extra_binds:
    - /var/lib/k8s_audit:/var/lib/k8s_audit
...
```

これにより、代替監査ポリシーを使用し、作成したwebhookバックエンドを使用するように、APIサーバーのコマンドライン引数が変更されます。

5. rke upを使用してRKEクラスターを再起動します。

IKS (IBM)

IKSは、Kubernetes監査イベントを単一の構成可能なWebhookバックエンドURLにルーティングすることをサポートしています。動的なAudit sinksはサポートされておらず、送信されるKubernetes監査イベントを制御する監査ポリシーを変更する機能はサポートされていません。

以下の手順は、Fluentdとの統合方法に関するIBM提供のドキュメントを改編したものです。 そこに記載されているクラスターおよびアプリのログを転送するためのIKSツールに精通している（または確認する）ことが期待されます。

制限：Kubernetesのデフォルトの監査ポリシーには、通常、RequestまたはRequestResponseレベルのイベントは含まれません。つまり、作成/変更されるオブジェクトを詳細に調べるルール（たとえば、ka.req.*およびka.resp.*を使用するルール） フィールド）はトリガーされません。 これには次のルールが含まれます。

- `Create Disallowed Pod`
- `Create Privileged Pod`
- `Create Sensitive Mount Pod`
- `Create HostNetwork Pod`
- `Pod Created in Kube Namespace`
- `Create NodePort Service`
- `Create/Modify Configmap With Private Credentials`
- `Attach to cluster-admin Role`
- `ClusterRole With Wildcard Created`
- `ClusterRole With Write Privileges Created`

- `ClusterRole With Pod Exec Created`

これらの手順では、FluentdからSysdigエージェントサービスにリダイレクトする方法について説明します。

1. webhookバックエンドURLをsysdig-agentサービスのIPアドレスに設定します。

   ```
   http://$(kubectl get service sysdig-agent -o=jsonpath={.spec.clusterIP} -n
   sysdig-agent):7765/k8s_audit
   ```

2. webhookバックエンドURLが設定されていることを確認します。

   ```
   ibmcloud ks cluster master audit-webhook get --cluster <cluster_name_or_ID>
   ```

3. クラスターマスターを更新して、WebhookをKubernetes APIサーバーに適用します。 マスターが更新されるまでに数分かかる場合があります。

   ```
   ibmcloud ks cluster master refresh --cluster <cluster_name_or_ID>
   ```

ネイティブKubernetes もしくは、 Minikube 1.11/1.12

これらの手順は、Minikube 1.2.0を使用して検証されました。 他のMinikubeバージョンも、Kubernetesバージョン1.11/1.12（webhookバックエンドのみをサポート）を実行している限り機能します。 以下のすべての場合において、「Minikube VM」は、Minikubeによって作成されたVMを指します。 --vm-driver = noneを使用している場合、これはローカルマシンを意味します。

4. マスターノードにディレクトリ/var/lib/k8s_auditを作成します。 （Minikubeでは、Minikube VM上にある必要があります。）
5. 提供されたaudit-policy.yamlファイルをディレクトリ/var/lib/k8s_auditにコピーします。（このディレクトリはAPIサーバーにマウントされ、監査/ウェブフックファイルへのアクセスを提供します。Minikubeでは、Minikube VM上にある必要があります。）
6. Webhook構成ファイルを作成し、各Kubernetes APIマスターノードのディレクトリ/var/lib/k8s_auditにコピーします。
7. /etc/kubernetes/manifests/kube-apiserver.yamlでKubernetes APIサーバーマニフェストを変更し、次のコマンドライン引数を追加します。

```
--audit-log-path=/var/lib/k8s_audit/k8s_audit_events.log
--audit-policy-file=/var/lib/k8s_audit/audit-policy.yaml
--audit-log-maxbackup=1
--audit-log-maxsize=10
--audit-webhook-config-file=/var/lib/k8s_audit/webhook-config.yaml
--audit-webhook-batch-max-wait=5s
```

コマンドライン引数は、プログラム/usr/local/bin/kube-apiserverへの引数としてコンテナ仕様で提供されます。 マニフェストの関連セクションは次のようになります。

```
spec:
  containers:
  - command:
    - kube-apiserver --allow-privileged=true --anonymous-auth=false
      --audit-log-path=/var/lib/k8s_audit/audit.log
      --audit-policy-file=/var/lib/k8s_audit/audit-policy.yaml
      --audit-log-maxbackup=1
      --audit-log-maxsize=10
      --audit-webhook-config-file=/var/lib/k8s_audit/webhook-config.yaml
      --audit-webhook-batch-max-wait=5s
      …
```

8. /etc/kubernetes/manifests/kube-apiserver.yamlにあるKubernetes APIサーバーマニフェストを変更して、/var/lib/k8s_auditのマウントをkube-apiserverコンテナに追加します。 関連するセクションは次のようになります。

```
volumeMounts:
- mountPath: /var/lib/k8s_audit/
  name: k8s-audit
  readOnly: true
     ...
volumes:
- hostPath:
  path: /var/lib/k8s_audit
  type: DirectoryOrCreate
  name: k8s-audit
…
```

9. マニフェストを変更すると、Kubernetes APIサーバーが自動的に再起動します。 再起動すると、Kubernetes監査イベントをSysdigエージェントのサービスにルーティングします。

ネイティブKubernetes もしくは Minikube 1.13+

これらの手順は、Minikube 1.2.0を使用して検証されました。その他のMinikubeバージョンも、Kubernetesバージョン1.13以降を実行している限り機能します。1.13を使用する場合、これらの命令は動的sinksを有効にしますが、それでもkube-apiserverコマンドライン引数の変更が必要です。

以下のすべての場合において、「Minikube VM」は、Minikubeによって作成されたVMを指します。--vm-driver = noneを使用している場合、これはローカルマシンを意味します。

1. マスターノードにディレクトリ/var/lib/k8s_auditを作成します。 （Minikubeでは、Minikube VM上にある必要があります。）
2. 提供されたaudit-policy.yamlファイルをディレクトリ/var/lib/k8s_auditにコピーします。 （このディレクトリはAPIサーバーにマウントされ、監査/ウェブフックファイルへのアクセスを提供します。Minikubeでは、Minikube VM上にある必要があります。）
3. /etc/kubernetes/manifests/kube-apiserver.yamlでKubernetes APIサーバーマニフェストを変更し、次のコマンドライン引数を追加します。

```
--audit-log-path=/var/lib/k8s_audit/k8s_audit_events.log
--audit-policy-file=/var/lib/k8s_audit/audit-policy.yaml
--audit-log-maxbackup=1
--audit-log-maxsize=10
--audit-dynamic-configuration
--feature-gates=DynamicAuditing=true
--runtime-config=auditregistration.k8s.io/v1alpha1=true
```

コマンドライン引数は、プログラム/usr/local/bin/kube-apiserverへの引数としてコンテナ仕様で提供されます。 マニフェストの関連セクションは次のようになります。

```
spec:
  containers:
  - command:
    - kube-apiserver --allow-privileged=true --anonymous-auth=false
```

```
--audit-log-path=/var/lib/k8s_audit/audit.log
--audit-policy-file=/var/lib/k8s_audit/audit-policy.yaml
--audit-log-maxbackup=1
--audit-log-maxsize=10
--audit-dynamic-configuration
--feature-gates=DynamicAuditing=true
--runtime-config=auditregistration.k8s.io/v1alpha1=true
...
```

4. /etc/kubernetes/manifests/kube-apiserver.yamlにあるKubernetes APIサーバーマニフェストを変更して、/var/lib/k8s_auditのマウントをkube-apiserverコンテナーに追加します。 関連するセクションは次のようになります。

```
volumeMounts:
- mountPath: /var/lib/k8s_audit/
  name: k8s-audit
  readOnly: true
    ...
volumes:
- hostPath:
  path: /var/lib/k8s_audit
  type: DirectoryOrCreate
  name: k8s-audit
...
```

   マニフェストを変更すると、Kubernetes APIサーバーが自動的に再起動します。

5. ダイナミックAudit Sinkを作成する
   ダイナミックAudit Sinkが作成されると、Kubernetes監査イベントがSysdigエージェントのサービスにルーティングされます。

Webhookまたは動的バックエンドを準備する

プラットフォーム固有の指示のほとんどは、これらの方法のいずれかを使用します。

Webhook構成ファイルを作成する

Sysdigは、ポート7765を介して、Sysdigエージェントサービスに関連付けられたIPに監査イベントを送信するテンプレート化されたリソースファイルを提供します。

実際のIPが環境変数AGENT_SERVICE_CLUSTERIPで定義されているという点で「テンプレート化」されており、envsubstなどのプログラムを使用してプラグインできます。

1. webhook-config.yaml.inをダウンロードします。

2. 以下を実行して、エージェントのインストール時または前提条件のステップで作成したsysdig-agentサービスに関連付けられたClusterIP IPアドレスをテンプレートファイルに入力します。

   AGENT_SERVICE_CLUSTERIP = $（kubectl get service sysdig-agent -o = jsonpath = {。spec.clusterIP} -n sysdig-agent）envsubst <webhook-config.yaml.in> webhook-config.yaml

   注： sysdig-agent.sysdig-agent.svc.cluster.localなどのサービスドメイン名はKubernetes APIサーバーからは解決できませんが（通常はポッドとして実行されますが、実際にはクラスターの一部ではありません）、ClusterIPそれらのサービスに関連付けられたルーティング可能です。

ヒント

webhookバックエンドを使用して監査イベントをルーティングすることは、Kubernetes v1.11 +から利用できる機能です。 背景情報については、Kubernetesのドキュメントを参照してください。

ダイナミック Audit Sinkを作成する

ダイナミック Audit Sinkを使用する場合、監査イベントをsysdigエージェントサービスに送信するAuditSinkオブジェクトを作成する必要があります。

Sysdigは、シンクの作成に使用できるテンプレートファイルを提供します。

1. audit-sink.yaml.inをダウンロードします。
2. 以下を実行して、エージェントのインストール時または前提条件のステップで作成したsysdig-agentサービスに関連付けられたClusterIP IPアドレスをテンプレートファイルに入力します。

```
AGENT_SERVICE_CLUSTERIP=$(kubectl get service sysdig-agent
-o=jsonpath={.spec.clusterIP} -n sysdig-agent) envsubst < audit-sink.yaml.in >
audit-sink.yaml
```

ヒント

AuditSink APIオブジェクトを使用したダイナミックAudit webhookは、Kubernetes v1.13 +から利用できます。 背景情報については、Kubernetesのドキュメントを参照してください。

統合をテストする

Kubernetes監査イベントがエージェントに適切に渡されることをテストするには、次のいずれかを実行できます。

- All K8s Object Modificationsポリシーを有効にし、デプロイメント、サービス、configmap、またはネームスペースを作成して、イベントが記録および転送されているかどうかを確認します。
- 疑わしいK8sアクティビティなどの他のポリシーを有効にし、テストします。
- falco-event-generator Dockerイメージを使用して、Sysdig Secureで提供されるデフォルトのルール/ポリシーの多くにマッピングするアクティビティを生成できます。 次のようなコマンドラインを使用してイメージを実行できます。

```
docker run -v $HOME/.kube:/root/.kube -it falcosecurity/falco-event-generator
k8s_audit
```

これにより、ネームスペース falco-event-generatorにリソースが作成されます。

参照：Sysdig Secure内でのFalcoの使用およびこのツールの詳細については、Falcoのネイティブドキュメント

（ベータ）構成変更を自動化するスクリプト

便宜上、Sysdigはスクリプトを作成しました：enable-k8s-audit.shは、EKSを除く上記のすべてのKubernetesディストリビューションの監査ログサポートを有効にするために必要な手順を実行します。

bash enable-k8s-audit.sh <distribution>で実行できます。<distribution>は次のいずれかです。

- minishift-3.11
- openshift-3.11
- openshift-4.2, openshift-4.3
- gke
- iks
- rke-1.13 (implies Kubernetes 1.13)
- kops
- minikube-1.13 (implies Kubernetes 1.13)
- minikube-1.12 (implies Kubernetes 1.11/1.12)

sysdig-cloud-scripts / k8s_audit_configディレクトリから実行する必要があります。
場合によっては、GCEプロジェクトID、IKSクラスター名などの入力を求められることがあります。
Minikube/Openshift-3.11/Minishift 3.11の場合、ssh/scpを使用してAPIマスターノードにファイルをコピーし、スクリプトを実行します。 それ以外の場合は、完全に自動化する必要があります。

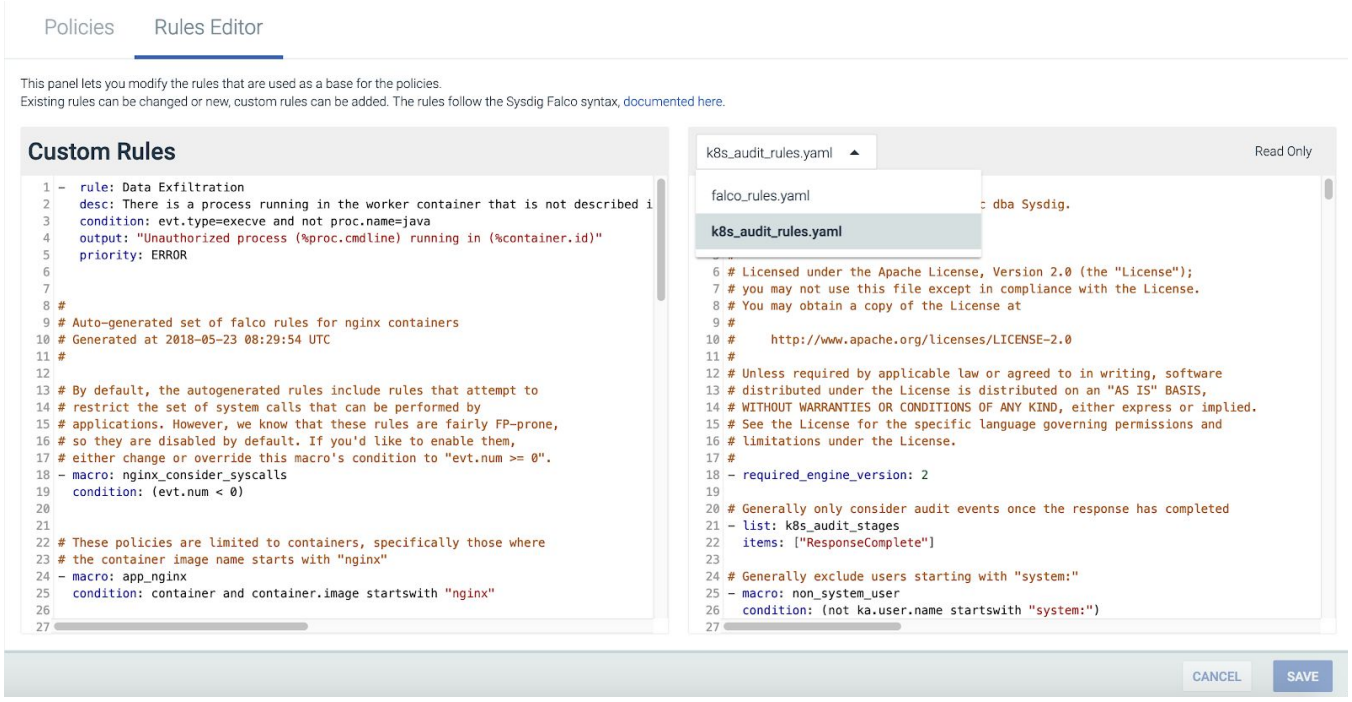
UIで表示

Kubernetes監査ログの新しいFalcoルールを使用するには、ポリシーを作成する必要があります。 ポリシーの作成については、ポリシーのドキュメントを参照してください。

監査ログルールの表示

Kubernetes監査ログルールは、ポリシーモジュールにあるSysdigポリシールールエディターで表示できます。 監査ルールを表示するには：

1. [`Policies`]モジュールから、[`Rules Editor`]タブに移動します。
2. デフォルトのルールのドロップダウンメニューを開き、k8s_audit_rules.yamlを選択します。



監査イベントを表示する

Kubernetes監査イベントは、クラスター内のSysdigエージェントデーモンセットにルーティングされるようになりました。

ポリシーが作成されると、Sysdig Secure Policy Eventsモジュールを介して監査イベントを監視できるようになります。