Red Hat Advanced Cluster Management for Kubernetes 2.4

可観測性

可観測性サービスの有効化およびカスタマイズを実行してマネージドクラスターを最適化する方法は、こちらを参照してください。

Last Updated: 2022-10-02

Red Hat Advanced Cluster Management for Kubernetes 2.4 可観測性

可観測性サービスの有効化およびカスタマイズを実行してマネージドクラスターを最適化する方法は、こちらを参照してください。

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

可観測性サービスの有効化およびカスタマイズを実行してマネージドクラスターを最適化する方法は、こちらを参照してください。

目次

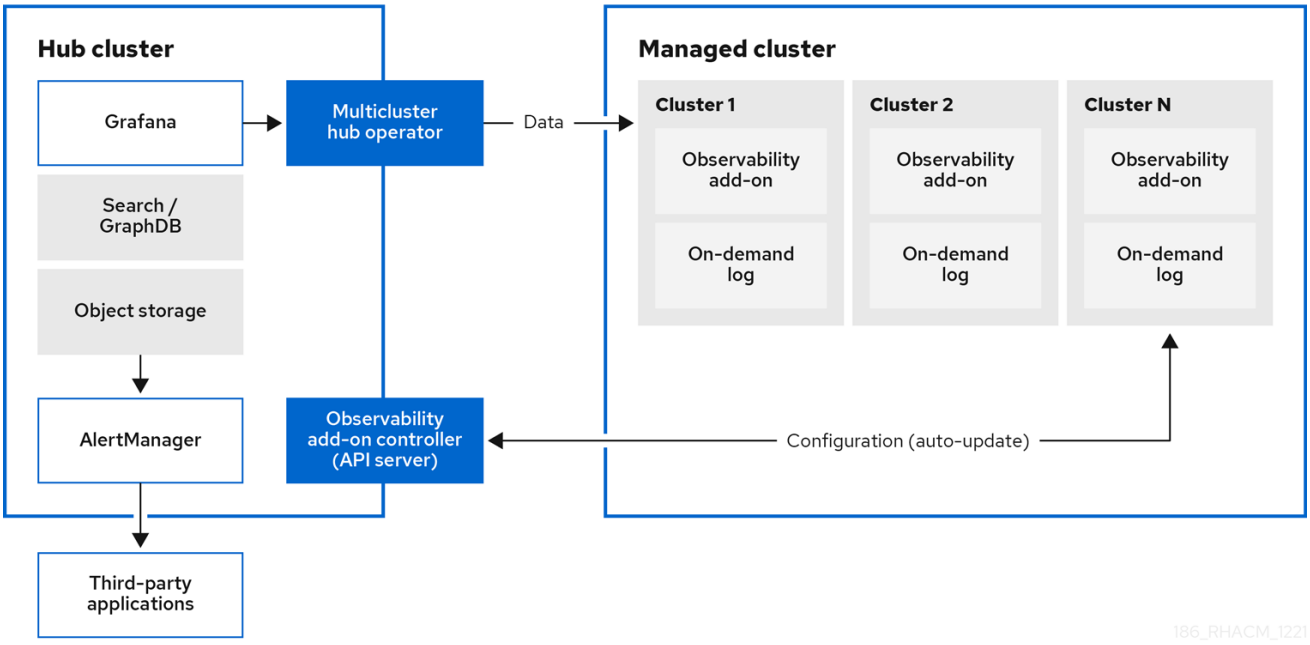
第1章 環境の監視の紹介

可観測性サービスを有効にすると、Red Hat Advanced Cluster Management for Kubernetes を使用して、マネージドクラスターに関する理解を深め、最適化することができます。この情報は、コストを節約し、不要なイベントを防ぐことができます。

- 環境の監視
- 可観測性サービスの有効化
- 可観測性のカスタマイズ
- Grafana ダッシュボードの設計
- Red Hat Insights の可観測性
- Insights PolicyReports の管理

1.1. 環境の監視

Red Hat Advanced Cluster Management for Kubernetes を使用して、マネージドクラスターに関する理解を深め、最適化することができます。ハブクラスターで可観測性サービス Operator (**multicluster-observability-operator**) を有効にして、マネージドクラスターの状態を監視します。以下のセクションでは、マルチクラスター可観測性サービスのアーキテクチャーについて説明します。



注記: **オンデマンドログ** は、特定 の Pod のログをリアルタイムで取得するエンジニア用のアクセスを提供します。ハブクラスターからのログは集約されません。これらのログは、検索サービスとコンソールの他の部分を使用してアクセスできます。

- 可観測性サービス
- メトリクスのタイプ
- 可観測性 Pod の容量要求
- 可観測性サービスで使用される永続ストア

- サポート

1.1.1. 可観測性サービス

デフォルトでは可観測性は、製品のインストール時に追加されますが、有効化されません。永続ストレージの要件で、可観測性サービスはデフォルトで有効化されていません。Red Hat Advanced Cluster Management は、以下の S3 互換のある、安定したオブジェクトストアをサポートします。

- Amazon S3
  注記: Thanos のオブジェクトストアインターフェースは、AWS S3 RESTFUL API 互換の API、または Minio や Ceph などのその他の S3 と互換のあるオブジェクトストアをサポートします。
- Google Cloud Storage
- Azure ストレージ
- Red Hat OpenShift Data Foundation
  重要: オブジェクトストアを設定する場合は、機密データを永続化する時に必要な暗号化要件を満たすようにしてください。サポートされるオブジェクトストアの全一覧は、Thanos のドキュメント を参照してください。

サービスを有効にすると、**observability-endpoint-operator** はインポートまたは作成された各クラスターに自動的にデプロイされます。このコントローラーは、Red Hat OpenShift Container Platform Prometheus からデータを収集してから、Red Hat Advanced Cluster Management ハブクラスターに送信します。

ハブクラスターが **local-cluster** として自己インポートする場合は、可観測性もそのクラスターで有効になり、メトリクスがハブクラスターから収集されます。

可観測性サービスは、Prometheus AlertManager のインスタンスをデプロイすることで、サードパーティーのアプリケーションでのアラートの転送が可能になります。また、ダッシュボード (静的) またはデータ検索を使用してデータの可視化を有効にする Grafana のインスタンスも含まれます。Red Hat Advanced Cluster Managementは、Grafanaのバージョン8.1.3をサポートします。Grafana ダッシュボードを設計することもできます。詳細は、「 Grafana ダッシュボードの設計」を 参照してください。

カスタムの レコーディングルール または アラートルール を作成して、可観測性サービスをカスタマイズできます。

可観測性有効化の詳細は、「可観測性サービスの有効化」を参照してください。

1.1.2. メトリクスのタイプ

デフォルトで、OpenShift Container Platform は Telemetry サービスを使用してメトリクスを Red Hat に送信します。**acm_managed_cluster_info** は、Red Hat Advanced Cluster Management で利用でき、Telemetry に含まれていますが、Red Hat Advanced Cluster Management **Observe 環境の概要** ダッシュボードには表示され **ません**。

OpenShift Container Platform ドキュメントで、Telemetry を使用して収集されて送信されるメトリクスのタイプについて確認します。詳細は、Telemetry で収集される 情報 を参照してください。

1.1.3. 可観測性 Pod の容量要求

可観測性サービスをインストールするには、可観測性コンポーネントで 2701mCPU および 11972Mi のメモリーが必要です。以下の表は、**observability-addons** が有効なマネージドクラスター 5 台の Pod 容量要求の一覧です。

表1.1 可観測性 Pod の容量要求

デプロイメントまたは StatefulSet	コンテナー名	CPU (mCPU)	メモリー (Mi)	レプリカ	Pod の合計 CPU	Pod の合計メモリー
observability-alertmanager	Alertmanager	4	200	3	12	600
	config-reloader	4	25	3	12	75
	alertmanager-proxy	1	20	3	3	60
observability-grafana	grafana	4	100	2	8	200
	grafana-dashboard-loader	4	50	2	8	100
observability-observatorium-api	observatorium-api	20	128	2	40	256
observability-observatorium-operator	observatorium-operator	100	100	1	10	50
observability-rbac-query-proxy	rbac-query-proxy	20	100	2	40	200
	oauth-proxy	1	20	2	2	40
observability-thanos-compact	thanos-compact	100	512	1	100	512
observability-thanos-query	thanos-query	300	1024	2	600	2048
observability-thanos-query-frontend	thanos-query-frontend	100	256	2	200	512

デプロイメントまたは StatefulSet	コンテナー名	CPU (mCPU)	メモリー (Mi)	レプリカ	Pod の合計 CPU	Pod の合計メモリー
observability-thanos-query-frontend-memcached	Memcached CRD	45	128	3	135	384
	exporter	5	50	3	15	150
observability-thanos-receive-controller	thanos-receive-controller	4	32	1	4	32
observability-thanos-receive-default	thanos-receive	300	512	3	900	1536
observability-thanos-rule	thanos-rule	50	512	3	150	1536
	configmap-reloader	4	25	3	12	75
observability-thanos-store-memcached	Memcached CRD	45	128	3	135	384
	exporter	5	50	3	15	150
observability-thanos-store-shard	thanos-store	100	1024	3	300	3072

1.1.4. 可観測性サービスで使用される永続ストア

Red Hat Advanced Cluster Management をインストールするときは、次の永続ボリューム (PV) を作成して、Persistent Volume Claims (PVC) を自動的にアタッチできるようにする必要があります。デフォルトのストレージクラスが指定されていない場合、またはデフォルト以外のストレージクラスを使用して PV をホストする場合は、**MultiClusterObservability** でストレージクラスを定義する必要があります。Prometheus が使用するものと同様に、ブロックストレージを使用することをお勧めします。また、**alertmanager**、**thanos-compactor**、**thanos-ruler**、**thanos-receive-default**、および **thanos-store-shard** の各レプリカには、独自の PV が必要です。次の表を参照します。

表1.2 永続ボリュームの表一覧

永続ボリューム名	目的

Alertmanager	Alertmanager は **nflog** データおよび通知なしのアラートをストレージに保存します。**nflog** は、通知されたレシーバーおよび、アクティブな通知と解決済みの通知、通知により特定されたコンテンツのハッシュダイジェストについての追記専用のログです。
thanos-compact	コンパクターは、処理の中間データとバケット状態キャッシュの保存にローカルのディスク領域が必要です。必要な領域は、下層にあるブロックサイズにより異なります。コンパクターには、すべてのソースブロックをダウンロードして、ディスクで圧縮ブロックを構築するために十分な領域が必要です。ディスク上のデータは、次回の再起動までに安全に削除でき、最初の試行でクラッシュループコンパクターの停止が解決されるはずです。ただし、次の再起動までにバケットの状態キャッシュを効果的に使用するためには、コンパクターの永続ディスクを用意することが推奨されます。
thanos-rule	thanos ruler は、固定の間隔でクエリーを発行して、選択したクエリー API に対して Prometheus 記録およびアラートルールを評価します。ルールの結果は、Prometheus 2.0 ストレージ形式でディスクに書き込まれます。このステートフルセットで保持されるデータの期間 (時間または日) は、API バージョンの **observability.open-cluster-management.io/v1beta1** で修正されました。**observability.open-cluster-management.io/v1beta2**: **RetentionInLocal** の API パラメーターとして公開されました。
thanos-receive-default	Thanos receiver は、受信データ (Prometheus リモート書き込みリクエスト) を受け入れて Prometheus TSDB のローカルインスタンスに書き込みます。TSDB ブロックは定期的に (2 時間)、長期的に保存および圧縮するためにオブジェクトストレージにアップロードされます。ローカルキャッシュを実行するこのステートフルセットで保持される期間 (時間または日) は、API バージョン **observability.open-cluster-management.io/v1beta** で修正されました。**observability.open-cluster-management.io/v1beta2**: **RetentionInLocal** の API パラメーターとして公開されました。
thanos-store-shard	これは、主に API ゲートウェイとして機能するため、大量のローカルディスク容量は必要ありません。これは、起動時に Thanos クラスターに参加して、アクセスできるデータを広告します。ローカルディスク上のすべてのリモートブロックに関する情報のサイズを小さく保ち、バケットと同期させます。このデータは通常、起動時間が長くなると、再起動時に安全に削除できます。

注記: 時系列の履歴データはオブジェクトストアに保存されます。Thanos は、オブジェクトストレージをメトリクスおよび関連するメタデータのプライマリーストレージとして使用します。オブジェクトストレージおよび downsampling 機能の詳細は、「可観測性サービスの有効化」を参照してください。

1.1.5. サポート

Red Hat Advanced Cluster Management は、Red Hat OpenShift Data Foundation（以前の Red Hat OpenShift Container Storage）によってテストされ、完全にサポートされています。

Red Hat Advanced Cluster Management は、S3 API と互換性のあるユーザー提供のオブジェクトストレージにおけるマルチクラスター可観測性 Operator の機能をサポートします。

Red Hat Advanced Cluster Management はビジネス的に妥当な範囲内で、根本原因の特定を支援します。

サポートチケットが発行され、根本的な原因がお客様が提供したS3互換オブジェクトストレージの結果であると判断された場合は、カスタマーサポートチャネルを使用して問題を解決する必要があります。

Red Hat Advanced Cluster Management は、お客様が起票したサポートチケットの根本的な原因が S3 互換性のあるオブジェクトストレージプロバイダーである場合に、問題修正サポートの確約はありません。

可観測性サービスの設定、メトリクスおよびその他のデータの表示方法は、可観測性のカスタマイズ を参照してください。

1.2. 可観測性サービスの有効化

可観測性サービス (**multicluster-observability-operator**) でマネージドクラスターの状態を監視します。

必要なアクセス権限: クラスターの管理者または **open-cluster-management:cluster-manager-admin** ロール。

- 前提条件
- 可観測性の有効化
- MultiClusterObservability CR の作成
- Red Hat OpenShift Container Platform コンソールからの可観測性の有効化
- 外部メトリクスクエリーの使用
- 可観測性の無効化

1.2.1. 前提条件

- Red Hat Advanced Cluster Management for Kubernetes がインストールされている。詳細は、ネットワーク接続時のオンラインインストール を参照してください。
- デフォルトのストレージクラスが指定されていない場合、**MultiClusterObservability** CR でストレージクラスを定義する必要があります。
- ストレージソリューションを作成するようにオブジェクトストアが設定されている。Red Hat Advanced Cluster Management は、安定したオブジェクトストアで以下のクラウドプロバイダーをサポートします。
  - Amazon Web Services S3 (AWS S3)
  - Red Hat Ceph（S3 互換 API）
  - Google Cloud Storage
  - Azure ストレージ
  - Red Hat OpenShift Data Foundation（旧称： Red Hat OpenShift Container Storage）
  - Red Hat OpenShift on IBM(ROKS)
    重要: オブジェクトストアを設定する場合は、機密データを永続化する時に必要な暗号化要件を満たすようにしてください。Thanos がサポートするオブジェクトストアの詳細は、Thanos のドキュメント を参照してください。

1.2.2. 可観測性の有効化

MultiClusterObservability カスタムリソース (CR) を作成して可観測性サービスを有効にします。可観測性を有効にする前に、「可観測性 Pod の容量要求」を参照してください。

注記: Red Hat Advanced Cluster Management が管理する OpenShift Container Platform マネージドクラスターで可観測性を有効または無効にすると、可観測性エンドポイント Operator は、ローカル Prometheus を自動的に再起動する alertmanager 設定を追加して **cluster-monitoring-config ConfigMap** を更新します。

可観測性サービスを有効にするには、以下の手順を実行します。

1. Red Hat Advanced Cluster Management ハブクラスターにログインします。
2. 以下のコンマを使用して可観測性サービスの namespace を作成します。

```
oc create namespace open-cluster-management-observability
```

3. プルシークレットを生成します。Red Hat Advanced Cluster Management が **open-cluster-management** namespace に インストールされている場合は、以下のコマンドを実行します。

```
DOCKER_CONFIG_JSON=`oc extract secret/multiclusterhub-operator-pull-secret -n open-cluster-management --to=-`
```

multiclusterhub-operator-pull-secret が namespace に定義されていない場合には、**pull-secret** を **openshift-config** namespace から **open-cluster-management-observability** namespace にコピーします。以下のコマンドを実行します。

```
DOCKER_CONFIG_JSON=`oc extract secret/pull-secret -n openshift-config --to=-`
```

-

次に **open-cluster-management-observability** namespace でプルリクエストを作成して、以下のコマンドを実行します。

```
oc create secret generic multiclusterhub-operator-pull-secret \
    -n open-cluster-management-observability \
    --from-literal=.dockerconfigjson="$DOCKER_CONFIG_JSON" \
    --type=kubernetes.io/dockerconfigjson
```

4. お使いのクラウドプロバイダーのオブジェクトストレージのシークレットを作成します。シークレットには、ストレージソリューションへの認証情報を追加する必要があります。たとえば、以下のコマンドを実行します。

```
oc create -f thanos-object-storage.yaml -n open-cluster-management-observability
```

サポートされるオブジェクトストアのシークレットの例を以下に示します。

- Red Hat Advanced Cluster Management では、以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_S3_BUCKET
      endpoint: YOUR_S3_ENDPOINT
      insecure: true
      access_key: YOUR_ACCESS_KEY
      secret_key: YOUR_SECRET_KEY
```

- Amazon S3 または S3 と互換性のある場合には、シークレットは以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_S3_BUCKET
      endpoint: YOUR_S3_ENDPOINT
      insecure: true
      access_key: YOUR_ACCESS_KEY
      secret_key: YOUR_SECRET_KEY
```

詳細は、『Amazon Simple Storage Service ユーザーガイド』を参照してください。

- Google の場合は、以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: GCS
    config:
      bucket: YOUR_GCS_BUCKET
      service_account: YOUR_SERVICE_ACCOUNT
```

詳細は、「Google Cloud Storage とは」を参照してください。

- Azure の場合は、以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: AZURE
    config:
      storage_account: YOUR_STORAGE_ACCT
      storage_account_key: YOUR_STORAGE_KEY
      container: YOUR_CONTAINER
      endpoint: blob.core.windows.net
      max_retries: 0
```

詳細は、Azure Storage のドキュメント を参照してください。

注記: Azure を Red Hat OpenShift Container Platform クラスターのオブジェクトストレージとして使用する場合には、クラスターに関連付けられたストレージアカウントはサポートされません。新規ストレージアカウントを作成する必要があります。

- Red Hat OpenShift Data Foundation では、シークレットは以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_RH_DATA_FOUNDATION_BUCKET
      endpoint: YOUR_RH_DATA_FOUNDATION_ENDPOINT
```

```
    insecure: false
    access_key: YOUR_RH_DATA_FOUNDATION_ACCESS_KEY
    secret_key: YOUR_RH_DATA_FOUNDATION_SECRET_KEY
```

詳細は、「Red Hat OpenShift Data Foundation」を参照してください。

- Red Hat OpenShift on IBM (ROKS) では、シークレットは以下のファイルのようになります。

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_ROKS_S3_BUCKET
      endpoint: YOUR_ROKS_S3_ENDPOINT
      insecure: true
      access_key: YOUR_ROKS_ACCESS_KEY
      secret_key: YOUR_ROKS_SECRET_KEY
```

詳細は、IBM Cloud のドキュメント「Cloud Object Storage」を参照してください。サービスの認証情報を使用してオブジェクトストレージに接続するようにしてください。詳細は、IBM Cloud のドキュメント、Cloud Object Store および Service Credentials を参照してください。

5. 以下のコマンドを使用して、クラウドプロバイダーの S3 アクセスキーおよびシークレットキーを取得できます。

```
YOUR_CLOUD_PROVIDER_ACCESS_KEY=$(oc -n open-cluster-management-observability get secret <object-storage-secret> -o jsonpath="{.data.thanos\.yaml}" | base64 --decode | grep access_key | awk '{print $2}')

echo $ACCESS_KEY

YOUR_CLOUD_PROVIDER_SECRET_KEY=$(oc -n open-cluster-management-observability get secret <object-storage-secret> -o jsonpath="{.data.thanos\.yaml}" | base64 --decode | grep secret_key | awk '{print $2}')

echo $SECRET_KEY
```

シークレットの **base64** 文字列のデコード、編集、エンコードが必要です。

1.2.2.1. MultiClusterObservability CR の作成

以下の手順を実行して、マネージドクラスターの **MultiClusterObservability** カスタムリソース (CR) を作成します。

1. **multiclusterobservability_cr.yaml** という名前の **MultiClusterObservability** カスタムリソースの YAML ファイルを作成します。
   可観測性については、以下のデフォルト YAML ファイルを確認してください。

```
apiVersion: observability.open-cluster-management.io/v1beta2
kind: MultiClusterObservability
metadata:
  name: observability
spec:
  observabilityAddonSpec: {}
  storageConfig:
    metricObjectStorage:
      name: thanos-object-storage
      key: thanos.yaml
```

advanced セクションで **retentionConfig** パラメーターの値を変更する必要がある場合があります。詳細は、Thanos Downsampling resolution and retention を参照してください。マネージドクラスターの数によっては、ステートフルセットのストレージ容量を更新する必要がある場合があります。詳細は、Observability API を参照してください。

2. インフラストラクチャーマシンセットにデプロイするには、**MultiClusterObservability** YAML の **nodeSelector** を更新して、セットのラベルを設定する必要があります。YAML の内容は以下のようになります。

```
nodeSelector:
  node-role.kubernetes.io/infra:
```

詳細は、インフラストラクチャーマシンセットの作成 を参照してください。

3. 以下のコマンドを実行して可観測性 YAML をクラスターに適用します。

```
oc apply -f multiclusterobservability_cr.yaml
```

Thanos、Grafana および AlertManager の **open-cluster-management-observability** namespace に全 Pod を作成します。Red Hat Advanced Cluster Management ハブクラスターに接続されたマネージドクラスターはすべて、メトリクスを Red Hat Advanced Cluster Management の可観測性サービスに送信できます。

4. Grafana ダッシュボードを起動して、可観測性サービスが有効になっており、データが生成されていることを確認します。コンソールの **Overview** ページまたは **Clusters** ページのいずれかから、コンソールヘッダーの近くにある **Grafana リンク** をクリックします。
注記: 可観測性データを収集しないように特定のマネージドクラスターを除外するには、クラスターに **observability: disabled** のクラスターラベルを追加します。

可観測性サービスを有効化します。可観測性サービスを有効にしたら、以下の機能が開始されます。

- マネージドクラスターからのアラートマネージャーはすべて、Red Hat Advanced Cluster Management ハブクラスターに転送されます。

- Red Hat Advanced Cluster Management ハブクラスターに接続されたマネージドクラスターはすべて、アラートを Red Hat Advanced Cluster Management の可観測性サービスに送信できます。Red Hat Advanced Cluster Management Alertmanager を設定して、重複を排除してグループ化し、アラートをメール、PagerDuty、または OpsGenie などの適切なレシーバー統合にルーティングすることができます。アラートの通知解除や抑制にも対応できます。
注記: Red Hat Advanced Cluster Management ハブクラスター機能へのアラート転送は、Red Hat OpenShift Container Platform バージョン 4.8 以降のマネージドクラスターでのみサポートされます。可観測性を有効にして Red Hat Advanced Cluster Management をインストールすると、OpenShift Container Platform v4.8 以降のアラートは自動的にハブクラスターに転送されます。

詳細は、「送信アラート」を参照してください。

1.2.3. Red Hat OpenShift Container Platform コンソールからの可観測性の有効化

オプションで、Red Hat OpenShift Container Platform コンソールから可観測性を有効にし、**open-cluster-management-observability** という名前のプロジェクトを作成します。**open-cluster-management-observability** プロジェクトに、**multiclusterhub-operator-pull-secret** という名前のイメージプルシークレットを作成してください。

open-cluster-management-observability プロジェクトに **thanos-object-storage** という名前のオブジェクトストレージシークレットを作成します。オブジェクトストレージシークレットの詳細を入力し、**Create** をクリックします。**注記:** 可観測性の有効化 セクションの手順 4 を参照して、シークレットの例を確認してください。

MultiClusterObservability CR インスタンスを作成します。**Observability components are deployed and running** のメッセージが表示されると、OpenShift Container Platform から可観測性サービスが正常に有効化されています。

1.2.3.1. 外部メトリクスクエリーの使用

可観測性には、外部 API があり、OpenShift ルート (**rbac-query-proxy**) を使用してメトリクスをクエリーできます。以下のタスクを確認して、**rbac-query-proxy** ルートを使用します。

- 以下のコマンドを使用して、ルートの詳細を取得できます。

  ```
  oc get route rbac-query-proxy -n open-cluster-management-observability
  ```

- **rbac-query-proxy** ルートにアクセスするには、OpenShift OAuth アクセストークンが必要です。トークンは、namespace 取得のパーミッションがあるユーザーまたはサービスアカウントと関連付ける必要があります。詳細は、ユーザーが所有する OAuth アクセストークンの管理 を参照してください。

- デフォルトの CA 証明書を取得し、**tls.crt** キーの内容をローカルファイルに保存します。以下のコマンドを実行します。

  ```
  oc -n openshift-ingress get secret router-certs-default -o jsonpath="{.data.tls\.crt}" | base64 -d > ca.crt
  ```

- 以下のコマンドを実行してメトリクスのクエリーを実行します。

  ```
  curl --cacert ./ca.crt -H "Authorization: Bearer {TOKEN}" https://{PROXY_ROUTE_URL}/api/v1/query?query={QUERY_EXPRESSION}
  ```

  注記: TheQUERY_EXPRESSION は標準の Prometheus クエリー式です。たとえば、**cluster_infrastructure_provider** メトリクスのクエリーには、前述したコマンドの URL を **https://{PROXY_ROUTE_URL}/api/v1/query?query=cluster_infrastructure_provider** の URL に置き換えます。詳細は、「prometheus のクエリー」を参照してください。

- **rbac-query-proxy** ルートの証明書を置き換えることもできます。

  - 証明書を生成するための OpenSSL コマンド を参照して、証明書を作成します。**csr.cnf** をカスタマイズする時に、**DNS.1** を **rbac-query-proxy** ルートのホスト名に更新します。

  - 以下のコマンドを実行して、生成された証明書を使用して **proxy-byo-ca** および **proxy-byo-cert** シークレットを作成します。

```
oc -n open-cluster-management-observability create secret tls proxy-byo-ca --cert
./ca.crt --key ./ca.key

oc -n open-cluster-management-observability create secret tls proxy-byo-cert --cert
./ingress.crt --key ./ingress.key
```

1.2.4. 可観測性の無効化

可観測性 リソースをアンインストールして、可観測性サービスを無効にします。OpenShift Container Platform コンソールナビゲーションから、**Operators** > **Installed Operators** > **Advanced Cluster Manager for Kubernetes** の順に選択します。**MultiClusterObservability** カスタムリソースを削除します。

可観測性サービスのカスタマイズ方法の詳細は、「可観測性のカスタマイズ」を参照してください。

1.3. 可観測性のカスタマイズ

可観測性サービスが収集するデータのカスタマイズ、管理、および表示については、以下のセクションを参照してください。

must-gather コマンドで可観測性リソース用に作成される新規情報についてのログを収集します。詳細は、トラブルシューティング ドキュメントの **Must-gather** セクションを参照してください。

- カスタムルールの作成
- AlertManager の設定
- カスタムメトリクスの追加
- デフォルトメトリクスの削除
- **詳細** 設定の追加
- コンソールからの **multiclusterobservability** CR レプリカの更新
- アラートの転送
- ルート認定のカスタマイズ
- データの表示および展開
  - etcd テーブルの表示
  - Kubernetes API サーバーダッシュボードのクラスターフリートサービスレベルの概要の表示
  - Kubernetes API サーバーダッシュボードのクラスターサービスレベルの概要の表示
- 可観測性の無効化

1.3.1. カスタムルールの作成

可観測性リソースに、Prometheus レコードルール および アラートルール を追加して、可観測性インストールのカスタムルールを作成します。詳細は、Prometheus configuration を参照してください。

- レコードルールでは、必要に応じてコストの掛かる式を事前に計算するか、コンピュートできます。結果は新たな時系列のセットとして保存されます。
- アラートルールでは、アラートを外部サービスに送信する方法に基づいてアラート条件を指定する機能を提供します。

Prometheus でカスタムルールを定義してアラート条件を作成し、通知を外部メッセージングサービスに送信します。**注記**: カスタムルールを更新すると、**observability-thanos-rule** Pod は自動的に再起動されます。

open-cluster-management-observability namespace に **thanos-ruler-custom-rules** という名前の ConfigMap を作成します。以下の例のように、キーは **custom_rules.yaml** という名前を指定する必要があります。設定には、複数のルールを作成できます。

- デフォルトでは、同梱のアラートルールは **open-cluster-management-observability** namespace の **thanos-ruler-default-rules** ConfigMap に定義されます。
  たとえば、CPU の使用状況が定義値を超えた場合に通知するカスタムのアラートルールを作成できます。YAML の内容は以下のようになります。

```
data:
  custom_rules.yaml: |
    groups:
      - name: cluster-health
        rules:
        - alert: ClusterCPUHealth-jb
          annotations:
            summary: Notify when CPU utilization on a cluster is greater than the defined
utilization limit
            description: "The cluster has a high CPU usage: {{ $value }} core for {{ $labels.cluster
}} {{ $labels.clusterID }}."
          expr: |
            max(cluster:cpu_usage_cores:sum) by (clusterID, cluster, prometheus) > 0
          for: 5s
          labels:
            cluster: "{{ $labels.cluster }}"
            prometheus: "{{ $labels.prometheus }}"
            severity: critical
```

- **thanos-ruler-custom-rules** ConfigMap 内にカスタムの録画ルールを作成することもできます。
  たとえば、Pod のコンテナーメモリーキャッシュの合計を取得できるようにする記録ルールを作成することができます。YAML の内容は以下のようになります。

```
data:
  custom_rules.yaml: |
    groups:
      - name: container-memory
        rules:
        - record: pod:container_memory_cache:sum
          expr: sum(container_memory_cache{pod!=""}) BY (pod, container)
```

  注記: これが最初の新規カスタムルールである場合には、すぐに作成されます。ConfigMap に変更が加えられると、設定は自動的に再読み込みされます。この設定は、**observability-thanos-ruler** サイドカー内の **config-reload** により再読み込みされます。

アラートルールが適切に機能していることを確認するには、Grafana ダッシュボードを起動し、**Explore** ページに移動し、**ALERTS** にクエリーを実行します。アラートは、アラートが開始された場合に Grafana でのみ利用できます。

1.3.2. AlertManager の設定

メール、Slack、PagerDuty などの外部メッセージングツールを統合し、AlertManager から通知を受信します。**open-cluster-management-observability** namespace で **alertmanager-config** シークレットを上書きして、統合を追加し、AlertManager のルートを設定します。以下の手順を実行して、カスタムのレシーバールールを更新します。

1. **alertmanager-config** シークレットからデータを抽出します。以下のコマンドを実行します。

```
oc -n open-cluster-management-observability get secret alertmanager-config --template='{{ index .data "alertmanager.yaml" }}' |base64 -d > alertmanager.yaml
```

2. 以下のコマンドを実行し、**alertmanager.yaml** ファイル設定を編集して保存します。

```
oc -n open-cluster-management-observability create secret generic alertmanager-config --from-file=alertmanager.yaml --dry-run -o=yaml |  oc -n open-cluster-management-observability replace secret --filename=-
```

更新したシークレットは以下の内容のようになります。

```
global
  smtp_smarthost: 'localhost:25'
  smtp_from: 'alertmanager@example.org'
  smtp_auth_username: 'alertmanager'
  smtp_auth_password: 'password'
templates:
- '/etc/alertmanager/template/*.tmpl'
route:
  group_by: ['alertname', 'cluster', 'service']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 3h
  receiver: team-X-mails
  routes:
  - match_re:
      service: ^(foo1|foo2|baz)$
    receiver: team-X-mails
```

変更内容は、変更後すぐに適用されます。AlertManager の例については、prometheus/alertmanager を参照してください。

1.3.3. カスタムメトリクスの追加

metrics_list.yaml ファイルにメトリクスを追加して、マネージドクラスターから収集されるようにします。

カスタムメトリクスを追加する前に、**oc get mco observability -o yaml** コマンドで、**mco observability** が有効になっていることを確認します。**status.conditions.message** の メッセージが **Observability components are deployed and running** となっていることを確認します。

observability-metrics-custom-allowlist.yaml という名前のファイルを作成し、**metrics_list.yaml** パラメーターにカスタムメトリクスの名前を追加します。ConfigMap の YAML は、以下の内容のようになります。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: observability-metrics-custom-allowlist
data:
  metrics_list.yaml: |
    names:
      - node_memory_MemTotal_bytes
    rules:
    - record: apiserver_request_duration_seconds:histogram_quantile_90
      expr:
histogram_quantile(0.90,sum(rate(apiserver_request_duration_seconds_bucket{job=\"apiserver\",
      verb!=\"WATCH\"}[5m])) by (verb,le))
```

- **names** セクションで、マネージドクラスターから収集されるカスタムメトリクスの名前を追加します。
- **rules** セクションで、**expr** および **record** パラメーターペアに値を 1 つだけ入力し、クエリー式を定義します。メトリクスは、マネージドクラスターの **record** パラメーターで定義される名前で収集されます。クエリー式の実行後の結果が、メトリクスの値として返されます。
- **names** と **rules** セクションはオプションです。セクションのいずれかまたは両方を使用できます。

oc apply -n open-cluster-management-observability -f observability-metrics-custom-allowlist.yaml のコマンドで、**open-cluster-management-observability** namespace に **observability-metrics-custom-allowlist** ConfigMap を作成します。

Grafana ダッシュボードから **Explore** ページからメトリクスをクエリーし、カスタムメトリクスからのデータが収集されていることを確認します。独自のダッシュボードでカスタムメトリクスを使用することもできます。ダッシュボードの表示に関する詳細は、Grafana ダッシュボードの設計 を参照してください。

1.3.4. デフォルトメトリクスの削除

マネージドクラスターで特定のメトリクス用にデータを収集しない場合は、**observability-metrics-custom-allowlist.yaml** ファイルからメトリクスを削除します。メトリクスを削除すると、メトリクスデータはマネージドクラスターでは収集されません。前述したように、**mco observability** が有効になっていることを確認します。

メトリクス名の先頭にハイフン **-** を指定して **metrics_list.yaml** パラメーターにデフォルトのメトリクス名を追加します。例： **-cluster_infrastructure_provider**

oc apply -n open-cluster-management-observability -f observability-metrics-custom-allowlist.yaml のコマンドで、**open-cluster-management-observability** namespace に **observability-metrics-custom-allowlist** ConfigMap を作成します。

特定のメトリクスがマネージドクラスターから収集されていないことを確認します。Grafana ダッシュボードからメトリクスをクエリーしても、メトリクスは表示されません。

1.3.5. 詳細 設定の追加

advanced 設定セクションを追加して、必要に応じて可観測性コンポーネントごとに保持内容を更新します。

MultiClusterObservability CR を編集し、**oc edit mco observability -o yaml** コマンドで **advanced** セクションを追加します。YAML ファイルは以下の内容のようになります。

```
spec:
  advanced:
    retentionConfig:
      blockDuration: 2h
      deleteDelay: 48h
      retentionInLocal: 24h
      retentionResolutionRaw: 30d
      retentionResolution5m: 180d
      retentionResolution1h: 0d
    receive:
      resources:
        limits:
          memory: 4096Gi
      replicas: 3
```

advanced 設定に追加できるすべてのパラメーターの説明は、Observability API を参照してください。

1.3.6. コンソールからの multiclusterobservability CR レプリカの更新

ワークロードが増加する場合は、可観測性 Pod のレプリカ数を増やします。ハブクラスターから Red Hat OpenShift Container Platform コンソールに移動します。**multiclusterobservability** カスタムリソース(CR)を見つけ、レプリカを変更するコンポーネントの **replicas** パラメーターの値を更新します。更新した YAML は以下の内容のようになります。

```
spec:
   advanced:
      receive:
         replicas: 6
```

observability CR 内のパラメーターの詳細は、Observability API を参照してください。

1.3.7. アラートの転送

可観測性を有効にした後には、OpenShift Container Platform マネージドクラスターからのアラートは自動的にハブクラスターに送信されます。**alertmanager-config** YAML ファイルを使用して、外部通知システムでアラートを設定できます。

alertmanager-config YAML ファイルの例を以下に示します。

```
global:
  slack_api_url: '<slack_webhook_url>'

route:
  receiver: 'slack-notifications'
  group_by: [alertname, datacenter, app]

receivers:
- name: 'slack-notifications'
```

```
  slack_configs:
  - channel: '#alerts'
    text: 'https://internal.myorg.net/wiki/alerts/{{ .GroupLabels.app }}/{{ .GroupLabels.alertname }}'
```

アラート転送用のプロキシーを設定する場合は、**alertmanager-config** YAML ファイルに次の **global** エントリーを追加します。

```
global:
  slack_api_url: '<slack_webhook_url>'
  http_config:
    proxy_url: http://****
```

詳細は、Prometheus Alertmanager のドキュメント を参照してください。

1.3.8. ルート認定のカスタマイズ

OpenShift Container Platform ルート認証をカスタマイズする場合は、ルートを **alt_names** セクションに追加する必要があります。OpenShift Container Platform ルートにアクセスできるようにするには、**alertmanager.apps.<domainname>**、**observatorium-api.apps.<domainname>**、**rbac-query-proxy.apps.<domainname>** の情報を追加します。

1.3.9. データの表示および展開

ハブクラスターから Grafana にアクセスして、マネージドクラスターからデータを表示します。特定のアラートを照会して、そのクエリーのフィルターを追加できます。

たとえば、単一ノードクラスターから **cluster_infrastructure_provider** をクエリーするには、以下のクエリー式 **cluster_infrastructure_provider{clusterType="SNO"}**を使用します。

注記: 単一ノードのマネージドクラスターで可観測性が有効になっている場合は、**ObservabilitySpec.resources.CPU.limits** パラメーターを設定しないでください。CPU 制限を設定すると、可観測性 Pod がマネージドクラスターの容量にカウントされます。詳細は、「管理ワークロードのパーティション設定」を参照してください。

1.3.9.1. etcd テーブルの表示

Grafana のハブクラスターダッシュボードから etcd テーブルを表示し、データストアとしての etcd の安定性を確認します。

ハブクラスターから Grafana リンクを選択して、ハブクラスターから収集された **etcd** テーブルデータを表示します。マネージドクラスターの **Leader election changes** が表示されます。

1.3.9.2. Kubernetes API サーバーダッシュボードのクラスターフリートサービスレベルの概要の表示

Grafana のハブクラスターダッシュボードから、Kubernetes API サービスレベルの概要を表示します。

Grafana ダッシュボードに移動した後に、**Kubernetes** > **Service-Level Overview** > **API Server** を選択して管理ダッシュボードメニューにアクセスします。**Fleet Overview** および **Top Cluster** の詳細が表示されます。

過去 7 日間または 30 日間のターゲットとする **サービスレベル目標** (SLO) 値を超えるか、または満たしているクラスターの合計数、オフラインクラスター、および API サーバー要求の期間を表示します。

1.3.9.3. Kubernetes API サーバーダッシュボードのクラスターサービスレベルの概要の表示

Grafana のハブクラスターダッシュボードから Kubernetes API サービスレベルの概要テーブルを表示します。

Grafana ダッシュボードに移動した後に、**Kubernetes** > **Service-Level Overview** > **API Server** を選択して管理ダッシュボードメニューにアクセスします。**Fleet Overview** および **Top Cluster** の詳細が表示されます。

過去 7 日間または 30 日間のエラーとなっている予算、残りのダウンタイム、および傾向を表示します。

1.3.10. 可観測性の無効化

可観測性を無効にして、Red Hat Advanced Cluster Management ハブクラスターでデータ収集を停止します。

1.3.10.1. 全クラスターでの可観測性の無効化

すべてのマネージドクラスターで可観測性コンポーネントを削除して、可観測性を無効にします。

enableMetrics を **false** に設定して、**multicluster-observability-operator** リソースを更新します。更新されたリソースは、以下のような変更内容になります。

```
spec:
  imagePullPolicy: Always
  imagePullSecret: multiclusterhub-operator-pull-secret
  observabilityAddonSpec: # The ObservabilityAddonSpec defines the global settings for all managed clusters which have observability add-on enabled
    enableMetrics: false #indicates the observability addon push metrics to hub server
```

1.3.10.2. 単一クラスターの可観測性の無効化

特定のマネージドクラスターの可観測性コンポーネントを削除して可観測性を無効にします。**managedclusters.cluster.open-cluster-management.io** のカスタムリソースに **observability: disabled** ラベルを追加します。

Red Hat Advanced Cluster Management コンソールの **Clusters** ページから、指定したクラスターに **observability=disabled** ラベルを追加します。

注記: 可観測性コンポーネントが含まれるマネージドクラスターをデタッチすると、**metrics-collector** デプロイメントが削除されます。

可観測性サービスを使用したコンソールでのデータの監視に関する詳細は、「環境の監視の紹介」を参照してください。

1.4. GRAFANA ダッシュボードの設計

grafana-dev インスタンスを作成して、Grafana ダッシュボードを設計できます。

- Grafana 開発者インスタンスの設定
- Grafana ダッシュボードの設計
- Grafana 開発者インスタンスのアンインストール

1.4.1. Grafana 開発者インスタンスの設定

まず、**stolostron/multicluster-observability-operator/** リポジトリーのクローンを作成し、**tools** フォルダーにあるスクリプトを実行できるようにします。Grafana 開発者インスタンスを設定するには、以下の手順を実行します。

1. **setup-grafana-dev.sh** を実行して、Grafana インスタンスを設定します。スクリプトを実行すると、**secret/grafana-dev-config**、**deployment.apps/grafana-dev**、**service/grafana-dev**、**ingress.extensions/grafana-dev**、**persistentvolumeclaim/grafana-dev** のリソースが作成されます。

```
./setup-grafana-dev.sh --deploy
secret/grafana-dev-config created
deployment.apps/grafana-dev created
service/grafana-dev created
ingress.extensions/grafana-dev created
persistentvolumeclaim/grafana-dev created
```

2. **switch-to-grafana-admin.sh** スクリプトを使用して、ユーザーロールを Grafana 管理者に切り替えます。

   a. Grafana の URL **https://$ACM_URL/grafana-dev/** を選択して、ログインします。

   b. 次に、以下のコマンドを実行して、切り替えユーザーを Grafana 管理者として追加します。たとえば、**kubeadmin** を使用してログインしたら、以下のコマンドを実行します。

```
./switch-to-grafana-admin.sh kube:admin
User <kube:admin> switched to be grafana admin
```

Grafana 開発者インスタンを設定します。

1.4.2. Grafana ダッシュボードの設計

Grafana インスタンスを設定したら、ダッシュボードを設計できます。Grafana コンソールを更新し、ダッシュボードを設計するには、以下の手順を実行します。

1. Grafana コンソールのナビゲーションパネルから **Create** アイコンを選択してダッシュボードを作成します。**Dashboard** を選択し、**Add new panel** をクリックします。
2. **New Dashboard/Edit Panel** ビューで、**Query** タブを選択します。
3. データソースセレクターから **Observatorium** を選択し、PromQL クエリーを入力してクエリーを設定します。
4. Grafana ダッシュボードヘッダーから、ダッシュボードヘッダーにある **Save** アイコンをクリックします。
5. 説明的な名前を追加し、**Save** をクリックします。

1.4.2.1. ConfigMap での Grafana ダッシュボードの設計

ConfigMap で Grafana ダッシュボードを設計するには、以下の手順を実行します。

1. **generate-dashboard-configmap-yaml.sh** スクリプトを使用してダッシュボードの ConfigMap を生成し、ローカルで ConfigMap を保存できます。

```
./generate-dashboard-configmap-yaml.sh "Your Dashboard Name"
Save dashboard <your-dashboard-name> to ./your-dashboard-name.yaml
```

前述のスクリプトを実行するパーミッションがない場合は、以下の手順を実行します。

a. ダッシュボードを選択し、**Dashboard 設定** アイコンをクリックします。

b. ナビゲーションパネルから **JSON Model** アイコンをクリックします。

c. ダッシュボード JSON データをコピーし、**data** セクションに貼り付けます。

d. **name** を、**$your-dashboard-name** に置き換えます。ConfigMap は、以下のファイルのようになります。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: $your-dashboard-name
  namespace: open-cluster-management-observability
  labels:
    grafana-custom-dashboard: "true"
data:
  $your-dashboard-name.json: |-
    $your_dashboard_json
```

注記: ダッシュボードが **General** フォルダーにない場合は、この ConfigMap の **annotations** セクションにフォルダー名を指定できます。

```
annotations:
  observability.open-cluster-management.io/dashboard-folder: Custom
```

ConfigMap の更新が完了したら、インストールしてダッシュボードを Grafana インスタンスにインポートできます。

1.4.3. Grafana 開発者インスタンスのアンインストール

インスタンスをアンインストールすると、関連するリソースも削除されます。以下のコマンドを実行します。

```
./setup-grafana-dev.sh --clean
secret "grafana-dev-config" deleted
deployment.apps "grafana-dev" deleted
service "grafana-dev" deleted
ingress.extensions "grafana-dev" deleted
persistentvolumeclaim "grafana-dev" deleted
```

1.5. RED HAT INSIGHTS の可観測性

Red Hat Insights は、Red Hat Advanced Cluster Management 可観測性と統合されており、クラスター内の既存の問題や発生しうる問題を特定できるように有効化されています。Red Hat Insights は、安定性、パフォーマンス、ネットワーク、およびセキュリティーリスクの特定、優先順位付け、および解決に役立ちます。Red Hat OpenShift Container Platform は、OpenShift Cluster Manager を使用してクラ

スターのヘルスモニタリングを提供します。OpenShift Cluster Manager はクラスターの健全性、使用状況、およびサイズに関する匿名化され、集計された情報を収集します。詳細は、Red Hat Insights の製品ドキュメント を参照してください。

OpenShift クラスターを作成またはインポートすると、マネージドクラスターからの匿名データは自動的に Red Hat に送信されます。この情報を使用してクラスターのヘルス情報を提供する insights を作成します。Red Hat Advanced Cluster Management 管理者は、このヘルス情報を使用して重大度に基づいてアラートを作成できます。

必要なアクセス権限:: クラスターの管理者

1.5.1. 前提条件

- Red Hat Insights が有効になっていることを確認する。詳細は、グローバルクラスタープルシークレットの変更によるリモートヘルスレポートの無効化 を参照してください。
- OpenShift Container Platform バージョン 4.0 以降をインストールしておく。
- OpenShift Cluster Manager に登録されているハブクラスターユーザーが OpenShift Cluster Manager の全 Red Hat Advanced Cluster Management マネージドクラスターを管理できる。

1.5.2. Red Hat Advanced Cluster Management コンソールからの Red Hat Insights

以下で、統合に関する機能の説明を確認します。

- **Clusters** ページからクラスターを選択すると、**Status** カードから **特定された問題の数** を選択できます。**Status** カードには、**ノード**、**アプリケーション**、**ポリシー違反** および **特定された問題** に関する情報が表示されます。**Identified issues** カードは、Red Hat Insights からの情報を表します。**Identified issues** のステータスには、重大度による問題数が表示されます。問題の対応レベルは、**Critical**、**Major**、**Low** および **Warning** の重大度に分類されます。
- 数字をクリックすると、**Potential issue** のサイドパネルが表示されます。パネルにすべての問題の概要およびチャートが表示されます。検索機能を使用して、推奨される修復を検索することもできます。修復オプションは、脆弱性の **説明**、脆弱性に関連する **カテゴリー**、および **全体的なリスク** を表示します。
- **説明** セクションから、脆弱性へのリンクを選択できます。**How to remediate** タブを選択して脆弱性を解決するための手順を表示します。**Reason** タブをクリックすると、脆弱性が発生した理由を確認することもできます。

詳細は、「Insight **PolicyReports**の管理」を参照してください。

1.6. INSIGHTS POLICYREPORTS の管理

Red Hat Advanced Cluster Management for Kubernetes **PolicyReports** は、**insights-client** で生成される違反です。**PolicyReports** は、インシデント管理システムに送信されるアラートの定義および設定に使用されます。違反がある場合には、**PolicyReport** からのアラートはインシデント管理システムに送信されます。

Insight **PolicyReports** の管理および表示方法については、以下のセクションを参照してください。

- Insight ポリシーレポートの検索
- コンソールから特定された問題の表示

1.6.1. Insight ポリシーレポートの検索

マネージドクラスター全体で、違反した特定の insight **PolicyReport** を検索できます。

Red Hat Advanced Cluster Management ハブクラスターにログインしたら、コンソールヘッダーの **Search** アイコンをクリックして **Search** ページに移動します。**kind:policyreport** のクエリーを入力します。

注記: PolicyReport 名はクラスターの名前と同じになります。

また、クエリーは、insight ポリシー違反およびカテゴリー別にさらに指定することもできます。**PolicyReport** 名を選択すると、関連付けられたクラスターの **Details** ページにリダイレクトされます。**Insights** サイドバーが自動的に表示されます。

検索サービスが無効になり、insight を検索する必要がある場合は、ハブクラスターから以下のコマンドを実行します。

+

```
oc get policyreport --all-namespaces
```

1.6.2. コンソールから特定された問題の表示

特定のクラスターで特定された問題を表示できます。

Red Hat Advanced Cluster Management クラスターにログインしたら、ナビゲーションメニューから **Overview** を選択します。重大度を選択して、対象の重大度に関連付けられた **PolicyReports** を表示します。**クラスターの問題** の概要カードから、クラスターの問題と重要性の詳細を表示します。

または、ナビゲーションメニューから **Clusters** を選択できます。テーブルからマネージドクラスターを選択して、詳細情報を表示します。**Status** カードから、特定された問題の数を表示します。

発生する可能性のある問題数を選択して、重大度チャートと、その問題に対して推奨される修復を表示します。脆弱性へのリンクをクリックすると、「**修復する方法**」と脆弱性の「**理由**」の手順を表示します。

注記: 問題の解決後には、30 分ごとに Red Hat Advanced Cluster Management により Red Hat Insights を受信し、Red Hat Insights は 2 時間ごとに更新されます。

PolicyReport からアラートメッセージを送信したコンポーネントを確認してください。**ガバナンス** ページに移動し、特定の **ポリシーレポート** を選択します。**Status** タブを選択し、**View details** リンクをクリックして **PolicyReport** YAML ファイルを表示します。

source パラメーターを見つけます。このパラメーターにより、違反を送信したコンポーネントが通知されます。値オプションは **grc** および **insights** です。

PolicyReports にカスタムアラートルールを作成する方法は、AlertManager の設定を参照してください。