Red Hat Advanced Cluster Management for Kubernetes 2.10

トラブルシューティング

トラブルシューティング

Last Updated: 2024-09-14

Red Hat Advanced Cluster Management for Kubernetes 2.10 トラブルシューティング

トラブルシューティング

法律上の通知

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

クラスターのトラブルシューティングトピックのリストを確認します。また、must-gather コマンドを使用してログを収集することもできます。

目次

第1章 トラブルシューティング

トラブルシューティングガイドをご使用の前に **oc adm must-gather** コマンドを実行して、詳細およびログを収集し、問題のデバッグ手順を行います。詳細は、must-gather コマンドを実行したトラブルシューティング を参照してください。

また、ロールベースのアクセス権限を確認してください。詳細は、ロールベースのアクセス制御 を参照してください。

1.1. 文書化されたトラブルシューティング

以下に、Red Hat Advanced Cluster Management for Kubernetes のトラブルシューティングのトピックリストを表示します。

インストール

インストールタスク関連の主なドキュメントは、インストールとアップグレード です。

- インストールステータスがインストールまたは保留中の状態のトラブルシューティング
- 再インストールに失敗する場合のトラブルシューティング
- Red Hat Advanced Cluster Management のアップグレード後に ocm-controller エラーのトラブルシューティング

バックアップおよび復元

バックアップと復元に関する主なドキュメントは、バックアップと復元 です。

- 復元ステータスがエラーで終了する場合のトラブルシューティング
- ハブクラスターのバックアップを復元すると汎用リソースが削除される

クラスター管理

クラスターの管理に関する主なドキュメントは、Operator クラスターライフサイクルのマルチクラスターエンジンの概要 です。

- オフラインクラスターのトラブルシューティング
- マネージドクラスターのインポート失敗に関するトラブルシューティング
- Pending Import ステータスのクラスターのトラブルシューティング
- 証明書を変更した後のインポート済みクラスターのオフラインでのトラブルシューティング
- クラスターのステータスが offline から available に変わる場合のトラブルシューティング
- VMware vSphere でのクラスター作成のトラブルシューティング
- ステータスが Pending または Failed のクラスターのコンソールでのトラブルシューティング
- OpenShift Container Platform バージョン 3.11 クラスターのインポートの失敗時のトラブルシューティング
- degraded 状態にある Klusterlet のトラブルシューティング

- オブジェクトストレージチャネルシークレットのトラブルシューティング
- クラスターの削除後も namespace が残る
- クラスターのインポート時の auto-import-secret-exists エラー
- VolSync 用の cinder Container Storage Interface (CSI) ドライバーのトラブルシューティング
- アドオン許容を無視するクラスタープロキシーアドオンのトラブルシューティング

Multicluster Global Hub

Multicluster Global Hub に関する主要なドキュメントを表示するには、{global-hub を参照してください。

- must-gather コマンドを実行したトラブルシューティング
- トラブルシューティングのためにプロビジョニングされた postgres データベースにアクセスする
- トラブルシューティングのためのデータベースのダンプと復元の使用
- コンプライアンスデータの復元

アプリケーション管理

アプリケーション管理に関する主なドキュメントを表示するには、アプリケーション管理 を参照してください。

- アプリケーションの Kubernetes デプロイメントバージョンのトラブルシューティング
- ローカルクラスターが選択されていない問題のトラブルシューティング

ガバナンス

- 複数行の YAML 解析のトラブルシューティング

セキュリティーガイドを表示するには、Risk and compliance を参照してください。

コンソールの可観測性

コンソールの可観測性には、ヘッダーおよびナビゲーション機能、検索、および移動機能が含まれます。可観測性ガイドを表示するには、Observability in the console を参照してください。

- grafana のトラブルシューティング
- 可観測性のトラブルシューティング
- OpenShift モニタリングサービスのトラブルシューティング
- metrics-collector のトラブルシューティング
- PostgreSQL 共有メモリーエラーのトラブルシューティング
- Thanos コンパクターのブロックエラーのトラブルシューティング

Submariner のネットワーキングとサービスディスカバリー

このセクションでは、Red Hat Advanced Cluster Management またはマルチクラスターエンジン Operator で Submariner を使用する場合に発生する可能性のある Submariner のトラブルシューティング手順を示します。Submariner の一般的なトラブルシューティング情報は、Submariner のドキュメントの Troubleshooting を参照してください。

Submariner ネットワーキングサービスとサービスディスカバリの主なドキュメントは、Submariner multicluster networking and service discovery です。

- インストール後に Submariner が接続されない場合のトラブルシューティング - 一般情報
- Submariner アドオンのステータスが低下してい場合のトラブルシューティング

1.2. MUST-GATHER コマンドを実行したトラブルシューティング

トラブルシューティングを開始するには、問題のデバッグを行う **must-gather** コマンドを実行する場合のトラブルシューティングシナリオについて確認し、このコマンドの使用を開始する手順を参照してください。

必要なアクセス権限: クラスターの管理者

1.2.1. Must-gather のシナリオ

- **シナリオ 1:** 文書化されたトラブルシューティング セクションを使用して、問題の解決策がまとめられているかどうかを確認します。本ガイドは、製品の主な機能別に設定されています。このシナリオでは、解決策が本書にまとめられているかどうかを、本ガイドで確認します。たとえば、クラスターの作成に関するトラブルシューティングの場合は、**クラスターの管理** セクションの解決策を探します。
- **シナリオ 2:** 問題の解決策の手順が文書にまとめられていない場合は、**must-gather** コマンドを実行し、その出力を使用して問題をデバッグします。
- **シナリオ 3:** **must-gather** コマンドの出力を使用して問題をデバッグできない場合は、出力を Red Hat サポートに共有します。

1.2.2. Must-gather の手順

must-gather コマンドの使用を開始するには、以下の手順を参照してください。

1. **must-gather** コマンドについて確認し、Red Hat OpenShift Container Platform の クラスターに関するデータの収集 に必要な前提条件をインストールします。
2. クラスターにログインします。データおよびディレクトリーの収集に使用する Red Hat Advanced Cluster Management for Kubernetes イメージを追加します。以下のコマンドを実行して、出力用にイメージとディレクトリーを挿入します。

   ```
   oc adm must-gather --image=registry.redhat.io/rhacm2/acm-must-gather-rhel9:v2.10 --dest-dir=<directory>
   ```

3. 通常のユースケースでは、**ハブ** クラスターにログインして、**must-gather** を実行する必要があります。
   注記: マネージドクラスターを確認する場合は、**cluster-scoped-resources** ディレクトリーにある **gather-managed.log** ファイルを検索します。

   ```
   <your-directory>/cluster-scoped-resources/gather-managed.log>
   ```

JOINED および AVAILABLE 列に **True** が設定されていないマネージドクラスターがないかを確認します。**must-gather** コマンドは、ステータスが **True** として関連付けられていないクラスター上で、実行できます。

4. 指定したディレクトリーに移動し、以下のレベルに整理されている出力を確認します。
   - ピアレベル 2 つ: **cluster-scoped-resources** と **namespace** のリソース
   - それぞれに対するサブレベル: クラススコープおよび namespace スコープの両方のリソースに対するカスタムリソース定義の API グループ。
   - それぞれに対する次のレベル: **kind** でソートされた YAML ファイル

1.2.3. 非接続環境での must-gather

非接続環境で **must-gather** コマンドを実行するには、次の手順を実行します。

1. 非接続環境では、Red Hat Operator のカタログイメージをミラーレジストリーにミラーリングします。詳細は、ネットワーク切断状態でのインストール を参照してください。
2. 次のコマンドを実行してすべての情報を収集します。**<2.x>** は、**<acm-must-gather>** (例: **2.10**) と **<multicluster-engine/must-gather>** (例: **2.5)** の両方でサポートされているバージョンに置き換えます。

```
REGISTRY=<internal.repo.address:port>
IMAGE1=$REGISTRY/rhacm2/acm-must-gather-rhel9:v<2.x>
oc adm must-gather --image=$IMAGE1 --dest-dir=<directory>
```

現在サポートされているリリースのいずれか、製品ドキュメントで問題が発生した場合は、Red Hat サポート にアクセスして、さらにトラブルシューティングを行ったり、ナレッジベースの記事を表示したり、サポートチームに連絡したり、ケースを開いたりすることができます。Red Hat 認証情報でログインする必要があります。

1.2.4. ホステッドクラスターの must-gather

Hosted Control Plane クラスターで問題が発生した場合は、**must-gather** コマンドを実行して、トラブルシューティングに役立つ情報を収集できます。

1.2.4.1. ホステッドクラスターの must-gather コマンドについて

このコマンドは、管理クラスターとホストされたクラスターの出力を生成します。

- マルチクラスターエンジン Operator ハブクラスターからのデータ:
  - クラススコープのリソース: これらのリソースは、管理クラスターのノード定義です。
  - **hypershift-dump** 圧縮ファイル: このファイルは、コンテンツを他の人と共有する必要がある場合に役立ちます。
  - namespace リソース: これらのリソースには、config map、サービス、イベント、ログなど、関連する namespace のすべてのオブジェクトが含まれます。
  - ネットワークログ: これらのログには、OVN ノースバウンドデータベースとサウスバウンドデータベース、およびそれぞれのステータスが含まれます。

- ホストされたクラスター: このレベルの出力には、ホストされたクラスター内のすべてのリソースが含まれます。

- ホストされたクラスターからのデータ:
  - クラスタースコープのリソース: これらのリソースには、ノードや CRD などのクラスター全体のオブジェクトがすべて含まれます。
  - namespace リソース: これらのリソースには、config map、サービス、イベント、ログなど、関連する namespace のすべてのオブジェクトが含まれます。

出力にはクラスターからのシークレットオブジェクトは含まれませんが、シークレットの名前への参照が含まれる可能性があります。

1.2.4.2. 前提条件

must-gather コマンドを実行して情報を収集するには、次の前提条件を満たす必要があります。

- **kubeconfig** ファイルが読み込まれ、マルチクラスターエンジン Operator ハブクラスターを指している。
- マルチクラスターエンジン Operator ハブクラスターへの cluster-admin アクセスがある。
- **HostedCluster** リソースの name 値と、カスタムリソースがデプロイされる namespace がある。

1.2.4.3. ホスト型クラスターの must-gather コマンドの入力

1. 以下のコマンドを実行して、ホスト型クラスターに関する情報を収集します。このコマンドでは、**hosted-cluster-namespace=HOSTEDCLUSTERNAMESPACE** パラメーターはオプションです。これを含めない場合、コマンドは、ホストされたクラスターがデフォルトの namespace (**clusters**) 内にあるかのように実行されます。

```
oc adm must-gather --image=quay.io/stolostron/backplane-must-gather:SNAPSHOTNAME
/usr/bin/gather hosted-cluster-namespace=HOSTEDCLUSTERNAMESPACE hosted-cluster-
name=HOSTEDCLUSTERNAME
```

2. コマンドの結果を圧縮ファイルに保存するには、**--dest-dir=NAME** パラメーターを含めて、**NAME** は、結果を保存するディレクトリーの名前に置き換えます。

```
oc adm must-gather --image=quay.io/stolostron/backplane-must-gather:SNAPSHOTNAME
/usr/bin/gather hosted-cluster-namespace=HOSTEDCLUSTERNAMESPACE hosted-cluster-
name=HOSTEDCLUSTERNAME --dest-dir=NAME ; tar -cvzf NAME.tgz NAME
```

1.2.4.4. 非接続環境での must-gather コマンドの入力

非接続環境で **must-gather** コマンドを実行するには、次の手順を実行します。

1. 非接続環境では、Red Hat Operator のカタログイメージをミラーレジストリーにミラーリングします。詳細は、ネットワーク切断状態でのインストール を参照してください。

2. 次のコマンドを実行して、ミラーレジストリーからイメージを参照するログを抽出します。

```
REGISTRY=registry.example.com:5000
IMAGE=$REGISTRY/multicluster-engine/must-gather-
```

```
rhel8@sha256:ff9f37eb400dc1f7d07a9b6f2da9064992934b69847d17f59e385783c071b9d8

oc adm must-gather --image=$IMAGE /usr/bin/gather hosted-cluster-
namespace=HOSTEDCLUSTERNAMESPACE hosted-cluster-
name=HOSTEDCLUSTERNAME --dest-dir=./data
```

1.2.4.5. 関連情報

- Hosted Control Plane のトラブルシューティングの詳細は、OpenShift Container Platform ドキュメントの Hosted Control Plane のトラブルシューティング を参照してください。

1.3. インストールステータスがインストールまたは保留中の状態のトラブルシューティング

Red Hat Advanced Cluster Management のインストール時に、**MultiClusterHub** は **Installing** フェーズのままとなるか、複数の Pod が **Pending** ステータスのままとなります。

1.3.1. 現象: Pending 状態で止まる

MultiClusterHub をインストールしてから、**MultiClusterHub** リソースの **status.components** フィールドからのコンポーネントの 1 つ以上で **ProgressDeadlineExceeded** と報告したまま 10 分以上経過しています。クラスターのリソース制約が問題となっている場合があります。

Multiclusterhub がインストールされている namespace の Pod を確認します。以下のようなステータスとともに **Pending** と表示される場合があります。

```
reason: Unschedulable
message: '0/6 nodes are available: 3 Insufficient cpu, 3 node(s) had taint {node-
role.kubernetes.io/master:
        }, that the pod didn't tolerate.'
```

このような場合には、ワーカーノードにはクラスターでの製品実行に十分なリソースがありません。

1.3.2. 問題の解決: ワーカーノードのサイズの調整

この問題が発生した場合は、大規模なワーカーノードまたは複数のワーカーノードでクラスターを更新する必要があります。クラスターのサイジングのガイドラインについては、クラスターのサイジング を参照してください。

1.4. 再インストールに失敗する場合のトラブルシューティング

Red Hat Advanced Cluster Management for Kubernetes を再インストールすると Pod が起動しません。

1.4.1. 現象: 再インストールの失敗

Red Hat Advanced Cluster Management のインストール後に Pod が起動しない場合は、Red Hat Advanced Cluster Management 以前にインストールされており、今回のインストールを試行する前にすべてのパーツが削除されていない可能性があります。

Pod はこのような場合に、インストールプロセスの完了後に起動しません。

1.4.2. 問題の解決: 再インストールの失敗

この問題が発生した場合は、以下の手順を実行します。

1. アンインストール の手順に従い、現在のコンポーネントを削除し、アンインストールプロセスを実行します。
2. Helm のインストール の手順に従い、Helm CLI バイナリーバージョン 3.2.0 以降をインストールします。
3. **oc** コマンドが実行できるように、Red Hat OpenShift Container Platform CLI が設定されていることを確認してください。**oc** コマンドの設定方法の詳細は、OpenShift Container Platform ドキュメントの OpenShift CLI スタートガイド を参照してください。
4. 以下のスクリプトをファイルにコピーします。

```
#!/bin/bash
ACM_NAMESPACE=<namespace>
oc delete mch --all -n $ACM_NAMESPACE
oc delete apiservice v1.admission.cluster.open-cluster-management.io
v1.admission.work.open-cluster-management.io
oc delete clusterimageset --all
oc delete clusterrole multiclusterengines.multicluster.openshift.io-v1-admin
multiclusterengines.multicluster.openshift.io-v1-crdview
multiclusterengines.multicluster.openshift.io-v1-edit
multiclusterengines.multicluster.openshift.io-v1-view open-cluster-
management:addons:application-manager open-cluster-management:admin-aggregate open-
cluster-management:cert-policy-controller-hub open-cluster-management:cluster-manager-
admin-aggregate open-cluster-management:config-policy-controller-hub open-cluster-
management:edit-aggregate open-cluster-management:iam-policy-controller-hub open-
cluster-management:policy-framework-hub open-cluster-management:view-aggregate
oc delete crd klusterletaddonconfigs.agent.open-cluster-management.io
placementbindings.policy.open-cluster-management.io policies.policy.open-cluster-
management.io userpreferences.console.open-cluster-management.io
discoveredclusters.discovery.open-cluster-management.io discoveryconfigs.discovery.open-
cluster-management.io
oc delete mutatingwebhookconfiguration ocm-mutating-webhook
managedclustermutators.admission.cluster.open-cluster-management.io multicluster-
observability-operator
oc delete validatingwebhookconfiguration
channels.apps.open.cluster.management.webhook.validator application-webhook-validator
multiclusterhub-operator-validating-webhook ocm-validating-webhook multicluster-
observability-operator multiclusterengines.multicluster.openshift.io
```

   スクリプトの **<namespace>** は、Red Hat Advanced Cluster Management がインストールされている namespace 名に置き換えます。namespace が消去され削除されるため、正しい namespace を指定するようにしてください。

5. スクリプトを実行して、アーティファクトを以前のインストールから削除します。
6. インストールを実行します。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。

1.5. RED HAT ADVANCED CLUSTER MANAGEMENT のアップグレード後に OCM-CONTROLLER エラーのトラブルシューティング

2.7.x から 2.8.x、さらに 2.9.0 にアップグレードすると、**multicluster-engine** namespace の **ocm-controller** がクラッシュします。

1.5.1. 現象: Red Hat Advanced Cluster Management のアップグレード後に ocm-controller エラーのトラブルシューティング

ManagedClusterSet および **ManagedClusterSetBinding** カスタムリソース定義をリスト表示しようとすると、次のエラーメッセージが表示されます。

```
Error from server: request to convert CR from an invalid group/version: cluster.open-cluster-management.io/v1beta1
```

前のメッセージは、**v1beta1** から **v1beta2** への **ManagedClusterSets** および **ManagedClusterSetBindings** カスタムリソース定義の移行が失敗したことを示しています。

1.5.2. 問題の解決: Red Hat Advanced Cluster Management アップグレード後の ocm-controller エラーのトラブルシューティング

このエラーを解決するには、API の移行を手動で開始する必要があります。以下の手順を実行します。

1. **cluster-manager** を以前のリリースに戻します。
   a. 以下のコマンドを使用して **multiclusterengine** を一時停止します。

```
oc annotate mce multiclusterengine pause=true
```

   b. 以下のコマンドを実行して、**cluster-manager** deployment のイメージを以前のバージョンに置き換えます。

```
oc patch deployment cluster-manager -n multicluster-engine -p \  '{"spec":{"template":{"spec":{"containers":[{"name":"registration-operator","image":"registry.redhat.io/multicluster-engine/registration-operator-rhel8@sha256:35999c3a1022d908b6fe30aa9b85878e666392dbbd685e9f3edcb83e3336d19f"}]}}}}'
export ORIGIN_REGISTRATION_IMAGE=$(oc get clustermanager cluster-manager -o jsonpath='{.spec.registrationImagePullSpec}')
```

   c. **ClusterManager** リソースの登録イメージ参照を以前のバージョンに置き換えます。以下のコマンドを実行します。

```
oc patch clustermanager cluster-manager --type='json' -p='[{"op": "replace", "path": "/spec/registrationImagePullSpec", "value": "registry.redhat.io/multicluster-engine/registration-rhel8@sha256:a3c22aa4326859d75986bf24322068f0aff2103cccc06e1001faaf79b9390515"}]'
```

2. 以下のコマンドを実行して、**ManagedClusterSets** および **ManagedClusterSetBindings** カスタムリソース定義を以前のリリースに戻します。

```
oc annotate crds managedclustersets.cluster.open-cluster-management.io operator.open-cluster-management.io/version-
oc annotate crds  managedclustersetbindings.cluster.open-cluster-management.io operator.open-cluster-management.io/version-
```

3. **cluster-manager** を再起動し、カスタムリソース定義が再作成されるまで待ちます。以下のコマンドを実行します。

```
oc -n multicluster-engine delete pods -l app=cluster-manager
oc wait crds managedclustersets.cluster.open-cluster-management.io --for=jsonpath="
{.metadata.annotations['operator\.open-cluster-management\.io/version']}"="2.3.3" --
timeout=120s
oc wait crds managedclustersetbindings.cluster.open-cluster-management.io --
for=jsonpath="{.metadata.annotations['operator\.open-cluster-
management\.io/version']}"="2.3.3" --timeout=120s
```

4. 以下のコマンドを使用して、ストレージバージョンの移行を開始します。

```
oc patch StorageVersionMigration managedclustersets.cluster.open-cluster-management.io -
-type='json' -p='[{"op":"replace", "path":"/spec/resource/version", "value":"v1beta1"}]'
oc patch StorageVersionMigration managedclustersets.cluster.open-cluster-management.io -
-type='json' --subresource status -p='[{"op":"remove", "path":"/status/conditions"}]'
oc patch StorageVersionMigration managedclustersetbindings.cluster.open-cluster-
management.io --type='json' -p='[{"op":"replace", "path":"/spec/resource/version",
"value":"v1beta1"}]'
oc patch StorageVersionMigration managedclustersetbindings.cluster.open-cluster-
management.io --type='json' --subresource status -p='[{"op":"remove",
"path":"/status/conditions"}]'
```

5. 次のコマンドを実行して、移行が完了するまで待ちます。

```
oc wait storageversionmigration managedclustersets.cluster.open-cluster-management.io --
for=condition=Succeeded --timeout=120s
oc wait storageversionmigration managedclustersetbindings.cluster.open-cluster-
management.io --for=condition=Succeeded --timeout=120s
```

6. **cluster-manager** を Red Hat Advanced Cluster Management 2.10 に復元します。これには数分の時間がかかる場合があります。以下のコマンドを実行します。

```
oc annotate mce multiclusterengine pause-
oc patch clustermanager cluster-manager --type='json' -p='[{"op": "replace", "path":
"/spec/registrationImagePullSpec", "value": "'$ORIGIN_REGISTRATION_IMAGE'"}]'
```

1.5.2.1. 検証

Red Hat Advanced Cluster Management が復元されていることを確認するには、以下のコマンドを実行します。

```
oc get managedclusterset
oc get managedclustersetbinding -A
```

コマンドを実行すると、**ManagedClusterSets** および **ManagedClusterSetBindings** リソースがエラーメッセージなしで表示されます。

1.6. オフラインクラスターのトラブルシューティング

クラスターのステータスがオフラインと表示される一般的な原因がいくつかあります。

1.6.1. 現象: クラスターのステータスがオフライン状態である

クラスターの作成手順を完了したら、Red Hat Advanced Cluster Management コンソールからアクセスできず、クラスターのステータスが **offline** と表示されます。

1.6.2. 問題の解決: クラスターのステータスがオフライン状態になっている

1. マネージドクラスターが利用可能かどうかを確認します。これは、Red Hat Advanced Cluster Management コンソールの **Clusters** エリアで確認できます。
   利用不可の場合は、マネージドクラスターの再起動を試行します。

2. マネージドクラスターのステータスがオフラインのままの場合は、以下の手順を実行します。

   a. ハブクラスターで **oc get managedcluster <cluster_name> -o yaml** コマンドを実行します。**<cluster_name>** は、クラスター名に置き換えます。

   b. **status.conditions** セクションを見つけます。

   c. **type: ManagedClusterConditionAvailable** のメッセージを確認して、問題を解決します。

1.7. マネージドクラスターのインポート失敗に関するトラブルシューティング

クラスターのインポートに失敗した場合は、クラスターのインポートが失敗した理由を判別するためにいくつかの手順を実行できます。

1.7.1. 現象: インポートされたクラスターを利用できない

クラスターのインポート手順を完了したら、Red Hat Advanced Cluster Management for Kubernetes コンソールからアクセスできません。

1.7.2. 問題の解決: インポートされたクラスターが利用できない

インポートの試行後にインポートクラスターが利用できない場合には、いくつかの理由があります。クラスターのインポートに失敗した場合は、インポートに失敗した理由が見つかるまで以下の手順を実行します。

1. Red Hat Advanced Cluster Management ハブクラスターで以下のコマンドを実行し、Red Hat Advanced Cluster Management インポートコントローラーが実行していることを確認します。

   ```
   kubectl -n multicluster-engine get pods -l app=managedcluster-import-controller-v2
   ```

   実行中の Pod が 2 つ表示されるはずです。Pod のいずれかが実行されていない場合には、以下のコマンドを実行してログを表示して理由を判別します。

   ```
   kubectl -n multicluster-engine logs -l app=managedcluster-import-controller-v2 --tail=-1
   ```

2. Red Hat Advanced Cluster Management ハブクラスターで以下のコマンドを実行し、Red Hat Advanced Cluster Management インポートコントローラーでマネージドクラスターのインポートシークレットが正常に生成されたかどうかを確認します。

   ```
   kubectl -n <managed_cluster_name> get secrets <managed_cluster_name>-import
   ```

インポートシークレットが存在しない場合は、以下のコマンドを実行してインポートコントローラーのログエントリーを表示し、作成されていない理由を判断します。

```
kubectl -n multicluster-engine logs -l app=managedcluster-import-controller-v2 --tail=-1 | grep importconfig-controller
```

3. Red Hat Advanced Cluster Management ハブクラスターで、マネージドクラスターが **local-cluster** で、Hive によってプロビジョニングされるか、自動インポートシークレットがある場合は、以下のコマンドを実行してマネージドクラスターのインポートステータスを確認します。

```
kubectl get managedcluster <managed_cluster_name> -o=jsonpath='{range .status.conditions[*]}{.type}{"\t"}{.status}{"\t"}{.message}{"\n"}{end}' | grep ManagedClusterImportSucceeded
```

ManagedClusterImportSucceeded が **true** でない場合には、コマンドの結果で失敗の理由が表示されます。

4. マネージドクラスターの Klusterlet ステータスが degraded 状態でないかを確認します。Klusterlet のパフォーマンスが低下した理由を特定するには、degraded 状態にある Klusterlet のトラブルシューティング を参照してください。

1.8. PENDING IMPORT ステータスのクラスターのトラブルシューティング

クラスターのコンソールで継続的に **Pending import** と表示される場合は、以下の手順を実行して問題をトラブルシューティングしてください。

1.8.1. 現象: ステータスが Pending Import クラスター

Red Hat Advanced Cluster Management コンソールを使用してクラスターをインポートした後に、コンソールで、クラスターのステータスが **Pending import** と表示されます。

1.8.2. 問題の特定: ステータスが Pending Import クラスター

1. マネージドクラスターで以下のコマンドを実行し、問題のある Kubernetes Pod 名を表示します。

```
kubectl get pod -n open-cluster-management-agent | grep klusterlet-registration-agent
```

2. マネージドクラスターで以下のコマンドを実行し、エラーのログエントリーを探します。

```
kubectl logs <registration_agent_pod> -n open-cluster-management-agent
```

registration_agent_pod は、手順 1 で特定した Pod 名に置き換えます。

3. 返された結果に、ネットワーク接続の問題があったと示すテキストがないかどうかを検索します。たとえば、**no such host** です。

1.8.3. 問題の解決: ステータスが Pending Import クラスター

1. ハブクラスターで以下のコマンドを実行して、問題のあるポート番号を取得します。

```
oc get infrastructure cluster -o yaml | grep apiServerURL
```

2. マネージドクラスターのホスト名が解決でき、ホストおよびポートへの送信接続が機能していることを確認します。
   マネージドクラスターで通信が確立できない場合は、クラスターのインポートが完了していません。マネージドクラスターのクラスターステータスは、**Pending import** になります。

1.9. クラスターがすでに存在するというエラーのトラブルシューティング

OpenShift Container Platform クラスターを Red Hat Advanced Cluster Management **MultiClusterHub** にインポートできず、**AlreadyExists** エラーを受け取る場合は、手順に従って問題をトラブルシューティングします。

1.9.1. 以前の動作: OpenShift Container Platform クラスターをインポートするときにエラーログがすでに存在する

OpenShift Container Platform クラスターを Red Hat Advanced Cluster Management **MultiClusterHub** にインポートすると、エラーログが表示されます。

```
error log:
Warning: apiextensions.k8s.io/v1beta1 CustomResourceDefinition is deprecated in v1.16+,
unavailable in v1.22+; use apiextensions.k8s.io/v1 CustomResourceDefinition
Error from server (AlreadyExists): error when creating "STDIN":
customresourcedefinitions.apiextensions.k8s.io "klusterlets.operator.open-cluster-management.io"
already exists
The cluster cannot be imported because its Klusterlet CRD already exists.
Either the cluster was already imported, or it was not detached completely during a previous detach
process.
Detach the existing cluster before trying the import again."
```

1.9.2. 問題の特定: OpenShift Container Platform クラスターのインポート時にすでに存在する

以下のコマンドを実行して、新しい Red Hat Advanced Cluster Management **MultiClusterHub** にインポートする Red Hat Advanced Cluster Management 関連のリソースがクラスター上にあるかどうかを確認します。

```
oc get all -n open-cluster-management-agent
oc get all -n open-cluster-management-agent-addon
```

1.9.3. 問題の解決: OpenShift Container Platform クラスターのインポート時にすでに存在する

次のコマンドを使用して、**klusterlet** カスタムリソースを削除します。

```
oc get klusterlet | grep klusterlet | awk '{print $1}' | xargs oc patch klusterlet --type=merge -p
'{"metadata":{"finalizers": []}}'
```

次のコマンドを実行して、既存のリソースを削除します。

```
oc delete namespaces open-cluster-management-agent open-cluster-management-agent-addon --
wait=false
oc get crds | grep open-cluster-management.io | awk '{print $1}' | xargs oc delete crds --wait=false
```

```
oc get crds | grep open-cluster-management.io | awk '{print $1}' | xargs oc patch crds --type=merge -p '{"metadata":{"finalizers": []}}'
```

1.10. VMWARE VSPHERE でのクラスター作成のトラブルシューティング

VMware vSphere で Red Hat OpenShift Container Platform クラスターを作成する時に問題が発生した場合は、以下のトラブルシューティング情報を参照して、この情報のいずれかが問題に対応しているかどうかを確認します。

注記: VMware vSphere でクラスター作成プロセスが失敗した場合に、リンクが有効にならずログが表示されないことがあります。上記が発生する場合は、**hive-controllers** Pod のログを確認して問題を特定できます。**hive-controllers** ログは **hive** namespace にあります。

1.10.1. 証明書の IP SAN エラーでマネージドクラスターの作成に失敗する

1.10.1.1. 現象: 証明書の IP SAN エラーでマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、証明書 IP SAN エラーを示すエラーメッセージでクラスターに問題が発生します。

1.10.1.2. 問題の特定: 証明書の IP SAN エラーでマネージドクラスターの作成に失敗する

マネージドクラスターのデプロイメントに失敗して、デプロイメントログに以下のエラーが返されます。

```
time="2020-08-07T15:27:55Z" level=error msg="Error: error setting up new vSphere SOAP client: Post https://147.1.1.1/sdk: x509: cannot validate certificate for xx.xx.xx.xx because it doesn't contain any IP SANs"
time="2020-08-07T15:27:55Z" level=error
```

1.10.1.3. 問題の解決: 証明書の IP SAN エラーでマネージドクラスターの作成に失敗する

認証情報の IP アドレスではなく VMware vCenter サーバー完全修飾ホスト名を使用します。また、VMware vCenter CA 証明書を更新して、IP SAN を組み込むこともできます。

1.10.2. 不明な証明局のエラーでマネージドクラスターの作成に失敗する

1.10.2.1. 現象: 不明な証明局のエラーでマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、証明書が不明な証明局により署名されているのでクラスターに問題が発生します。

1.10.2.2. 問題の特定: 不明な証明局のエラーでマネージドクラスターの作成に失敗する

マネージドクラスターのデプロイメントに失敗して、デプロイメントログに以下のエラーが返されます。

```
Error: error setting up new vSphere SOAP client: Post https://vspherehost.com/sdk: x509: certificate signed by unknown authority"
```

1.10.2.3. 問題の解決: 不明な証明局のエラーでマネージドクラスターの作成に失敗する

認証情報の作成時に認証局の正しい証明書が入力されていることを確認します。

1.10.3. 証明書の期限切れでマネージドクラスターの作成に失敗する

1.10.3.1. 現象: 証明書の期限切れでマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、証明書の期限が切れているか、有効にしていないため、クラスターに問題が発生します。

1.10.3.2. 問題の特定: 証明書の期限切れでマネージドクラスターの作成に失敗する

マネージドクラスターのデプロイメントに失敗して、デプロイメントログに以下のエラーが返されます。

```
x509: certificate has expired or is not yet valid
```

1.10.3.3. 問題の解決: 証明書の期限切れでマネージドクラスターの作成に失敗する

ESXi ホストの時間が同期されていることを確認します。

1.10.4. タグ付けの権限が十分ではないためマネージドクラスターの作成に失敗する

1.10.4.1. 現象: タグ付けの権限が十分ではないためマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、タグ付けの使用に十分な権限がないためクラスターに問題が発生します。

1.10.4.2. 問題の特定: タグ付けの権限が十分にないためにマネージドクラスターの作成に失敗する

マネージドクラスターのデプロイメントに失敗して、デプロイメントログに以下のエラーが返されます。

```
time="2020-08-07T19:41:58Z" level=debug msg="vsphere_tag_category.category: Creating..."
time="2020-08-07T19:41:58Z" level=error
time="2020-08-07T19:41:58Z" level=error msg="Error: could not create category: POST
https://vspherehost.com/rest/com/vmware/cis/tagging/category: 403 Forbidden"
time="2020-08-07T19:41:58Z" level=error
time="2020-08-07T19:41:58Z" level=error msg="  on ../tmp/openshift-install-436877649/main.tf line
54, in resource \"vsphere_tag_category\" \"category\":"
time="2020-08-07T19:41:58Z" level=error msg="  54: resource \"vsphere_tag_category\" \"category\"
{"
```

1.10.4.3. 問題の解決: タグ付けの権限が十分ではないためマネージドクラスターの作成に失敗する

VMware vCenter が必要とするアカウントの権限が正しいことを確認します。詳細は、インストール時に削除されたイメージレジストリー を参照してください。

1.10.5. 無効な dnsVIP でマネージドクラスターの作成に失敗する

1.10.5.1. 現象: 無効な dnsVIP でマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、dnsVIP が無効であるため、クラスターに問題が発生します。

1.10.5.2. 問題の特定: 無効な dnsVIP でマネージドクラスターの作成に失敗する

VMware vSphere で新しいマネージドクラスターをデプロイしようとして以下のメッセージが表示されるのは、VMware Installer Provisioned Infrastructure (IPI) をサポートしない以前の OpenShift Container Platform リリースイメージを使用しているためです。

```
failed to fetch Master Machines: failed to load asset \\\"Install Config\\\": invalid \\\"install-config.yaml\\\" file: platform.vsphere.dnsVIP: Invalid value: \\\"\\\": \\\"\\\" is not a valid IP
```

1.10.5.3. 問題の解決: 無効な dnsVIP でマネージドクラスターの作成に失敗する

VMware インストーラーでプロビジョニングされるインフラストラクチャーをサポートする OpenShift Container Platform で、新しいバージョンのリリースイメージを選択します。

1.10.6. ネットワークタイプが正しくないためマネージドクラスターの作成に失敗する

1.10.6.1. 現象: ネットワークタイプが正しくないためマネージドクラスターの作成に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、間違ったネットワークタイプが指定されているため、クラスターに問題が発生します。

1.10.6.2. 問題の特定: ネットワークタイプが正しくないためマネージドクラスターの作成に失敗する

VMware vSphere で新しいマネージドクラスターをデプロイしようとして以下のメッセージが表示されるのは、VMware Installer Provisioned Infrastructure (IPI) をサポートしない以前の OpenShift Container Platform イメージを使用しているためです。

```
time="2020-08-11T14:31:38-04:00" level=debug msg="vsphereprivate_import_ova.import: Creating..."
time="2020-08-11T14:31:39-04:00" level=error
time="2020-08-11T14:31:39-04:00" level=error msg="Error: rpc error: code = Unavailable desc = transport is closing"
time="2020-08-11T14:31:39-04:00" level=error
time="2020-08-11T14:31:39-04:00" level=error
time="2020-08-11T14:31:39-04:00" level=fatal msg="failed to fetch Cluster: failed to generate asset \"Cluster\": failed to create cluster: failed to apply Terraform: failed to complete the change"
```

1.10.6.3. 問題の解決: ネットワークタイプが正しくないためマネージドクラスターの作成に失敗する

指定の VMware クラスターに対して有効な VMware vSphere ネットワークタイプを選択します。

1.10.7. ディスクの変更処理のエラーでマネージドクラスターの作成に失敗する

1.10.7.1. 現象: ディスクの変更処理のエラーが原因でマネージドクラスターの追加に失敗する

VMware vSphere で新規の Red Hat OpenShift Container Platform クラスターを作成した後に、ディスク変更処理時にエラーによりクラスターに問題が発生します。

1.10.7.2. 問題の特定: ディスクの変更処理のエラーが原因でマネージドクラスターの追加に失敗する

以下のようなメッセージがログに表示されます。

```
ERROR
ERROR Error: error reconfiguring virtual machine: error processing disk changes post-clone: disk.0:
ServerFaultCode: NoPermission: RESOURCE (vm-71:2000), ACTION (queryAssociatedProfile):
RESOURCE (vm-71), ACTION (PolicyIDByVirtualDisk)
```

1.10.7.3. 問題の解決: ディスクの変更処理のエラーが原因でマネージドクラスターの追加に失敗する

VMware vSphere クライアントを使用してユーザーに **プロファイル駆動型のストレージ権限** の **全権限** を割り当てます。

1.11. RED HAT OPENSTACK PLATFORM でマネージドクラスターの作成が不明な権限エラーで失敗する

Red Hat OpenStack Platform で Red Hat OpenShift Container Platform クラスターを作成する時に問題が発生した場合は、以下のトラブルシューティング情報を参照して、この情報のいずれかが問題に対応しているかどうかを確認します。

1.11.1. 現象: マネージドクラスターの作成が不明な権限エラーで失敗する

自己署名証明書を使用して Red Hat OpenStack Platform で新しい Red Hat OpenShift Container Platform クラスターを作成した後、クラスターは失敗し、不明な権限エラーを示すエラーメッセージが表示されます。

1.11.2. 問題の特定: マネージドクラスターの作成が不明な権限エラーで失敗する

マネージドクラスターのデプロイが失敗し、次のエラーメッセージが返されます。

x509: certificate signed by unknown authority

1.11.3. 問題の解決: マネージドクラスターの作成が不明な権限エラーで失敗する

次のファイルが正しく設定されていることを確認します。

1. **clouds.yaml** ファイルは、**cacert** パラメーターで **ca.crt** ファイルへのパスを指定する必要があります。ignition shim の生成時に、**cacert** パラメーターが OpenShift インストーラーに渡されます。以下の例を参照してください。

   ```
   clouds:
     openstack:
       cacert: "/etc/pki/ca-trust/source/anchors/ca.crt"
   ```

2. **certificatesSecretRef** パラメーターは、**ca.crt** ファイルと一致するファイル名を持つシークレットを参照する必要があります。以下の例を参照してください。

   -

```
spec:
  baseDomain: dev09.red-chesterfield.com
  clusterName: txue-osspoke
  platform:
    openstack:
      cloud: openstack
      credentialsSecretRef:
        name: txue-osspoke-openstack-creds
      certificatesSecretRef:
        name: txue-osspoke-openstack-certificatebundle
```

一致するファイル名でシークレットを作成するには、次のコマンドを実行します。

```
oc create secret generic txue-osspoke-openstack-certificatebundle --from-
file=ca.crt=ca.crt.pem -n $CLUSTERNAME
```

3. **ca.cert** ファイルのサイズは、63,000 バイト未満である必要があります。

1.12. OPENSHIFT CONTAINER PLATFORM バージョン 3.11 クラスターのインポートの失敗時のトラブルシューティング

1.12.1. 現象: OpenShift Container Platform バージョン 3.11 クラスターのインポートに失敗する

Red Hat OpenShift Container Platform バージョン 3.11 クラスターのインポートを試行すると、以下の内容のようなログメッセージでインポートに失敗します。

```
customresourcedefinition.apiextensions.k8s.io/klusterlets.operator.open-cluster-management.io
configured
clusterrole.rbac.authorization.k8s.io/klusterlet configured
clusterrole.rbac.authorization.k8s.io/open-cluster-management:klusterlet-admin-aggregate-clusterrole
configured
clusterrolebinding.rbac.authorization.k8s.io/klusterlet configured
namespace/open-cluster-management-agent configured
secret/open-cluster-management-image-pull-credentials unchanged
serviceaccount/klusterlet configured
deployment.apps/klusterlet unchanged
klusterlet.operator.open-cluster-management.io/klusterlet configured
Error from server (BadRequest): error when creating "STDIN": Secret in version "v1" cannot be
handled as a Secret:
v1.Secret.ObjectMeta:
v1.ObjectMeta.TypeMeta: Kind: Data: decode base64: illegal base64 data at input byte 1313, error
found in #10 byte of ...|dhruy45="},"kind":"|..., bigger context
...|tye56u56u568yuo7i67i67i67o556574i"},"kind":"Secret","metadata":{"annotations":{"kube|...
```

1.12.2. 問題の特定: OpenShift Container Platform バージョン 3.11 クラスターのインポートに失敗する

この問題は多くの場合、インストールされている **kubectl** コマンドラインツールのバージョンが 1.11 以前であるために発生します。以下のコマンドを実行して、実行中の **kubectl** コマンドラインツールのバージョンを表示します。

```
kubectl version
```

返されたデータがバージョンが 1.11 以前の場合は、**問題の解決: OpenShift Container Platform バージョン 3.11 クラスターのインポートに失敗する** に記載される修正のいずれかを実行します。

1.12.3. 問題の解決: OpenShift Container Platform バージョン 3.11 クラスターのインポートに失敗する

この問題は、以下のいずれかの手順を実行して解決できます。

- 最新バージョンの **kubectl** コマンドラインツールをインストールします。
  1. **kubectl** ツールの最新バージョンを、Kubernetes ドキュメントの kubectl のインストールとセットアップ からダウンロードします。
  2. **kubectl** ツールのアップグレード後にクラスターを再度インポートします。
- import コマンドが含まれるファイルを実行します。
  1. CLI を使用したマネージドクラスターのインポート の手順を開始します。
  2. クラスターの import コマンドを作成する場合には、この import コマンドを **import.yaml** という名前の YAML ファイルにコピーします。
  3. 以下のコマンドを実行して、ファイルからクラスターを再度インポートします。

     ```
     oc apply -f import.yaml
     ```

1.13. 証明書を変更した後のインポート済みクラスターのオフラインでのトラブルシューティング

カスタムの **apiserver** 証明書のインストールはサポートされますが、証明書情報を変更する前にインポートされたクラスターの 1 つまたは複数でステータスが **offline** になります。

1.13.1. 現象: 証明書の変更後にクラスターがオフラインになる

証明書シークレットを更新する手順を完了すると、オンラインだった 1 つ以上のクラスターがコンソールに **offline** ステータスを表示するようになります。

1.13.2. 問題の特定: 証明書の変更後にクラスターがオフラインになる

カスタムの API サーバー証明書の情報を更新すると、インポートされ、新しい証明書が追加される前に稼働していたクラスターのステータスが **offline** になります。

オフラインのマネージドクラスターの **open-cluster-management-agent** namespace にある Pod のログで、証明書に問題があるとのエラーが見つかります。以下の例のようなエラーがログに表示されます。

以下の **work-agent** ログを参照してください。

```
E0917 03:04:05.874759       1 manifestwork_controller.go:179] Reconcile work test-1-klusterlet-addon-workmgr fails with err: Failed to update work status with err Get "https://api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/namespaces/test-1/manifestworks/test-1-klusterlet-addon-workmgr": x509: certificate signed by unknown authority
```

```
E0917 03:04:05.874887       1 base_controller.go:231] "ManifestWorkAgent" controller failed to sync "test-1-klusterlet-addon-workmgr", err: Failed to update work status with err Get "api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/namespaces/test-1/manifestworks/test-1-klusterlet-addon-workmgr": x509: certificate signed by unknown authority
E0917 03:04:37.245859       1 reflector.go:127] k8s.io/client-go@v0.19.0/tools/cache/reflector.go:156: Failed to watch *v1.ManifestWork: failed to list *v1.ManifestWork: Get "api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/namespaces/test-1/manifestworks?resourceVersion=607424": x509: certificate signed by unknown authority
```

以下の **registration-agent** ログを確認してください。

```
I0917 02:27:41.525026       1 event.go:282] Event(v1.ObjectReference{Kind:"Namespace", Namespace:"open-cluster-management-agent", Name:"open-cluster-management-agent", UID:"", APIVersion:"v1", ResourceVersion:"", FieldPath:""}): type: 'Normal' reason: 'ManagedClusterAvailableConditionUpdated' update managed cluster "test-1" available condition to "True", due to "Managed cluster is available"
E0917 02:58:26.315984       1 reflector.go:127] k8s.io/client-go@v0.19.0/tools/cache/reflector.go:156: Failed to watch *v1beta1.CertificateSigningRequest: Get "https://api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/managedclusters?allowWatchBookmarks=true&fieldSelector=metadata.name%3Dtest-1&resourceVersion=607408&timeout=9m33s&timeoutSeconds=573&watch=true"": x509: certificate signed by unknown authority
E0917 02:58:26.598343       1 reflector.go:127] k8s.io/client-go@v0.19.0/tools/cache/reflector.go:156: Failed to watch *v1.ManagedCluster: Get "https://api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/managedclusters?allowWatchBookmarks=true&fieldSelector=metadata.name%3Dtest-1&resourceVersion=607408&timeout=9m33s&timeoutSeconds=573&watch=true": x509: certificate signed by unknown authority
E0917 02:58:27.613963       1 reflector.go:127] k8s.io/client-go@v0.19.0/tools/cache/reflector.go:156: Failed to watch *v1.ManagedCluster: failed to list *v1.ManagedCluster: Get "https://api.aaa-ocp.dev02.location.com:6443/apis/cluster.management.io/v1/managedclusters?allowWatchBookmarks=true&fieldSelector=metadata.name%3Dtest-1&resourceVersion=607408&timeout=9m33s&timeoutSeconds=573&watch=true"": x509: certificate signed by unknown authority
```

1.13.3. 問題の解決: 証明書の変更後にクラスターがオフラインになる

マネージドクラスターが **local-cluster** であるか、Red Hat Advanced Cluster Management for Kubernetes を使用して作成された場合、マネージドクラスターを再インポートするには、10 分以上待つ必要があります。

マネージドクラスターをすぐに再インポートするには、ハブクラスター上のマネージドクラスターのインポートシークレットを削除し、Red Hat Advanced Cluster Management を使用して再インポートします。以下のコマンドを実行します。

```
oc delete secret -n <cluster_name> <cluster_name>-import
```

<cluster_name> を、インポートするマネージドクラスターの名前に置き換えます。

Red Hat Advanced Cluster Management を使用してインポートされたマネージドクラスターを再インポートする場合は、以下の手順を実行してマネージドクラスターを再度インポートします。

1. ハブクラスターで、次のコマンドを実行してマネージドクラスターのインポートシークレットを再作成します。

```
oc delete secret -n <cluster_name> <cluster_name>-import
```

<cluster_name> を、インポートするマネージドクラスターの名前に置き換えます。

2. ハブクラスターで、次のコマンドを実行して、マネージドクラスターのインポートシークレットを YAML ファイルに公開します。

```
oc get secret -n <cluster_name> <cluster_name>-import -ojsonpath='{.data.import\.yaml}' |
base64 --decode  > import.yaml
```

<cluster_name> を、インポートするマネージドクラスターの名前に置き換えます。

3. マネージドクラスターで、次のコマンドを実行して **import.yaml** ファイルを適用します。

```
oc apply -f import.yaml
```

注記: 前の手順では、マネージドクラスターがハブクラスターから切り離されません。この手順により、必要なマニフェストがマネージドクラスターの現在の設定 (新しい証明書情報を含む) で更新されます。

1.14. クラスターの削除後も NAMESPACE が残る

マネージドクラスターを削除すると、通常 namespace はクラスターの削除プロセスの一部として削除されます。まれに namespace は一部のアーティファクトが含まれた状態で残る場合があります。このような場合は、namespace を手動で削除する必要があります。

1.14.1. 現象: クラスターの削除後も namespace が残る

マネージドクラスターの削除後に namespace が削除されません。

1.14.2. 問題の解決: クラスターの削除後も namespace が残る

namespace を手作業で削除するには、以下の手順を実行します。

1. 次のコマンドを実行して、<cluster_name> namespace に残っているリソースのリストを作成します。

```
oc api-resources --verbs=list --namespaced -o name | grep -E
'^secrets|^serviceaccounts|^managedclusteraddons|^roles|^rolebindings|^manifestworks|^lease
|^managedclusterinfo|^appliedmanifestworks'|^clusteroauths' | xargs -n 1 oc get --show-kind -
-ignore-not-found -n <cluster_name>
```

cluster_name は、削除を試みたクラスターの namespace 名に置き換えます。

2. 以下のコマンドを入力してリストを編集し、ステータスが **Delete** ではないリストから特定したリソースを削除します。

```
oc edit <resource_kind> <resource_name> -n <namespace>
```

resource_kind は、リソースの種類に置き換えます。**resource_name** は、リソース名に置き換えます。**namespace** は、リソースの namespace に置き換えます。

3. メタデータで **finalizer** 属性の場所を特定します。

4. vi エディターの **dd** コマンドを使用して、Kubernetes 以外のファイナライザーを削除します。
5. **:wq** コマンドを入力し、リストを保存して **vi** エディターを終了します。
6. 以下のコマンドを入力して namespace を削除します。

```
oc delete ns <cluster-name>
```

cluster-name を、削除する namespace の名前に置き換えます。

1.15. クラスターのインポート時の AUTO-IMPORT-SECRET-EXISTS エラー

クラスターのインポートは、auto import secret exists というエラーメッセージで失敗します。

1.15.1. 現象: クラスターのインポート時の Auto-import-secret-exists エラー

管理用のハイブクラスターをインポートすると、**auto-import-secret already exists** というエラーが表示されます。

1.15.2. 問題の解決: クラスターのインポート時の Auto-import-secret-exists エラー

この問題は、Red Hat Advanced Cluster Management で以前に管理されていたクラスターをインポートしようとすると発生します。これが生じると、クラスターを再インポートしようとすると、シークレットは競合します。

この問題を回避するには、以下の手順を実行します。

1. 既存の **auto-import-secret** を手動で削除するには、ハブクラスターで以下のコマンドを実行します。

```
oc delete secret auto-import-secret -n <cluster-namespace>
```

namespace は、お使いのクラスターの namespace に置き換えます。

2. クラスターインポートの概要 の手順を使用して、クラスターを再度インポートします。

1.16. VOLSYNC 用の CINDER CONTAINER STORAGE INTERFACE (CSI) ドライバーのトラブルシューティング

VolSync を使用する場合、または cinder Container Storage Interface (CSI) ドライバーのデフォルト設定を使用する場合、使用中の PVC でエラーが発生する可能性があります。

1.16.1. 症状: ボリュームスナップショット のエラー状態

スナップショットを使用するように VolSync **ReplicationSource** または **ReplicationDestination** を設定できます。また、**ReplicationSource** および **ReplicationDestination** で **storageclass** および **volumesnapshotclass** を設定できます。cinder **volumesnapshotclass** には、デフォルト値が **false** の、**force-create** というパラメーターがあります。**volumesnapshotclass** でこの **force-create** パラメーターを指定すると、cinder が使用中の PVC から **volumesnapshot** を取得できません。その結果、**volumesnapshot** はエラー状態になります。

1.16.2. 問題の解決: パラメーターを true に設定する

1. cinder CSI ドライバーの新しい **volumesnapshotclass** を作成します。
2. **force-create** パラメーターを **true** に変更します。以下のサンプル YAML を参照してください。

```
apiVersion: snapshot.storage.k8s.io/v1
deletionPolicy: Delete
driver: cinder.csi.openstack.org
kind: VolumeSnapshotClass
metadata:
  annotations:
    snapshot.storage.kubernetes.io/is-default-class: 'true'
  name: standard-csi
parameters:
  force-create: 'true'
```

1.17. MUST-GATHER コマンドを実行したトラブルシューティング

must-gather コマンドを実行して詳細、ログを収集し、問題のデバッグ手順を実行します。このデバッグ情報は、サポートリクエストを開くときにも役立ちます。**oc adm must-gather** CLI コマンドは、問題のデバッグによく必要となる次のような情報をクラスターから収集します。

- リソース定義
- サービスログ

1.17.1. 前提条件

must-gather コマンドを実行するには、次の前提条件を満たす必要があります。

- **cluster-admin** ロールを持つユーザーとして、グローバルハブおよびマネージドハブクラスターにアクセスします。
- OpenShift Container Platform CLI (oc) がインストールされている。

1.17.2. must-gather コマンドの実行

must-gather コマンドを使用して情報を収集するには、次の手順を実行します。

1. OpenShift Container Platform ドキュメントの クラスターに関するデータの収集 を読み、**must-gather** コマンドについて学び、必要な前提条件をインストールします。
2. グローバルハブクラスターにログインします。一般的な使用例では、グローバルハブクラスターにログインしているときに次のコマンドを実行します。

```
oc adm must-gather --image=quay.io/stolostron/must-gather:SNAPSHOTNAME
```

   マネージドハブクラスターを確認する場合は、それらのクラスターで **must-gather** コマンドを実行します。

3. オプション: 結果を **SOMENAME** ディレクトリーに保存する場合は、前の手順で 1 つではなく次のコマンドを実行できます。

```
oc adm must-gather --image=quay.io/stolostron/must-gather:SNAPSHOTNAME --dest-dir=
<SOMENAME> ; tar -cvzf <SOMENAME>.tgz <SOMENAME>
```

ディレクトリーには別の名前を指定できます。

注記: コマンドには、gzipped tarball ファイルを作成するために必要な追加が含まれます。

以下の情報は、**must-gather** コマンドから収集されます。

- 2 つのピアレベル: **cluster-scoped-resources** と **namespaces** リソース。
- それぞれに対するサブレベル: クラスタースコープおよび namespace スコープの両方のリソースに対するカスタムリソース定義の API グループ。
- それぞれの次のレベル: 種類別にソートされた YAML ファイル。
- グローバルハブクラスターの場合は、**namespaces** リソースの **PostgresCluster** と **Kafka** を確認できます。
- グローバルハブクラスターの場合は、Multicluster Global Hub 関連の Pod と、**namespaces** リソースの **Pod** 内のログを確認できます。
- マネージドハブクラスターの場合、Multicluster Global Hub エージェント Pod と **namespaces** リソースの **Pod** 内のログを確認できます。

1.18. プロビジョニングされた POSTGRESQL データベースにアクセスすることでのトラブルシューティング

プロビジョニングされた PostgreSQL データベースにアクセスして、multicluster global hub の問題のトラブルシューティングに役立つメッセージを表示できます。サービスの種類に応じて、プロビジョニングされた PostgreSQL データベースにアクセスするには 3 つの方法があります。

- **ClusterIP** サービスの使用
  1. 次のコマンドを実行して、postgres 接続 URI を確認します。

     ```
     oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-
     template='{{index (.data) "uri" | base64decode}}'
     ```

  2. 次のコマンドを実行してデータベースにアクセスします。

     ```
     oc exec -it $(kubectl get pods -n multicluster-global-hub-postgres -l postgres-
     operator.crunchydata.com/role=master -o jsonpath='{.items..metadata.name}') -c
     database -n multicluster-global-hub-postgres -- psql -U postgres -d hoh -c "SELECT 1"
     ```

- **NodePort** サービスの使用
  1. 次のコマンドを実行して、サービスを NodePort に変更し、ホストをノード IP に設定し、ポートを 32432 に設定します。

     ```
     oc patch postgrescluster hoh -n multicluster-global-hub-postgres -p '{"spec":{"service":
     {"type":"NodePort", "nodePort": 32432}}}'  --type merge
     ```

  2. 次のコマンドを実行してユーザー名を追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "user" | base64decode}}'
```

3. 次のコマンドを実行してパスワードを追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "password" | base64decode}}'
```

4. 次のコマンドを実行して、データベース名を追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "dbname" | base64decode}}'
```

- **LoadBalancer**

1. 次のコマンドを実行して、サービスタイプを **LoadBalancer** に設定します。

```
oc patch postgrescluster hoh -n multicluster-global-hub-postgres -p '{"spec":{"service":{"type":"LoadBalancer"}}}'  --type merge
```

デフォルトのポートは 5432 です

2. 次のコマンドを実行してホスト名を設定します。

```
kubectl get svc -n multicluster-global-hub-postgres hoh-ha -ojsonpath='{.status.loadBalancer.ingress[0].hostname}'
```

3. 次のコマンドを実行してユーザー名を追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "user" | base64decode}}'
```

4. 次のコマンドを実行してパスワードを追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "password" | base64decode}}'
```

5. 次のコマンドを実行して、データベース名を追加します。

```
oc get secrets -n multicluster-global-hub-postgres hoh-pguser-postgres -o go-template='{{index (.data) "dbname" | base64decode}}'
```

1.19. トラブルシューティングのためのデータベースのダンプと復元の使用

運用環境では、データベース管理タスクとして PostgreSQL データベースを定期的にバックアップします。バックアップは、Multicluster Global Hub のデバッグにも使用できます。

1.19.1. ダバッグ用にデータベースの出力をダンプする

問題をデバッグするために、Multicluster Global Hub データベースに出力をダンプする必要がある場合があります。PostgreSQL データベースには、データベースの内容をダンプするための **pg_dump** コマンドラインツールが用意されています。localhost データベースサーバーからデータをダンプするには、

次のコマンドを実行します。

```
pg_dump hoh > hoh.sql
```

リモートサーバー上にある Multicluster Global Hub データベースを圧縮形式でダンプするには、次の例に示すように、コマンドラインオプションを使用して接続の詳細を制御します。

```
pg_dump -h my.host.com -p 5432 -U postgres -F t hoh -f hoh-$(date +%d-%m-%y_%H-%M).tar
```

1.19.2. ダンプからデータベースを復元する

PostgreSQL データベースを復元するには、**psql** または **pg_restore** コマンドラインツールを使用できます。**psql** ツールは、**pg_dump** によって作成されたプレーンテキストファイルを復元するために使用されます。

```
psql -h another.host.com -p 5432 -U postgres -d hoh < hoh.sql
```

pg_restore ツールは、**pg_dump** によって非プレーンテキスト形式 (カスタム、tar、またはディレクトリー) のいずれかで作成されたアーカイブから PostgreSQL データベースを復元するために使用されます。

```
pg_restore -h another.host.com -p 5432 -U postgres -d hoh hoh-$(date +%d-%m-%y_%H-%M).tar
```

1.20. コンプライアンスデータの復元

Grafana データソースは主に、**history.local_compliance** という名前のテーブルから取得されます。そのレコードは、毎晩 00:00:00 に開始される要約ルーチンによって生成されます。通常、要約プロセスを手動で実行する必要はありません。場合によっては、コンプライアンスジョブの実行時に予期しないエラーが発生する可能性があるため、データベースに手動でログインして概要プロセス全体を実行し、生成されなかったデータを回復する必要があります。手動での要約プロセスの実行 の手順に従ってデータを回復できます。

1.20.1. オプション: 既存のテーブルをパーティションテーブルに手動でアップグレードする

GA より前に Multicluster Global Hub の初期バージョンをインストールしている場合は、現在の Multicluster Global Hub Operator と互換性があるようにテーブルをアップグレードする必要があります。アップグレードの主な目的は、**event.local_policies**、**event.local_root_policies**、および **history.local_compliance** テーブルをパーティション化されたテーブルに変換することです。

次の例は、日付が **2023-08** に設定された **event.local_policies** テーブルの変換を示しています。他の 2 つのテーブルのアップグレード手順は同様です。

1. ターゲットがパーティション化されていることを確認してください。

   ```
   SELECT relname, relkind FROM pg_class WHERE relname = 'local_policies';
   ```

   テーブルの出力は次の例のようになります。

relname	relkind	local_policies	r

relkind が **p** の場合、現在のテーブルはパーティション化されます。存在する場合は、残りの手順をスキップして、他のテーブルをアップグレードできます。

2. 通常のテーブルをパーティションテーブルに変換します。

```
-- start a transaction
BEGIN;
-- Rename the legacy TABLE_NAME
ALTER TABLE event.local_policies RENAME TO local_policies_old;
-- Partition tables: https://github.com/stolostron/multicluster-global-
hub/blob/main/operator/pkg/controllers/hubofhubs/database/2.tables.sql#L283-L318
CREATE TABLE IF NOT EXISTS event.local_policies (
   event_name character varying(63) NOT NULL,
   policy_id uuid NOT NULL,
   cluster_id uuid NOT NULL,
   leaf_hub_name character varying(63) NOT NULL,
   message text,
   reason text,
   count integer NOT NULL DEFAULT 0,
   source jsonb,
   created_at timestamp without time zone DEFAULT now() NOT NULL,
   compliance local_status.compliance_type NOT NULL,
   -- Rename the constraint to avoid conflicts
   CONSTRAINT local_policies_unique_partition_constraint UNIQUE (event_name, count,
created_at)
) PARTITION BY RANGE (created_at);
-- Create partitions, load the old data to the previous partition table
CREATE TABLE IF NOT EXISTS event.local_policies_2023_08 PARTITION OF
event.local_policies FOR VALUES FROM ('2023-08-01') TO ('2023-09-01');
CREATE TABLE IF NOT EXISTS event.local_policies_2023_07 PARTITION OF
event.local_policies FOR VALUES FROM ('2000-01-01') TO ('2023-08-01');

-- Move the records from regular table to partition table
INSERT INTO event.local_policies SELECT * FROM event.local_polcies_old;
DROP TABLE IF EXISTS event.local_policies_old;
-- commit the transaction
COMMIT;
```

テーブル名と現在の日付に従って、次の値を置き換えることができます。

- **event.local_policies_2023_08** は、**現在の月** の接尾辞が付いたパーティション名です。例として 8 月を使用します。
- **'2023-08-01'** と **'2023-09-01'** は、現在の月のパーティションの最小境界と最大境界です。
- **event.local_policies_2023_07** は、**前月** (7 月) の接尾辞が付いたパーティション名です。
- **'2000-01-01'** と **'2023-08-01'** は、前月のパーティションの最小境界と最大境界です。

1.21. クラスターのステータスが OFFLINE から AVAILABLE に変わる場合のトラブルシューティング

マネージドクラスターのステータスは、環境またはクラスターを手動で変更することなく、**offline** と **available** との間で切り替わります。

1.21.1. 現象: クラスターのステータスが offline から available に変わる

マネージドクラスターからハブクラスターへのネットワーク接続が不安定な場合に、マネージドクラスターのステータスが **offline** と **available** との間で順に切り替わると、ハブクラスターにより報告されます。

ハブクラスターとマネージドクラスター間の接続は、**leaseDurationSeconds** の間隔値で検証されるリースを通じて維持されます。連続 5 回の **leaseDurationSeconds** 値の試行以内にリースが検証されない場合、クラスターは **offline** としてマークされます。

たとえば、クラスターは **60 seconds** の **leaseDurationSeconds** の間隔で 5 分後に **offline** としてマークされます。この設定は、接続の問題や遅延などの理由により不適切であり、不安定になる可能性があります。

1.21.2. 問題の解決: クラスターのステータスが offline から available に変わる

5 回の検証試行はデフォルトであり変更できませんが、**leaseDurationSeconds** の間隔は変更できます。

クラスターを **offline** としてマークする時間を分単位で決定し、その値を 60 倍にして秒に変換します。次に、デフォルトの試行回数 5 で割ります。その結果が **leaseDurationSeconds** 値になります。

1. 次のコマンドを入力してハブクラスターの **ManagedCluster** 仕様を編集します。ただし、**cluster-name** はマネージドクラスターの名前に置き換えます。

```
oc edit managedcluster <cluster-name>
```

2. 以下のサンプル YAML にあるように、**ManagedCluster** 仕様の **leaseDurationSeconds** の値を増やします。

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  name: <cluster-name>
spec:
  hubAcceptsClient: true
  leaseDurationSeconds: 60
```

3. ファイルを保存し、適用します。

1.22. ステータスが PENDING または FAILED のクラスターのコンソールでのトラブルシューティング

作成してたクラスターのステータスがコンソールで Pending または Failed と表示されている場合は、以下の手順を実行して問題のトラブルシューティングを実行します。

1.22.1. 現象: コンソールでステータスが Pending または Failed のクラスターのトラブルシューティング

Red Hat Advanced Cluster Management for Kubernetes コンソールで新規クラスターを作成した後に、コンソールでクラスターのステータスが Pending または Failed と表示されてそれ以上進みません。

1.22.2. 問題の特定: コンソールでステータスが Pending または Failed のクラスター

クラスターのステータスが **Failed** と表示される場合は、クラスターの詳細ページに移動して、提供されたログへのリンクに進みます。ログが見つからない場合や、クラスターのステータスが **Pending** と表示される場合は、以下の手順を実行してログを確認します。

- 手順 1

  1. ハブクラスターで以下のコマンドを実行し、新規クラスターの namespace に作成した Kubernetes Pod の名前を表示します。

     ```
     oc get pod -n <new_cluster_name>
     ```

     new_cluster_name は、作成したクラスター名に置き換えます。

  2. 名前に **provision** の文字列が含まれる Pod が表示されていない場合は、手順 2 に進みます。タイトルに **provision** が含まれる Pod があった場合は、ハブクラスターで以下のコマンドを実行して、その Pod のログを表示します。

     ```
     oc logs <new_cluster_name_provision_pod_name> -n <new_cluster_name> -c hive
     ```

     new_cluster_name_provision_pod_name は、作成したクラスター名の後に **provision** が含まれる Pod 名を指定するように置き換えます。

  3. ログでエラーを検索してください。この問題の原因が解明する場合があります。

- 手順 2
  名前に **provision** が含まれる Pod がない場合は、問題がプロセスの初期段階で発生しています。ログを表示するには、以下の手順を実行します。

  1. ハブクラスターで以下のコマンドを実行してください。

     ```
     oc describe clusterdeployments -n <new_cluster_name>
     ```

     new_cluster_name は、作成したクラスター名に置き換えます。クラスターのインストールログの詳細は、Red Hat OpenShift ドキュメントの インストールログの収集 を参照してください。

  2. リソースの **Status.Conditions.Message** と **Status.Conditions.Reason** のエントリーに問題に関する追加の情報があるかどうかを確認します。

1.22.3. 問題の解決: コンソールでステータスが Pending または Failed のクラスター

ログでエラーを特定した後に、エラーの解決方法を決定してから、クラスターを破棄して、作り直してください。

以下の例では、サポート対象外のゾーンを選択している可能性を示すログエラーと、解決に必要なアクションが提示されています。

```
No subnets provided for zones
```

クラスターの作成時に、サポートされていないリージョンにあるゾーンを 1 つ以上選択しています。問題解決用にクラスターを再作成する時に、以下のアクションの 1 つを実行します。

- リージョン内の異なるゾーンを選択します。
- 他のゾーンをリストしている場合は、サポートを提供しないゾーンを省略します。

- お使いのクラスターに、別のリージョンを選択します。

ログから問題を特定した後に、クラスターを破棄し、再作成します。

クラスター作成の詳細は、クラスター作成の概要 を参照してください。

1.23. GRAFANA のトラブルシューティング

Grafana エクスプローラーで時間のかかるメトリックをクエリーすると、**Gateway Time-out** エラーが発生する可能性があります。

1.23.1. 現象: Grafana エクスプローラーゲートウェイのタイムアウト

Grafana エクスプローラーで時間のかかるメトリックをクエリーするときに **Gateway Time-out** エラーが発生した場合は、**open-cluster-management-observability** namespace の Grafana が原因でタイムアウトが発生した可能性があります。

1.23.2. 問題の解決: Grafana を設定する

この問題が発生した場合は、以下の手順を実行します。

1. Grafana のデフォルト設定に想定のタイムアウト設定があることを確認します。
   a. Grafana のデフォルトタイムアウト設定を確認するには、以下のコマンドを実行します。

   ```
   oc get secret grafana-config -n open-cluster-management-observability -o jsonpath="
   {.data.grafana\.ini}" | base64 -d | grep dataproxy -A 4
   ```

   以下のタイムアウト設定が表示されるはずです。

   ```
   [dataproxy]
   timeout = 300
   dial_timeout = 30
   keep_alive_seconds = 300
   ```

   b. Grafana のデフォルトのデータソースクエリータイムアウトを確認するには、以下のコマンドを実行します。

   ```
   oc get secret/grafana-datasources -n open-cluster-management-observability -o
   jsonpath="{.data.datasources\.yaml}" | base64 -d | grep queryTimeout
   ```

   以下のタイムアウト設定が表示されるはずです。

   ```
   queryTimeout: 300s
   ```

2. Grafana のデフォルト設定に予想されるタイムアウト設定があることを確認した場合は、次のコマンドを実行して、**open-cluster-management-observability** namespace で Grafana を設定できます。

   ```
   oc annotate route grafana -n open-cluster-management-observability --overwrite
   haproxy.router.openshift.io/timeout=300s
   ```

Grafana ページを更新し、再度メトリクスのクエリーを試行します。**Gateway Time-out** エラーが表示されなくなりました。

1.24. 配置ルールでローカルクラスターが選択されていない場合のトラブルシューティング

マネージドクラスターは配置ルールで選択されますが、**local-cluster** (同じく管理されているハブクラスター) は選択されません。配置ルールユーザーには、**local-cluster** namespace の **managedcluster** リソースを取得するためのパーミッションは付与されません。

1.24.1. 現象: ローカルクラスターがマネージドクラスターとして選択されていない問題のトラブルシューティング

すべてのマネージドクラスターは配置ルールで選択されますが、**local-cluster** は選択されません。配置ルールユーザーには、**local-cluster** namespace の **managedcluster** リソースを取得するためのパーミッションは付与されません。

1.24.2. 問題の解決: マネージドクラスターとして選択されていないローカルクラスターのトラブルシューティング

非推奨: PlacementRule

この問題を解決するには、**local-cluster** namespace に **managedcluster** 管理パーミッションを付与する必要があります。以下の手順を実行します。

1. マネージドクラスターのリストに **local-cluster** が含まれていること、配置ルールの **decisions** リストで **local-cluster** が表示されていないことを確認します。以下のコマンドを実行して結果を表示します。

   ```
   % oc get managedclusters
   ```

   出力例を確認すると、**local-cluster** が結合されているものの、**PlacementRule** の YAML にないことが分かります。

   ```
   NAME            HUB ACCEPTED   MANAGED CLUSTER URLS   JOINED   AVAILABLE   AGE
   local-cluster   true                                  True     True        56d
   cluster1        true                                  True     True        16h
   ```

   ```
   apiVersion: apps.open-cluster-management.io/v1
   kind: PlacementRule
   metadata:
     name: all-ready-clusters
     namespace: default
   spec:
     clusterSelector: {}
   status:
     decisions:
     - clusterName: cluster1
       clusterNamespace: cluster1
   ```

2. YAML ファイルに **Role** を作成し、**local-cluster** namespace に **managedcluster** 管理パーミッションを付与します。以下の例を参照してください。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: managedcluster-admin-user-zisis
  namespace: local-cluster
rules:
- apiGroups:
  - cluster.open-cluster-management.io
  resources:
  - managedclusters
  verbs:
  - get
```

3. **RoleBinding** リソースを作成し、配置ルールユーザーに **local-cluster** namespace へのアクセスを許可します。以下の例を参照してください。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: managedcluster-admin-user-zisis
  namespace: local-cluster
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: managedcluster-admin-user-zisis
  namespace: local-cluster
subjects:
- kind: User
  name: zisis
  apiGroup: rbac.authorization.k8s.io
```

1.25. アプリケーションの KUBERNETES デプロイメントバージョンのトラブルシューティング

非推奨の Kubernetes **apiVersion** を使用するマネージドクラスターはサポートされない可能性があります。非推奨の API バージョンの詳細は、Kubernetes issue を参照してください。

1.25.1. 現象: アプリケーションデプロイメントバージョン

サブスクリプションの YAML ファイルのアプリケーションリソースの 1 つまたは複数で、非推奨の API が使用されている場合に、以下のようなエラーが発生する可能性があります。

```
failed to install release: unable to build kubernetes objects from release manifest: unable to recognize
"": no matches for
kind "Deployment" in version "extensions/v1beta1"
```

または、**old.yaml** などの名前で YAML ファイルに新しい Kubernetes API バージョンを追加している場合は、以下のエラーが発生する可能性があります。

```
error: unable to recognize "old.yaml": no matches for kind "Deployment" in version
"deployment/v1beta1"
```

1.25.2. 解決: アプリケーションデプロイメントバージョン

1. リソースの **apiVersion** を更新します。たとえば、サブスクリプション YAML ファイルの **Deployment** の種類のエラーが表示された場合は、**apiVersion** を **extensions/v1beta1** から **apps/v1** に更新する必要があります。
以下の例を参照してください。

```
apiVersion: apps/v1
kind: Deployment
```

2. マネージドクラスターで以下のコマンドを実行して、利用可能なバージョンを確認します。

```
kubectl explain <resource>
```

3. **VERSION** を確認します。

1.26. DEGRADED 状態にある KLUSTERLET のトラブルシューティング

Klusterlet の状態が Degraded の場合は、マネージドクラスターの Klusterlet エージェントの状態を診断しやすくなります。Klusterlet の状態が Degraded になると、マネージドクラスターの Klusterlet エージェントで発生する可能性のあるエラーに対応する必要があります。Klusterlet の degraded の状態が **True** に設定されている場合は、以下の情報を参照します。

1.26.1. 現象: Klusterlet の状態が degraded である

マネージドクラスターで Klusterlet をデプロイした後に、**KlusterletRegistrationDegraded** または **KlusterletWorkDegraded** の状態が **True** と表示されます。

1.26.2. 問題の特定: Klusterlet の状態が degraded である

1. マネージドクラスターで以下のコマンドを実行して、Klusterlet のステータスを表示します

```
kubectl get klusterlets klusterlet -oyaml
```

2. **KlusterletRegistrationDegraded** または **KlusterletWorkDegraded** をチェックして、状態が **True** に設定されいるかどうかを確認します。記載されている Degraded の状態は、**問題の解決** に進みます。

1.26.3. 問題の解決: Klusterlet の状態が degraded である

ステータスが Degraded のリストおよびこれらの問題の解決方法を参照してください。

- **KlusterletRegistrationDegraded** の状態が **True** で、この状態の理由が **BootStrapSecretMissing** の場合は、**open-cluster-management-agent** namespace にブートストラップのシークレットを作成する必要があります。

- **KlusterletRegistrationDegraded** の状態が **True** と表示され、状態の理由が **BootstrapSecretError** または **BootstrapSecretUnauthorized** の場合は、現在のブートストラップシークレットが無効です。現在のブートストラップシークレットを削除して、**open-cluster-management-agent** namespace で有効なブートストラップシークレットをもう一度作成します。

- **KlusterletRegistrationDegraded** および **KlusterletWorkDegraded** が **True** と表示され、状態の理由が **HubKubeConfigSecretMissing** の場合は、Klusterlet を削除して作成し直します。
- **KlusterletRegistrationDegraded** および **KlusterletWorkDegraded** が **True** と表示され、状態の理由が **ClusterNameMissing**、**KubeConfigMissing**、**HubConfigSecretError**、または **HubConfigSecretUnauthorized** の場合は、**open-cluster-management-agent** namespace からハブクラスターの kubeconfig シークレットを削除します。登録エージェントは再度ブートストラップして、新しいハブクラスターの kubeconfig シークレットを取得します。
- **KlusterletRegistrationDegraded** が **True** と表示され、状態の理由が **GetRegistrationDeploymentFailed** または **UnavailableRegistrationPod** の場合は、状態のメッセージを確認して、問題の詳細を取得して解決してみてください。
- **KlusterletWorkDegraded** が **True** と表示され、状態の理由が **GetWorkDeploymentFailed** または **UnavailableWorkPod** の場合は、状態のメッセージを確認して、問題の詳細を取得し、解決してみてください。

1.27. オブジェクトストレージチャネルシークレットのトラブルシューティング

SecretAccessKey を変更すると、オブジェクトストレージチャネルのサブスクリプションは、更新されたシークレットを自動的に取得できず、エラーが発生します。

1.27.1. 現象: オブジェクトストレージチャネルシークレット

オブジェクトストレージチャネルのサブスクリプションは、更新されたシークレットを自動的に取得できません。そのため、サブスクリプションオペレーターが調整できなくなり、オブジェクトストレージからマネージドクラスターにリソースがデプロイされなくなります。

1.27.2. 問題の解決: オブジェクトストレージチャネルシークレット

シークレットを作成するには、認証情報を手動で入力してから、チャネル内のシークレットを参照する必要があります。

1. サブスクリプションオペレーターに単一の調整を生成するために、サブスクリプション CR にアノテーションを付けます。以下の **data** 仕様を参照してください。

```
apiVersion: apps.open-cluster-management.io/v1
kind: Channel
metadata:
  name: deva
  namespace: ch-obj
  labels:
    name: obj-sub
spec:
  type: ObjectBucket
  pathname: http://ec2-100-26-232-156.compute-1.amazonaws.com:9000/deva
  sourceNamespaces:
    - default
  secretRef:
    name: dev
---
apiVersion: v1
kind: Secret
metadata:
```

```
  name: dev
  namespace: ch-obj
  labels:
    name: obj-sub
data:
  AccessKeyID: YWRtaW4=
  SecretAccessKey: cGFzc3dvcmRhZG1pbg==
```

2. **oc annotate** を実行してテストします。

```
oc annotate appsub -n <subscription-namespace> <subscription-name> test=true
```

コマンドの実行後に、アプリケーションコンソールに移動して、リソースがマネージドクラスターにデプロイされていることを確認してください。または、マネージドクラスターにログインして、アプリケーションリソースが特定の namespace で作成されているかどうかを確認できます。

1.28. 可観測性のトラブルシューティング

可観測性コンポーネントをインストールした後に、コンポーネントが停止し、**Installing** のステータスが表示されます。

1.28.1. 現象: MultiClusterObservability リソースの状態が停止する

可観測性のカスタムリソース定義 (CRD) をインストールして作成した後に可観測性のステータスが **Installing** で停止すると、**spec:storageConfig:storageClass** パラメーターに値が定義されていない場合があります。または、可観測性コンポーネントは自動的にデフォルトの **storageClass** を検索しますが、ストレージの値が指定されていないと、コンポーネントが **Installing** ステータスのままになります。

1.28.2. 問題の解決: MultiClusterObservability リソースの状態が停止する

この問題が発生した場合は、以下の手順を実行します。

1. 可観測性コンポーネントがインストールされていることを確認します。
   a. **multicluster-observability-operator** を確認するには、以下のコマンドを実行します。

```
kubectl get pods -n open-cluster-management|grep observability
```

   b. 適切な CRD が存在することを確認するには、以下のコマンドを実行します。

```
kubectl get crd|grep observ
```

   以下の CRD が表示されるまで、コンポーネントを有効化しないでください。

```
multiclusterobservabilities.observability.open-cluster-management.io
observabilityaddons.observability.open-cluster-management.io
observatoria.core.observatorium.io
```

2. ベアメタルクラスター用に独自の storageClass を作成する場合は、NFS を使用した永続ストレージ を参照してください。

3. 可観測性コンポーネントがデフォルトの storageClass を検索できるようにするには、**multicluster-observability-operator** カスタムリソース定義の **storageClass** パラメーターを更新します。パラメーターは、以下のような値になります。

```
storageclass.kubernetes.io/is-default-class: "true"
```

インストールが完了すると、可観測性コンポーネントのステータスが **Ready** に更新されます。インストールの完了に失敗すると、ステータスが **Fail** と表示されます。

1.29. OPENSHIFT モニタリングサービスのトラブルシューティング

マネージドクラスターの可観測性サービスは OpenShift Container Platform モニタリングスタックからメトリクスを収集する必要があります。**metrics-collector** は、OpenShift Container Platform モニタリングスタックが準備状態にならないと、インストールされません。

1.29.1. 現象: OpenShift モニタリングサービスが準備状態にならない

endpoint-observability-operator-x Pod は、**prometheus-k8s** サービスが **openshift-monitoring** namespace で利用可能かどうかを確認します。このサービスが **openshift-monitoring** namespace に存在しない場合は、**metrics-collector** はデプロイされません。**Failed to get prometheus resource** というエラーメッセージが表示される可能性があります。

1.29.2. 問題の解決: OpenShift モニタリングサービスが準備状態にならない

この問題が発生した場合は、以下の手順を実行します。

1. OpenShift Container Platform クラスターにログイン。
2. **openshift-monitoring** namespace にアクセスし、**prometheus-k8s** サービスが利用可能であることを確認。
3. マネージドクラスターの **open-cluster-management-addon-observability** namespace で、**endpoint-observability-operator-x** Pod を再起動。

1.30. METRICS-COLLECTOR のトラブルシューティング

マネージドクラスターで **observability-client-ca-certificate** シークレットが更新されないと、内部サーバーのエラーが発生する可能性があります。

1.30.1. 現象: metrics-collector が observability-client-ca-certificate を検証できない

メトリックを利用できないマネージドクラスターが存在する可能性があります。この場合は、**metrics-collector** デプロイメントから以下のエラーが発生することがあります。

```
error: response status code is 500 Internal Server Error, response body is x509: certificate signed by unknown authority (possibly because of "crypto/rsa: verification error" while trying to verify candidate authority certificate "observability-client-ca-certificate")
```

1.30.2. 問題の解決: metrics-collector が observability-client-ca-certificate を検証できない

この問題が発生した場合は、以下の手順を実行します。

1. ターゲットのマネージドクラスターにログインします。

2. **open-cluster-management-addon-observability** namespace にある **observability-controller-open-cluster-management.io-observability-signer-client-cert** というシークレットを削除します。以下のコマンドを実行します。

```
oc delete secret observability-controller-open-cluster-management.io-observability-signer-client-cert -n open-cluster-management-addon-observability
```

注記: **observability-controller-open-cluster-management.io-observability-signer-client-cert** は、新しい証明書で自動的に再作成されます。

metrics-collector-deployment デプロイメントが再度作成され、**observability-controller-open-cluster-management.io-observability-signer-client-cert** シークレットが更新されます。

1.31. POSTGRESQL 共有メモリーエラーのトラブルシューティング

大規模な環境の場合、検索結果とアプリケーションのトポロジービューに影響を与える PostgreSQL 共有メモリーエラーが発生する可能性があります。

1.31.1. 現象: PostgreSQL 共有メモリーエラー

search-api ログに **ERROR: could not resize shared memory segment "/PostgreSQL.1083654800" to 25031264 bytes: No space left on device (SQLSTATE 53100)** のようなエラーメッセージが表示されます。

1.31.2. 問題の解決: PostgreSQL 共有メモリーエラー

この問題を解決するには、**search-postgres** ConfigMap にある PostgreSQL リソースを更新します。リソースを更新するには、次の手順を実行します。

1. 次のコマンドを実行して、**open-cluster-management** プロジェクトに切り替えます。

```
oc project open-cluster-management
```

2. **search-postgres** Pod のメモリーを増やします。次のコマンドは、メモリーを **16Gi** に増やします。

```
oc patch search -n open-cluster-management search-v2-operator --type json -p '[{"op": "add", "path": "/spec/deployments/database/resources", "value": {"limits": {"memory": "16Gi"}, "requests": {"memory": "32Mi", "cpu": "25m"}}}]'
```

3. 次のコマンドを実行して、検索 Operator が変更を上書きしないようにします。

```
oc annotate search search-v2-operator search-pause=true
```

4. 次のコマンドを実行して、**search-postgres** YAML ファイル内のリソースを更新します。

```
oc edit cm search-postgres -n open-cluster-management
```

リソースを増やすには、次の例を参照してください。

```
  postgresql.conf: |-
```

```
work_mem = '128MB' # Higher values allocate more memory
max_parallel_workers_per_gather = '0' # Disables parallel queries
shared_buffers = '1GB' # Higher values allocate more memory
```

終了する前に、必ず変更を保存してください。

5. 次のコマンドを実行して、**postgres** と **api** Pod を再起動します。

```
oc delete pod search-postgres-xyz search-api-xzy
```

6. 変更を確認するには、**search-postgres** YAML ファイルを開き、次のコマンドを実行して **postgresql.conf:** に加えた変更が存在することを確認します。

```
oc get cm search-postgres -n open-cluster-management -o yaml
```

環境変数の追加について、詳しくは 検索のカスタマイズと設定 を参照してください。

1.32. THANOS コンパクターのブロックエラーのトラブルシューティング

Thanos コンパクターのブロックが破損していることを示すブロックエラーメッセージが表示される場合があります。

1.32.1. 現象: Thanos コンパクターのブロックエラー

Red Hat Advanced Cluster Management for Kubernetes をアップグレードし、**oc logs observability-thanos-compact-0** コマンドを使用して Thanos コンパクターのログを確認すると、ログに次のエラーメッセージが表示されます。

```
ts=2024-01-24T15:34:51.948653839Z caller=compact.go:491 level=error msg="critical error detected; halting" err="compaction: group 0@15699422364132557315: compact blocks [/var/thanos/compact/compact/0@15699422364132557315/01HKZGQGJCKQWF3XMA8EXAMPLE /var/thanos/compact/compact/0@15699422364132557315/01HKZQK7TD06J2XWGR5EXAMPLE /var/thanos/compact/compact/0@15699422364132557315/01HKZYEZ2DVDQXF1STVEXAMPLE /var/thanos/compact/compact/0@15699422364132557315/01HM05APAHXBQSNC0N5EXAMPLE]: populate block: chunk iter: cannot populate chunk 8 from block 01HKZYEZ2DVDQXF1STVEXAMPLE: segment index 0 out of range"
```

1.32.2. 問題の解決: thanos bucket verify コマンドを追加する

オブジェクトストレージ設定に **thanos bucket verify** コマンドを追加します。以下の手順を実行します。

1. オブジェクトストレージ設定に **thanos bucket verify** コマンドを追加して、ブロックエラーを解決します。次のコマンドを使用して、**observability-thanos-compact** Pod を設定します。

```
oc rsh observability-thanos-compact-0
[..]
thanos tools bucket verify -r --objstore.config="$OBJSTORE_CONFIG" --objstore-backup.config="$OBJSTORE_CONFIG" --id=01HKZYEZ2DVDQXF1STVEXAMPLE
```

2. 前のコマンドが機能しない場合は、ブロックが破損している可能性があるため、削除対象としてマークする必要があります。以下のコマンドを実行します。

```
thanos tools bucket mark --id "01HKZYEZ2DVDQXF1STVEXAMPLE" --
objstore.config="$OBJSTORE_CONFIG" --marker=deletion-mark.json --details=DELETE
```

3. 削除をブロックした場合は、次のコマンドを実行してマークされたブロックをクリーンアップします。

```
thanos tools bucket cleanup --objstore.config="$OBJSTORE_CONFIG"
```

1.33. インストール後に SUBMARINER が接続できない場合のトラブルシューティング

Submariner を設定しても正常に実行できない場合は、次の手順を実行して問題を診断します。

1.33.1. 現象: Submariner がインストール後に接続されない

インストール後、Submariner ネットワークが通信していません。

1.33.2. 問題の特定: Submariner がインストール後に接続されない

Submariner のデプロイ後にネットワーク接続が確立されない場合は、トラブルシューティング手順を開始します。Submariner をデプロイすると、プロセスが完了するまでに数分かかる場合があることに注意してください。

1.33.3. 問題の解決: Submariner がインストール後に接続されない

デプロイメントした後で Submariner が正常に実行できない場合は、次の手順を実行します。

1. Submariner のコンポーネントが正しくデプロイメントされているかどうかを判断するには、次の要件を確認してください。
   - **submariner-addon** Pod は、ハブクラスターの **open-cluster-management** namespace で実行されています。
   - 次の Pod は、各マネージドクラスターの **submariner-operator** namespace で実行されています。
     - submariner-addon
     - submariner-gateway
     - submariner-routeagent
     - submariner-operator
     - submariner-globalnet (ClusterSet で Globalnet が有効になっている場合のみ)
     - submariner-lighthouse-agent
     - submariner-lighthouse-coredns
     - submariner-networkplugin-syncer (指定された CNI 値が **OVNKubernetes** の場合のみ)
     - submariner-metrics-proxy

2. **subctl diagnose all** コマンドを実行して、必要な Pod (**submariner-addon** Pod は除く) のステータスを確認します。

3. 必ず **must-gather** コマンドを実行して、デバッグ問題で役立つログを収集してください。

1.34. SUBMARINER アドオンのステータスが低下してい場合のトラブルシューティング

クラスターセット内のクラスターに Submariner アドオンを追加すると、**Connection status**、**Agent status**、および **Gateway nodes** のステータスにクラスターの予期しないステータスが表示されます。

1.34.1. 現象: Submariner のアドオンステータスが低下している

クラスターセット内のクラスターに Submariner アドオンを追加すると、クラスターの **Gateway nodes**、**Agent status**、および **Connection status** に次のステータスが表示されます。

- ラベルの付いたゲートウェイノード
  - **Progressing**: ゲートウェイノードにラベルを付けるプロセスが開始されました。
  - **Nodes not labeled**: ゲートウェイノードはラベル付けされていません。おそらく、それらにラベルを付けるプロセスが完了していないためです。
  - **Nodes not labeled**: おそらくプロセスが別のプロセスの終了を待機しているため、ゲートウェイノードはまだラベル付けされていません。
  - Nodes labeled: ゲートウェイノードにはラベルが付けられています。
- エージェントのステータス
  - Progressing: Submariner エージェントのインストールが開始されました。
  - Degraded: Submariner エージェントが正しく実行されていません。おそらく、まだ進行中です。
- 接続状態
  - Progressing: Submariner アドオンとの接続を確立するプロセスが開始されました。
  - Degraded: 接続の準備ができていません。アドオンをインストールしたばかりの場合は、プロセスがまだ進行中である可能性があります。接続がすでに確立されて実行された後である場合は、2 つのクラスターが相互に接続を失っています。複数のクラスターがある場合、いずれかのクラスターが切断状態にあると、すべてのクラスターに **Degraded** ステータスが表示されます。

また、接続されているクラスターと切断されているクラスターも表示されます。

1.34.2. 問題の解決: Submariner のアドオンステータスが低下している

- degraded ステータスは、プロセスが完了すると自動的に解決することがよくあります。表のステータスをクリックすると、プロセスの現在のステップを確認できます。その情報を使用して、プロセスが終了したかどうかを判断し、他のトラブルシューティング手順を実行する必要があります。
- それ自体で解決しない問題の場合は、次の手順を実行して問題のトラブルシューティングを行います。

1. 次の条件が存在する場合、**subctl** ユーティリティーで **diagnose** コマンドを使用して、Submariner 接続でいくつかのテストを実行できます。

   a. **Agent status** または **Connection status** は **Degraded** 状態です。**diagnose** コマンドは、問題に関する詳細な分析を提供します。

   b. コンソールではすべてが緑色ですが、ネットワーク接続が正しく機能していません。**diagnose** コマンドは、コンソールの外部に他の接続またはデプロイメントの問題がないことを確認するのに役立ちます。問題を特定するために、デプロイメント後に **diagnostics** コマンドを実行することを推奨します。
   コマンドの実行方法の詳細は、Submariner の **diagnose** を参照してください。

2. **Connection status** で問題が続く場合は、**subctl** ユーティリティーツールの **diagnose** コマンドを実行して、2 つの Submariner クラスター間の接続のより詳細なステータスを取得することから始めることができます。コマンドの形式は次のとおりです。

   ```
   subctl diagnose all --kubeconfig <path-to-kubeconfig-file>
   ```

   path-to-kubeconfig-file を **kubeconfig** ファイルへのパスに置き換えます。コマンドの詳細は、Submariner のドキュメントの **diagnose** を参照してください。

3. ファイアウォールの設定を確認してください。接続の問題は、クラスターの通信を妨げるファイアウォールのアクセス許可の問題が原因で発生する場合があります。これにより、**Connection status** が degraded として表示される可能性があります。次のコマンドを実行して、ファイアウォールの問題を確認します。

   ```
   subctl diagnose firewall inter-cluster <path-to-local-kubeconfig> <path-to-remote-cluster-kubeconfig>
   ```

   path-to-local-kubeconfig を、いずれかのクラスターの **kubeconfig** ファイルへのパスに置き換えます。

   path-to-remote-kubeconfig を、他のクラスターの **kubeconfig** ファイルへのパスに置き換えます。**subctl** ユーティリティーツールで **verify** コマンドを実行して、2 つの Submariner クラスター間の接続をテストできます。コマンドの基本的な形式は次のとおりです。

4. **Connection status** で問題が続く場合は、**subctl** ユーティリティーツールで **verify** コマンドを実行して、2 つの Submariner クラスター間の接続をテストできます。コマンドの基本的な形式は次のとおりです。

   ```
   subctl verify --kubecontexts <cluster1>,<cluster2> [flags]
   ```

   cluster1 と **cluster2** を、テストしているクラスターの名前に置き換えます。コマンドの詳細は、Submariner のドキュメントの **verify** を参照してください。

5. トラブルシューティング手順で問題が解決したら、**subctl** ツールで **benchmark** コマンドを使用して、追加の診断を実行するときに比較する基準を確立します。
   コマンドのオプションの詳細は、Submariner のドキュメントの **benchmark** を参照してください。

1.35. 復元ステータスがエラーで終了する場合のトラブルシューティング

バックアップを復元すると、リソースは正しく復元されますが、Red Hat Advanced Cluster Management の復元リソースには **FinishedWithErrors** ステータスが表示されます。

1.35.1. 現象: 復元ステータスのトラブルシューティングがエラーで終了する

Red Hat Advanced Cluster Management は **FinishedWithErrors** ステータスを示し、Red Hat Advanced Cluster Management の復元で作成された 1 つ以上の Velero 復元リソースは **PartiallyFailed** ステータスを示します。

1.35.2. 問題の解決: 復元ステータスのトラブルシューティングがエラーで終了する

空のバックアップから復元する場合は、**FinishedWithErrors** ステータスを無視しても安全です。

Red Hat Advanced Cluster Management for Kubernetes の復元では、すべての Velero リストアリソースの累積ステータスが表示されます。1 つのステータスが **PartiallyFailed** で、他のステータスが **Completed** の場合、少なくとも 1 つの問題があることが示すために累積ステータスとして **PartiallyFailed** が表示されます。

この問題を解決するには、ステータスが **PartiallyFailed** になっているすべての Velero 復元リソースのステータスを個別に確認し、さらにログで詳細を確認します。ログはオブジェクトストレージから直接取得するか、**DownloadRequest** カスタムリソースを使用して OADP Operator からダウンロードできます。

コンソールから **DownloadRequest** を作成するには、次の手順を実行します。

1. **Operators** > **Installed Operators** > **Create DownloadRequest** に移動します。
2. **Kind** として **BackupLog** を選択し、コンソールの指示に従って **DownloadRequest** の作成を完了します。

1.36. ハブクラスターのバックアップを復元すると汎用リソースが削除される

ハブクラスターのバックアップを復元し、**Restore.cluster.open-cluster-management.io** リソースで **cleanupBeforeRestore: CleanupRestored** パラメーターを使用すると、**acm-resources-generic-schedule** バックアップで作成されたリソースが削除される可能性があります。

1.36.1. 現象: ハブクラスターのバックアップを復元すると汎用リソースが削除される

acm-resources-generic-schedule バックアップでバックアップされたリソースは、復元されたハブクラスターには表示されません。バックアップオペレーターのログを確認すると、次のようなメッセージが表示されます。

```
_2023-06-08T13:42:48.066572033Z 2023-06-08T13:42:48.066Z    INFO    Deleting resource DRPlacementControl [c1-helloworld-placement-1-drpc.c1-helloworld]    {"controller": "restore", "controllerGroup": "cluster.open-cluster-management.io", "controllerKind": "Restore", "restore": {"name":"restore-acm","namespace":"open-cluster-management-backup"}
```

1.36.2. 問題の解決: ハブクラスターのバックアップを復元すると汎用リソースが削除される

次の状況が発生した場合、リソースは削除されます。

- **acm-resources-generic-schedule** バックアップによってバックアップされたリソースが、**cluster.open-cluster-management.io/backup** ラベルを持つ Secret または ConfigMap リソースタイプと一致しない。

- **Restore.cluster.open-cluster-management.io** リソースを使用するリストアを実行し、**cleanupBeforeRestore:** 値を **CleanupRestored** に設定します。

- 最新の Red Hat Advanced Cluster Management バックアップセットには **acm-resources-schedule** バックアップが含まれていないため、古いバージョンのバックアップが選択されます。その結果、**acm-resources-schedule** バックアップのタイムスタンプは、**acm-resources-generic-schedule** バックアップとは異なります。復元後の操作中に CleanRestore オプションが処理されると、すべての汎用リソースは **acm-resources-schedule backup** と同じタイムスタンプを持たないため、クリーンアップされます。

この問題を解決するには、復元操作を再度実行して、**cleanupBeforeRestore:** 値を **None** に設定します。

1.37. 複数行の YAML 解析のトラブルシューティング

fromSecret 関数を使用して **Secret** リソースのコンテンツを **Route** リソースに追加すると、コンテンツが正しく表示されません。

1.37.1. 現象: 複数行の YAML 解析のトラブルシューティング

マネージドクラスターとハブクラスターが同じクラスターである場合、証明書データは編集されるため、内容はテンプレート JSON 文字列として解析されません。次のエラーメッセージが表示される場合があります。

```
message: >-
        [spec.tls.caCertificate: Invalid value: "redacted ca certificate
        data": failed to parse CA certificate: data does not contain any
        valid RSA or ECDSA certificates, spec.tls.certificate: Invalid
        value: "redacted certificate data": data does not contain any valid
        RSA or ECDSA certificates, spec.tls.key: Invalid value: "": no key specified]
```

1.37.2. 問題の解決: 複数行の YAML 解析のトラブルシューティング

ハブクラスターとマネージドクラスターの **fromSecret** 値を取得するように証明書ポリシーを設定します。**autoindent** 機能を使用して、次の内容で証明書ポリシーを更新します。

```
            tls:
              certificate: |
                {{ print "{{hub fromSecret "open-cluster-management" "minio-cert" "tls.crt" hub}}" |
base64dec | autoindent }}
```