OpenShift Container Platform 4.5

サポート

OpenShift Container Platform のサポート

Last Updated: 2021-07-13

OpenShift Container Platform 4.5 サポート

OpenShift Container Platform のサポート

法律上の通知

Copyright © 2021 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

本書では、OpenShift Container Platform についての Red Hat サポートを得る方法についての情報を提供します。また、Telemetry および Insights Operator を使用したリモートヘルスモニタリングについての情報も含まれます。

目次

第1章 サポート

1.1. サポート

本書で説明されている手順、または OpenShift Container Platform について問題が発生した場合は、Red Hat カスタマーポータル にアクセスしてください。カスタマーポータルでは、次のことができます。

- Red Hat 製品に関するアーティクルおよびソリューションについての Red Hat ナレッジベースの検索またはブラウズ。
- Red Hat サポートに対するサポートケースの送信。
- 他の製品ドキュメントへのアクセス。

クラスターの問題を特定するには、Red Hat OpenShift Cluster Manager で Insights を使用できます。Insights により、問題の詳細と、利用可能な場合は問題の解決方法に関する情報が提供されます。

本書の改善が提案される場合や、エラーが見つかった場合は、**Documentation** コンポーネントの **OpenShift Container Platform** 製品に対して、Bugzilla レポート を送信してください。セクション名や OpenShift Container Platform バージョンなどの具体的な情報を提供してください。

1.2. RED HAT ナレッジベースについて

Red Hat ナレッジベース は、お客様が Red Hat の製品やテクノロジーを最大限に活用できるようにするための豊富なコンテンツを提供します。Red Hat ナレッジベースは、Red Hat 製品のインストール、設定、および使用に関する記事、製品ドキュメント、および動画で構成されています。さらに、簡潔な根本的な原因についての説明や修正手順を説明した既知の問題のソリューションを検索できます。

1.3. RED HAT ナレッジベースの検索

OpenShift Container Platform の問題が発生した場合には、初期検索を実行して、解決策を Red Hat ナレッジベース内ですでに見つけることができるかどうかを確認できます。

前提条件

- Red Hat カスタマーポータルのアカウントがある。

手順

1. Red Hat カスタマーポータル にログインします。
2. 主な Red Hat カスタマーポータルの検索フィールドには、問題に関連する入力キーワードおよび文字列を入力します。これらには、以下が含まれます。
   - OpenShift Container Platform コンポーネント (**etcd** など)
   - 関連する手順 (**installation** など)
   - 明示的な失敗に関連する警告、エラーメッセージ、およびその他の出力
3. **Search** をクリックします。
4. **OpenShift Container Platform** 製品フィルターを選択します。

5. **ナレッジベース** のコンテンツタイプフィルターを選択します。

1.4. サポートケースの送信

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- Red Hat カスタマーポータルのアカウントがある。
- Red Hat の標準またはプレミアムサブスクリプションがある。

手順

1. Red Hat カスタマーポータル にログインし、**SUPPORT CASES → Open a case** を選択します。
2. 問題の該当するカテゴリー (**Defect / Bug** など)、製品 (**OpenShift Container Platform**)、および製品バージョン (すでに自動入力されていない場合は **4.5**) を選択します。
3. 報告されている問題に対する一致に基づいて提案される Red Hat ナレッジベースソリューションの一覧を確認してください。提案されている記事が問題に対応していない場合は、**Continue** をクリックします。
4. 問題についての簡潔で説明的な概要と、確認されている現象および予想される動作についての詳細情報を入力します。
5. 報告されている問題に対する一致に基づいて提案される Red Hat ナレッジベースソリューションの更新された一覧を確認してください。ケース作成プロセスでより多くの情報を提供すると、この一覧の絞り込みが行われます。提案されている記事が問題に対応していない場合は、**Continue** をクリックします。
6. アカウント情報が予想通りに表示されていることを確認し、そうでない場合は適宜修正します。
7. 自動入力された OpenShift Container Platform クラスター ID が正しいことを確認します。正しくない場合は、クラスター ID を手動で取得します。
   - OpenShift Container Platform Web コンソールを使用してクラスター ID を手動で取得するには、以下を実行します。
     a. **Home → Dashboards → Overview** に移動します。
     b. **Details** セクションの **Cluster ID** フィールドで値を見つけます。
   - または、OpenShift Container Platform Web コンソールで新規サポートケースを作成し、クラスター ID を自動的に入力することができます。
     a. ツールバーから、**(?) Help → Open Support Case** に移動します。
     b. **Cluster ID** 値が自動的に入力されます。
   - OpenShift CLI (**oc**) を使用してクラスター ID を取得するには、以下のコマンドを実行します。

```
$ oc get clusterversion -o jsonpath='{.items[].spec.clusterID}{"\n"}'
```

8. プロンプトが表示されたら、以下の質問に入力し、**Continue** をクリックします。
    - 動作はどこで発生しているか?どの環境を使用しているか?
    - 動作はいつ発生するか?頻度は?繰り返し発生するか?特定のタイミングで発生するか?
    - 時間枠およびビジネスへの影響について提供できるどのような情報があるか?
9. 関連する診断データファイルをアップロードし、**Continue** をクリックします。まず **oc adm must-gather** コマンドを使用して収集されるデータと、そのコマンドによって収集されない問題に固有のデータを含めることが推奨されます。
10. 関連するケース管理の詳細情報を入力し、**Continue** をクリックします。
11. ケースの詳細をプレビューし、**Submit** をクリックします。

1.5. 追加リソース

- クラスターの問題を特定する方法についての詳細は、Insights の使用によるクラスター関連の問題の特定について参照してください。

第2章 接続クラスターを使用したリモートヘルスモニタリング

2.1. リモートヘルスモニタリングについて

OpenShift Container Platform は、クラスターについての Telemetry および設定データを収集し、Telemeter Client および Insights Operator を使用してこれを Red Hat にレポートします。Red Hat に提供されるデータは、本書で説明されている利点を提供します。

Telemetry および Insights Operator 経由でデータを Red Hat にレポートするクラスターは **接続クラスター (connected cluster)** と見なされます。

Telemetry は、Red Hat が OpenShift Container Platform Telemeter Client で Red Hat に送信される情報を記述するために使用する用語です。軽量の属性は、サブスクリプション管理の自動化、クラスターの健全性の監視、サポートの支援、お客様のエクスペリエンスの向上を図るために接続されたクラスターから Red Hat に送信されます。

Insights Operator は OpenShift Container Platform 設定データを収集し、これを Red Hat に送信します。データは、クラスターがさらされる可能性のある問題に関する洞察を生み出すために使用されます。これらの洞察は、cloud.redhat.com/openshift のクラスター管理者に通信されます。

これらの 2 つのプロセスについての詳細は、本書を参照してください。

Telemetry および Insights Operator の利点

Telemetry および Insights Operator はエンドユーザーに以下の利点を提供します。

- **問題の特定および解決の強化。**エンドユーザーには正常と思われるイベントも、Red Hat が複数のお客様の幅広い視点から観察します。この視点により、一部の問題はより迅速に特定され、エンドユーザーがサポートケースを作成したり、Bugzilla を作成しなくても解決することが可能です。

- **高度なリリース管理。**OpenShift Container Platform は **candidate**、**fast**、および **stable** リリースチャネルを提供し、これにより更新ストラテジーを選択することができます。リリースの **fast** から **stable** に移行できるかどうかは、更新の成功率やアップグレード時に確認されるイベントに依存します。接続されたクラスターが提供する情報により、Red Hat はリリースの品質を **stable** チャネルに引き上げ、**fast** チャネルで見つかった問題により迅速に対応することができます。

- **ターゲットが絞られた新機能の優先付け。**収集されるデータは、最も使用される OpenShift Container Platform の領域に関する洞察を提供します。この情報により、Red Hat はお客様に最も大きな影響を与える新機能の開発に重点的に取り組むことができます。

- **効率されたサポートエクスペリエンス。**Red Hat カスタマーポータル でサポートチケットを作成する際に、接続されたクラスターのクラスター ID を指定できます。これにより、Red Hat は接続された情報を使用してクラスター固有の効率化されたサポートエクスペリエンスを提供することができます。本書には、強化されたサポートエクスペリエンスについての詳細情報を提供しています。

- **予測分析。**cloud.redhat.com/openshift に表示されるクラスターについての洞察は、接続されたクラスターから収集される情報によって有効にされます。Red Hat は、OpenShift Container Platform クラスターがさらされる問題を特定するのに役立つディープラーニング (深層学習)、機械学習、および人工知能の自動化の適用に取り組んでいます。

2.1.1. Telemetry について

Telemetry は厳選されたクラスターモニタリングメトリクスのサブセットを Red Hat に送信します。Telemeter Client はメトリクス値を 4 分 30 秒ごとにフェッチし、データを Red Hat にアップロードします。これらのメトリクスについては、本書で説明しています。

このデータのストリームは、Red Hat によってリアルタイムでクラスターをモニターし、お客様に影響を与える問題に随時対応するために使用されます。またこれにより、Red Hat がサービスへの影響を最小限に抑えつつつアップグレードエクスペリエンスの継続的な改善に向けた OpenShift Container Platform のアップグレードの展開を可能にします。

このデバッグ情報は、サポートケースでレポートされるデータへのアクセスと同じ制限が適用された状態で Red Hat サポートおよびエンジニアリングチームが利用できます。接続クラスターのすべての情報は、OpenShift Container Platform をより使用しやすく、より直感的に使用できるようにするために Red Hat によって使用されます。

追加リソース

- クラスターの更新またはアップグレードについての詳細は、OpenShift Container Platform の更新についてのドキュメントを参照してください。

2.1.1.1. Telemetry で収集される情報

以下の情報は、Telemetry によって収集されます。

- インストール時に生成される一意でランダムな識別子
- OpenShift Container Platform クラスターのバージョン情報、および更新バージョンの可用性を特定するために使用されるインストールの更新の詳細を含むバージョン情報
- クラスターごとに利用可能な更新の数、更新に使用されるチャネルおよびイメージリポジトリー、更新の進捗情報、および更新で発生するエラーの数などの更新情報
- OpenShift Container Platform がデプロイされているプラットフォームの名前およびデータセンターの場所
- CPU コアの数およびそれぞれに使用される RAM の容量を含む、クラスター、マシンタイプ、およびマシンについてのサイジング情報
- etcd メンバーの数および etcd クラスターに保存されるオブジェクトの数
- クラスターにインストールされている OpenShift Container Platform フレームワークコンポーネントおよびそれらの状態とステータス
- コンポーネント、機能および拡張機能に関する使用状況の情報
- テクノロジープレビューおよびサポート対象外の設定に関する使用状況の詳細
- 動作が低下したソフトウェアに関する情報
- **NotReady** とマークされているノードについての情報
- 動作が低下した Operator の「関連オブジェクト」として一覧表示されるすべての namespace のイベント
- Red Hat サポートがお客様にとって有用なサポートを提供するのに役立つ設定の詳細。これには、クラウドインフラストラクチャーレベルのノード設定、ホスト名、IP アドレス、Kubernetes Pod 名、namespace、およびサービスが含まれます。

- 証明書の有効性についての情報

Telemetry は、ユーザー名やパスワードなどの識別情報を収集しません。Red Hat は、個人情報を収集することを意図していません。Red Hat は、個人情報が誤って受信したことを検知した場合に、該当情報を削除します。Telemetry データが個人データを構成する場合において、Red Hat のプライバシー方針については、「Red Hat Privacy Statement 」を参照してください。

追加リソース

- Telemetry が OpenShift Container Platform で Prometheus から収集する属性を一覧表示する方法についての詳細は、「Telemetry によって収集されるデータの表示」を参照してください。

- Telemetry が Prometheus から収集する属性の一覧については、アップストリームの cluster-monitoring-operator ソースコードを参照してください。

- Telemetry はデフォルトでインストールされ、有効にされます。リモートヘルスレポートをオプトアウトする必要がある場合は、「リモートヘルスレポートのオプトアウト」を参照してください。

2.1.2. Insights Operator について

Insights Operator は設定およびコンポーネントの障害ステータスを定期的に収集し、デフォルトで 2 時間ごとにそのデータを Red Hat に報告します。この情報により、Red Hat は設定や Telemetry で報告されるデータよりも深層度の高いデータを評価できます。

OpenShift Container Platform のユーザーは、Red Hat OpenShift Cluster Manager に各クラスターのレポートを表示できます。問題が特定されると、Insights は詳細を提供します。利用可能な場合は、問題の解決方法に関する手順が提供されます。

Insights Operator は、ユーザー名、パスワード、または証明書などの識別情報を収集しません。Red Hat Insights のデータ収集とコントロールの詳細は、「Red Hat Insights Data & Application Security 」を参照してください。

Red Hat は、接続されたすべてのクラスター情報を使用して、以下を実行します。

- Red Hat OpenShift Cluster Manager でクラスターの問題をプロアクティブに特定し、解決策と防止手段となるアクションを提供します。

- 集計される情報および重要な情報を製品およびサポートチームに提供し、OpenShift Container Platform の強化を図ります。

- OpenShift Container Platform の直感的な使用方法

追加リソース

- Insights Operator はデフォルトでインストールされ、有効にされます。リモートヘルスレポートをオプトアウトする必要がある場合は、「リモートヘルスレポートのオプトアウト」を参照してください。

2.1.2.1. Insights Operator によって収集される情報

以下の情報は、Insights Operator によって収集されます。

- OpenShift Container Platform バージョンおよび環境に固有の問題を特定するためのクラスターおよびそのコンポーネントについての一般的な情報

- 誤った設定や設定するパラメーターに固有の問題の判別に使用するクラスターのイメージレジストリー設定などの設定ファイル
- クラスターコンポーネントで発生するエラー
- 実行中の更新の進捗情報、およびコンポーネントのアップグレードのステータス
- Amazon Web Services などの OpenShift Container Platform がデプロイされるプラットフォームや、クラスターが置かれるリージョンについての詳細情報
- Operator が問題を報告する場合、 **openshift-*** および **kube-*** プロジェクトのコア OpenShift Container Platform についての情報が収集されます。これには、状態、リソース、セキュリティーコンテキスト、ボリューム情報などが含まれます。

追加リソース

- Insights Operator によって収集されるデータを確認する方法についての詳細は、「Insights Operator によって収集されるデータの表示」を参照してください。
- Insights Operator のソースコードは確認したり、提供したりできます。Insights Operator によって収集される項目の一覧については、Insights Operator のアップストリームプロジェクトを参照してください。

2.1.3. Telemetry および Insights Operator データフローについて

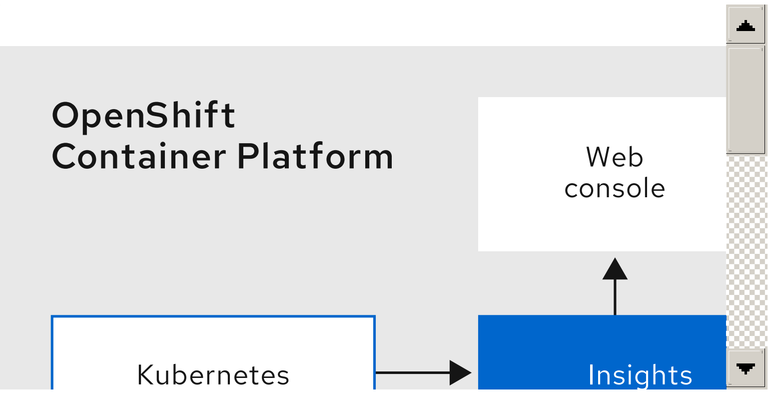
Telemeter Client は、Prometheus API から選択した時系列データを収集します。時系列データは、処理するために 4 分 30 秒ごとに api.openshift.com にアップロードされます。

Insights Operator は、選択したデータを Kubernetes API および Prometheus API からアーカイブに収集します。アーカイブは、処理するために 2 時間ごとに cloud.redhat.com にアップロードされます。さらに Insights Operator は、cloud.redhat.com から最新の Insights 分析をダウンロードします。これは、OpenShift Container Platform Web コンソールの **Overview** ページに含まれる **Insights status** ポップアップを設定するために使用されます。

Red Hat との通信はすべて、Transport Layer Security (TLS) および相互証明書認証を使用して、暗号化されたチャネル上で行われます。すべてのデータは移動中および停止中に暗号化されます。

顧客データを処理するシステムへのアクセスは、マルチファクター認証と厳格な認証制御によって制御されます。アクセスは関係者以外極秘で付与され、必要な操作に制限されます。

Telemetry および Insights Operator データフロー



追加リソース

- OpenShift Container Platform モニタリングスタックについての詳細は、モニタリングスタックについて参照してください。
- ファイアウォールを設定し、Telemetry および Insights のエンドポイントの有効にする方法についての詳細は、「ファイアウォールの設定」を参照してください。

2.1.4. リモートヘルスモニタリングデータの使用方法に関する追加情報

リモートヘルスモニタリングを有効にするために収集される情報は、Telemetry および Insights Operator によって収集 される情報について参照し てください。

本書の前のセクションで説明したように、Red Hat は、サポートおよびアップグレードの提供、パフォーマンス/設定の最適化、サービスへの影響の最小化、脅威の特定および修復、トラブルシューティング、オファリングおよびユーザーエクスペリエンスの強化、問題への対応および課金の目的で (該当する場合)、Red Hat 製品のお客様の使用についてのデータを収集します。

収集における対策

Red Hat は、Telemetry および設定データを保護する目的で定められた技術および組織上の対策を講じます。

共有

Red Hat は、ユーザーエクスペリエンスの向上に向けて、Telemetry および Insights Operator で収集されるデータを内部で共有する場合があります。Red Hat は、以下の目的で Red Hat のビジネスパートナーと、お客様を特定しない集約された形式で Telemetry および設定データを共有する場合があります。つまり、パートナーが 市場およびお客様の Red Hat のオファリングの使用についてより良く理解できるように支援することを目的とするか、またはそれらのパートナーと共同でサポートしている製品の統合を効果的に行うことを目的としています。

サードパーティーのサービスプロバイダー

Red Hat は、Telemetry および設定データの収集と保管を支援する特定のサービスプロバイダーと連携する場合があります。

ユーザーコントロール/Telemetry および設定データ収集の有効化および無効化

「リモートヘルスレポートのオプトアウト」の手順に従って、OpenShift Container Platform Telemetry および Insights Operator を無効にすることができき ます。

2.2. リモートヘルスモニタリングによって収集されるデータの表示

管理者は、Telemetry および Insights Operator によって収集されるメトリクスを確認できます。

2.2.1. Telemetry によって収集されるデータの表示

Telemetry でキャプチャーされるクラスターとコンポーネントの時系列データを表示することができます。

前提条件

- OpenShift CLI (**oc**) のインストール。
- **cluster-admin** ロールまたは **cluster-monitoring-view** ロールのいずれかを持つユーザーとしてクラスターにログインする必要があります。

手順

1. OpenShift Container Platform クラスターで実行される Prometheus サービスの URL を見つけます。

```
$ oc get route prometheus-k8s -n openshift-monitoring -o jsonpath="{.spec.host}"
```

2. URL に移動します。

3. このクエリーを **Expression** 入力ボックスに入力し、**Execute** を押します。

```
{__name__=~"cluster:usage:.*|count:up0|count:up1|cluster_version|cluster_version_available_u
pdates|cluster_operator_up|cluster_operator_conditions|cluster_version_payload|cluster_install
er|cluster_infrastructure_provider|cluster_feature_set|instance:etcd_object_counts:sum|ALERT
S|code:apiserver_request_total:rate:sum|cluster:capacity_cpu_cores:sum|cluster:capacity_mem
ory_bytes:sum|cluster:cpu_usage_cores:sum|cluster:memory_usage_bytes:sum|openshift:cpu_
usage_cores:sum|openshift:memory_usage_bytes:sum|workload:cpu_usage_cores:sum|worklo
ad:memory_usage_bytes:sum|cluster:virt_platform_nodes:sum|cluster:node_instance_type_cou
nt:sum|cnv:vmi_status_running:count|node_role_os_version_machine:cpu_capacity_cores:sum
node_role_os_version_machine:cpu_capacity_sockets:sum|subscription_sync_total|csv_succee
ded|csv_abnormal|ceph_cluster_total_bytes|ceph_cluster_total_used_raw_bytes|ceph_health_s
tatus|job:ceph_osd_metadata:count|job:kube_pv:count|job:ceph_pools_iops:total|job:ceph_pool
s_iops_bytes:total|job:ceph_versions_running:count|job:noobaa_total_unhealthy_buckets:sum|j
b:noobaa_bucket_count:sum|job:noobaa_total_object_count:sum|noobaa_accounts_num|noob
aa_total_usage|console_url|cluster:network_attachment_definition_instances:max|cluster:netwo
rk_attachment_definition_enabled_instance_up:max|insightsclient_request_send_total|cam_app
_workload_migrations|cluster:apiserver_current_inflight_requests:sum:max_over_time:2m|clust
er:telemetry_selected_series:count",alertstate=~"firing|"}
```

このクエリーは、Telemetry が実行中の OpenShift Container Platform クラスターの Prometheus サービスに対して行う要求をレプリケートし、Telemetry によってキャプチャーされる時系列の完全なセットを返します。

2.2.2. Insights Operator によって収集されるデータの表示

Insights Operator で収集されるデータを確認することができます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてのクラスターへのアクセスがあること。

手順

1. Insights Operator の現在実行中の Pod の名前を検索します。

```
$ INSIGHTS_OPERATOR_POD=$(oc get pods --namespace=openshift-insights -o custom-
columns=:metadata.name --no-headers  --field-selector=status.phase=Running)
```

2. Insights Operator で収集される最近のデータアーカイブをコピーします。

```
$ oc cp openshift-insights/$INSIGHTS_OPERATOR_POD:/var/lib/insights-operator ./insights-
data
```

最近の Insights Operator アーカイブが **insights-data** ディレクトリーで利用可能になります。

2.3. リモートヘルスレポートのオプトアウト

クラスターの健全性や使用状況についてのデータのレポートをオプトアウトする必要が生じる可能性があります。

リモートヘルスレポートをオプトアウトするには、以下を実行する必要があります。

1. グローバルクラスタープルシークレットを変更してリモートヘルスレポートを無効にします。
2. この変更されたプルシークレットを使用するようにクラスターを更新します。

2.3.1. リモートヘルスレポートを無効した場合の影響

OpenShift Container Platform では、使用状況についての情報のレポートをオプトアウトできます。ただし、接続クラスターは Red Hat が問題により迅速に対応し、お客様をより効果的にサポートし、製品のアップグレードによるクラスターへの影響をより明確に把握することを可能にします。また接続されたクラスターにより、サブスクリプションとエンタイトルメントのプロセスが単純化され、Red Hat OpenShift Cluster Manager サービスによってクラスターおよびサブスクリプションのステータスについての概要を提供することが可能になります。

そのため、実稼働クラスターでのオプトアウトが必要な場合であっても、実稼働以前の環境やテストクラスターでは健全性および使用状況についてのレポートを有効な状態にしておくことが強く推奨されます。これにより、Red Hat は OpenShift Container Platform をご使用の環境に適合させ、製品関連の問題により迅速に対応する上で貢献することができます。

接続クラスターのオプトアウトによる影響には、以下が含まれます。

- Red Hat はサポートケースが作成されない限り、製品アップグレードの正常性やクラスターの健全性を監視することができません。
- Red Hat は設定データを使用して、お客様のサポートケースの優先付けや、お客様にとって重要な設定を特定することができません。
- Red Hat OpenShift Cluster Manager は健全性や使用状況についての情報を含むクラスターについてのデータを表示できません。
- 使用状況の自動レポート機能を使用できないため、サブスクリプションのエンタイトルメント情報は cloud.redhat.com で手動で入力する必要があります。

ネットワークが制限された環境の場合も、プロキシーの適切な設定により Telemetry および Insights データは依然としてレポートされます。

2.3.2. 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

既存のグローバルクラスタープルシークレットを変更して、リモートヘルスレポートを無効にすることができます。これにより、Telemetry と Insights Operator の両方が無効になります。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. グローバルクラスタープルシークレットをローカルファイルシステムにダウンロードします。

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

2. テキストエディターで、ダウンロードした **.dockerconfigjson** ファイルを編集します。
3. 以下のように **cloud.openshift.com** JSON エントリーを削除します。

```
"cloud.openshift.com":{"auth":"<hash>","email":"<email_address>"}
```

4. ファイルを保存します。

この変更されたプルシークレットを使用できるようにクラスターを更新できます。

2.3.3. グローバルクラスタープルシークレットの更新

クラスターのグローバルプルシークレットを更新できます。

> **警告**
>
> クラスターリソースは新規のプルシークレットに合わせて調整する必要がありますが、これにより、クラスターのユーザビリティーが一時的に制限される可能性があります。

> **警告**
>
> グローバルプルシークレットを更新すると、Machine Config Operator (MCO) が変更を同期している間にノードが再起動します。

前提条件

- アップロードする新規または変更されたプルシークレットファイルがある。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

- 以下のコマンドを実行して、クラスターのグローバルプルシークレットを更新します。

```
$ oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=<pull-secret-location> ①
```

① 新規プルシークレットファイルへのパスを指定します。

この更新はすべてのノードにロールアウトされます。これには、クラスターのサイズに応じて多少時間がかかる場合があります。この間に、ノードがドレイン (解放) され、Pod は残りのノードで再スケジュールされます。

2.4. INSIGHTS を使用したクラスターの問題の特定

Insights は、Insights Operator の送信データを繰り返し分析します。OpenShift Container Platform のユーザーは、Red Hat OpenShift Cluster Manager の各クラスターの **Insights** タブにレポートを表示できます。

2.4.1. クラスターの潜在的な問題の表示

本セクションでは、Red Hat OpenShift Cluster Manager で Insights レポートを表示する方法を説明します。

Insights はクラスターを繰り返し分析し、最新の結果を表示することに注意してください。問題を修正した場合や新しい問題が検出された場合などに、これらの結果は変更する可能性があります。

前提条件

- クラスターが Red Hat OpenShift Cluster Manager に登録されている。
- リモートヘルスレポートが有効になっている (デフォルト)。
- Red Hat OpenShift Cluster Manager にログインしている。

手順

1. 左側のペインで **Clusters** メニューをクリックします。
2. クラスターの名前をクリックして、クラスターの詳細を表示します。
3. クラスターの **Insights** タブを開きます。
   その結果に応じて、タブには以下のいずれかが表示されます。
   - **Your cluster passed all health checks** Insights がいずれの問題も特定しなかった場合。
   - Insights が検出さる問題の一覧。これらの問題には、リスクに基づいて優先度 (低「low」、中「moderate」、重要「importanto」および重大「critical」) が付けられます。
   - **No health checks to display** Insights がクラスターを分析していない場合。この分析は、クラスターがインストールされ、インターネットに接続された直後に開始します。
4. 問題がタブに表示される場合、エントリーの前にある > アイコンをクリックして詳細を確認してください。
   この問題によっては、詳細情報に Red Hat ナレッジベースアーティクルへのリンクが含まれることがあります。問題の解決方法の詳細については、**How to remediate this issue** をクリックしてください。

第3章 クラスターに関するデータの収集

サポートケースを作成する際、ご使用のクラスターについてのデバッグ情報を Red Hat サポートに提供していただくと Red Hat のサポートに役立ちます。

以下を提供することが推奨されます。

- **oc adm must-gather** コマンドを使用して収集されるデータ
- 一意のクラスター ID

3.1. MUST-GATHER ツールについて

oc adm must-gather CLI コマンドは、以下のような問題のデバッグに必要となる可能性のあるクラスターからの情報を収集します。

- リソース定義
- 監査ログ
- サービスログ

--image 引数を指定してコマンドを実行する際にイメージを指定できます。イメージを指定する際、ツールはその機能または製品に関連するデータを収集します。

oc adm must-gather を実行すると、新しい Pod がクラスターに作成されます。データは Pod で収集され、**must-gather.local** で始まる新規ディレクトリーに保存されます。このディレクトリーは、現行の作業ディレクトリーに作成されます。

3.2. RED HAT サポート用のクラスターについてのデータの収集

oc adm must-gather CLI コマンドを使用して、クラスターについてのデバッグ情報を収集できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてのクラスターへのアクセスがあること。
- OpenShift Container Platform CLI (**oc**) がインストールされていること。

手順

1. **must-gather** データを保存するディレクトリーに移動します。
2. **oc adm must-gather** コマンドを実行します。

```
$ oc adm must-gather
```

注記

このコマンドが失敗する場合 (クラスターで Pod をスケジュールできない場合など)、**oc adm inspect** コマンドを使用して特定リソースについての情報を収集します。収集する推奨リソースについては、Red Hat サポートにお問い合わせください。

注記

クラスターがネットワークが制限された環境を使用している場合、追加の手順を実行する必要があります。ミラーレジストリーに信頼される CA がある場合、まず信頼される CA をクラスターに追加する必要があります。ネットワークが制限された環境のすべてのクラスターについて、**oc adm must-gather** コマンドを使用する前に、デフォルトの **must-gather** イメージをイメージストリームとしてインポートする必要があります。

```
$ oc import-image is/must-gather -n openshift
```

3. 作業ディレクトリーに作成された **must-gather** ディレクトリーから圧縮ファイルを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar cvaf must-gather.tar.gz must-gather.local.5421342344627712289/ ①
```

① **must-gather-local.5421342344627712289/** を実際のディレクトリー名に置き換えてください。

4. 圧縮ファイルを Red Hat カスタマーポータル で作成したサポートケースに添付します。

3.3. 特定の機能に関するデータ収集

oc adm must-gather CLI コマンドを **--image** または **--image-stream** 引数と共に使用して、特定に機能についてのデバッグ情報を収集できます。**must-gather** ツールは複数のイメージをサポートするため、単一のコマンドを実行して複数の機能についてのデータを収集できます。

表3.1 サポート対象の must-gather イメージ

イメージ	目的
registry.redhat.io/container-native-virtualization/cnv-must-gather-rhel8:v2.4.9	OpenShift Virtualization のデータ収集。
registry.redhat.io/openshift-serverless-1/svls-must-gather-rhel8	OpenShift Serverless のデータ収集。
registry.redhat.io/openshift-service-mesh/istio-must-gather-rhel8	Red Hat OpenShift Service Mesh のデータ収集。
registry.redhat.io/rhmtc/openshift-migration-must-gather-rhel8:v1.4	Migration Toolkit for Containers のデータ収集。
registry.redhat.io/ocs4/ocs-must-gather-rhel8	Red Hat OpenShift Container Storage のデータ収集。
registry.redhat.io/openshift4/ose-cluster-logging-operator	Red Hat OpenShift クラスターロギングのデータ収集。

イメージ	目的
registry.redhat.io/openshift4/ose-local-storage-mustgather-rhel8	ローカルストレージ Operator のデータ収集。

注記

特定の機能データに加えてデフォルトの**must-gather** データを収集するには、**--image-stream=openshift/must-gather** 引数を追加します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてのクラスターへのアクセスがあること。
- OpenShift Container Platform CLI (**oc**) がインストールされていること。

手順

1. **must-gather** データを保存するディレクトリーに移動します。
2. **oc adm must-gather** コマンドを 1 つまたは複数の **--image** または **--image-stream** 引数と共に実行します。たとえば、以下のコマンドは、デフォルトのクラスターデータと OpenShift Virtualization に固有の情報の両方を収集します。

```
$ oc adm must-gather \
 --image-stream=openshift/must-gather \ ①
 --image=registry.redhat.io/container-native-virtualization/cnv-must-gather-rhel8:v2.4.9 ②
```

① デフォルトの OpenShift Container Platform **must-gather** イメージ

② OpenShift Virtualization の must-gather イメージ

must-gather ツールを追加の引数と共に使用し、クラスターロギングおよびクラスター内の Cluster Logging Operator に関連するデータを収集できます。クラスターロギングの場合、以下のコマンドを実行します。

```
$ oc adm must-gather --image=$(oc -n openshift-logging get deployment.apps/cluster-logging-operator \
 -o jsonpath='{.spec.template.spec.containers[?(@.name == "cluster-logging-operator")].image}')
```

例3.1 クラスターロギングの must-gather の出力例

```
├── cluster-logging
│  ├── clo
│  │  ├── cluster-logging-operator-74dd5994f-6ttgt
│  │  ├── clusterlogforwarder_cr
│  │  ├── cr
│  │  ├── csv
│  │  ├── deployment
```

```
│   │   └── logforwarding_cr
│   ├── collector
│   │   ├── fluentd-2tr64
│   ├── curator
│   │   └── curator-1596028500-zkz4s
│   ├── eo
│   │   ├── csv
│   │   ├── deployment
│   │   └── elasticsearch-operator-7dc7d97b9d-jb4r4
│   ├── es
│   │   ├── cluster-elasticsearch
│   │   │   ├── aliases
│   │   │   ├── health
│   │   │   ├── indices
│   │   │   ├── latest_documents.json
│   │   │   ├── nodes
│   │   │   ├── nodes_stats.json
│   │   │   └── thread_pool
│   │   ├── cr
│   │   ├── elasticsearch-cdm-lp8l38m0-1-794d6dd989-4jxms
│   │   └── logs
│   │       ├── elasticsearch-cdm-lp8l38m0-1-794d6dd989-4jxms
│   ├── install
│   │   ├── co_logs
│   │   ├── install_plan
│   │   ├── olmo_logs
│   │   └── subscription
│   └── kibana
│       ├── cr
│       ├── kibana-9d69668d4-2rkvz
├── cluster-scoped-resources
│   └── core
│       ├── nodes
│       │   ├── ip-10-0-146-180.eu-west-1.compute.internal.yaml
│       └── persistentvolumes
│           ├── pvc-0a8d65d9-54aa-4c44-9ecc-33d9381e41c1.yaml
├── event-filter.html
├── gather-debug.log
└── namespaces
    ├── openshift-logging
    │   ├── apps
    │   │   ├── daemonsets.yaml
    │   │   ├── deployments.yaml
    │   │   ├── replicasets.yaml
    │   │   └── statefulsets.yaml
    │   ├── batch
    │   │   ├── cronjobs.yaml
    │   │   └── jobs.yaml
    │   ├── core
    │   │   ├── configmaps.yaml
    │   │   ├── endpoints.yaml
    │   │   ├── events
    │   │   │   ├── curator-1596021300-wn2ks.162634ebf0055a94.yaml
    │   │   │   ├── curator.162638330681bee2.yaml
    │   │   │   ├── elasticsearch-delete-app-1596020400-gm6nl.1626341a296c16a1.yaml
    │   │   │   ├── elasticsearch-delete-audit-1596020400-9l9n4.1626341a2af81bbd.yaml
```

```
│   │   │   ├── elasticsearch-delete-infra-1596020400-v98tk.1626341a2d821069.yaml
│   │   │   ├── elasticsearch-rollover-app-1596020400-cc5vc.1626341a3019b238.yaml
│   │   │   ├── elasticsearch-rollover-audit-1596020400-s8d5s.1626341a31f7b315.yaml
│   │   │   ├── elasticsearch-rollover-infra-1596020400-7mgv8.1626341a35ea59ed.yaml
│   │   ├── events.yaml
│   │   ├── persistentvolumeclaims.yaml
│   │   ├── pods.yaml
│   │   ├── replicationcontrollers.yaml
│   │   ├── secrets.yaml
│   │   └── services.yaml
│   ├── openshift-logging.yaml
│   ├── pods
│   │   ├── cluster-logging-operator-74dd5994f-6ttgt
│   │   │   ├── cluster-logging-operator
│   │   │   │   └── cluster-logging-operator
│   │   │   │       └── logs
│   │   │   │           ├── current.log
│   │   │   │           ├── previous.insecure.log
│   │   │   │           └── previous.log
│   │   │   └── cluster-logging-operator-74dd5994f-6ttgt.yaml
│   │   ├── cluster-logging-operator-registry-6df49d7d4-mxxff
│   │   │   ├── cluster-logging-operator-registry
│   │   │   │   └── cluster-logging-operator-registry
│   │   │   │       └── logs
│   │   │   │           ├── current.log
│   │   │   │           ├── previous.insecure.log
│   │   │   │           └── previous.log
│   │   │   ├── cluster-logging-operator-registry-6df49d7d4-mxxff.yaml
│   │   │   └── mutate-csv-and-generate-sqlite-db
│   │   │       └── mutate-csv-and-generate-sqlite-db
│   │   │           └── logs
│   │   │               ├── current.log
│   │   │               ├── previous.insecure.log
│   │   │               └── previous.log
│   │   ├── curator-1596028500-zkz4s
│   │   ├── elasticsearch-cdm-lp8l38m0-1-794d6dd989-4jxms
│   │   ├── elasticsearch-delete-app-1596030300-bpgcx
│   │   │   ├── elasticsearch-delete-app-1596030300-bpgcx.yaml
│   │   │   └── indexmanagement
│   │   │       └── indexmanagement
│   │   │           └── logs
│   │   │               ├── current.log
│   │   │               ├── previous.insecure.log
│   │   │               └── previous.log
│   │   ├── fluentd-2tr64
│   │   │   ├── fluentd
│   │   │   │   └── fluentd
│   │   │   │       └── logs
│   │   │   │           ├── current.log
│   │   │   │           ├── previous.insecure.log
│   │   │   │           └── previous.log
│   │   │   ├── fluentd-2tr64.yaml
│   │   │   └── fluentd-init
│   │   │       └── fluentd-init
│   │   │           └── logs
│   │   │               ├── current.log
```

```
│   │   │               ├── previous.insecure.log
│   │   │               └── previous.log
│   │   ├── kibana-9d69668d4-2rkvz
│   │   │   ├── kibana
│   │   │   │   └── kibana
│   │   │   │       └── logs
│   │   │   │           ├── current.log
│   │   │   │           ├── previous.insecure.log
│   │   │   │           └── previous.log
│   │   │   ├── kibana-9d69668d4-2rkvz.yaml
│   │   │   └── kibana-proxy
│   │   │       └── kibana-proxy
│   │   │           └── logs
│   │   │               ├── current.log
│   │   │               ├── previous.insecure.log
│   │   │               └── previous.log
│   └── route.openshift.io
│       └── routes.yaml
└── openshift-operators-redhat
    ├── ...
```

3. 作業ディレクトリーに作成された **must-gather** ディレクトリーから圧縮ファイルを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar cvaf must-gather.tar.gz must-gather.local.5421342344627712289/ ①
```

① **must-gather-local.5421342344627712289/** を実際のディレクトリー名に置き換えてください。

4. 圧縮ファイルを Red Hat カスタマーポータル で作成したサポートケースに添付します。

3.4. クラスター ID の取得

Red Hat サポートに情報を提供する際には、クラスターに固有の識別子を提供していただくと役に立ちます。OpenShift Container Platform Web コンソールを使用してクラスター ID を自動入力できます。Web コンソールまたは OpenShift CLI (**oc**) を使用してクラスター ID を手動で取得することもできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてのクラスターへのアクセスがあること。
- Web コンソールまたはインストールされている OpenShift CLI (**oc**) へのアクセスがあること。

手順

- Web コンソールを使用してサポートケースを開き、クラスター ID の自動入力を行うには、以下を実行します。
    a. ツールバーから、**(?) Help** → **Open Support Case** に移動します。
    b. **Cluster ID** 値が自動的に入力されます。

- Web コンソールを使用してクラスター ID を手動で取得するには、以下を実行します。
  a. **Home** → **Dashboards** → **Overview** に移動します。
  b. 値は **Details** セクションの **Cluster ID** フィールドで利用できます。
- OpenShift CLI (**oc**) を使用してクラスター ID を取得するには、以下のコマンドを実行します。

```
$ oc get clusterversion -o jsonpath='{.items[].spec.clusterID}{"\n"}'
```

3.5. SOSREPORT について

sosreport は、設定の詳細、システム情報、および診断データを Red Hat Enterprise Linux (RHEL) および Red Hat Enterprise Linux CoreOS (RHCOS) システムから収集するツールです。**sosreport** は、ノードに関連する診断情報を収集するための標準化した方法を提供します。この情報は、問題の診断のために Red Hat サポートに提供できます。

サポートによっては、Red Hat サポートは特定の OpenShift Container Platform ノードの **sosreport** アーカイブを収集するよう依頼する場合があります。たとえば、**oc adm must-gather** の出力に含まれないシステムログまたは他のノード固有のデータを確認する必要がある場合があります。

3.6. OPENSHIFT CONTAINER PLATFORM クラスターノードの SOSREPORT アーカイブの生成

OpenShift Container Platform 4.5 クラスターノードの **sosreport** を生成する方法として、デバッグ Pod を使用することが推奨されます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- ホストへの SSH アクセスがあること。
- OpenShift CLI (**oc**) がインストールされている。
- Red Hat の標準またはプレミアムサブスクリプションがある。
- Red Hat カスタマーポータルのアカウントがある。
- 既存の Red Hat サポートケース ID がある。

手順

1. クラスターノードの一覧を取得します。

```
$ oc get nodes
```

2. ターゲットノードのデバッグセッションに入ります。この手順は、**<node_name>-debug** というデバッグ Pod をインスタンス化します。

```
$ oc debug node/my-cluster-node
```

3. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

   ```
   # chroot /host
   ```

   注記

   Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

4. **sosreport** を実行するために必要なバイナリーおよびプラグインが含まれる **toolbox** コンテナーを起動します。

   ```
   # toolbox
   ```

   注記

   既存の **toolbox** Pod がすでに実行されている場合、**toolbox** コマンドは以下を出力します: **'toolbox-' already exists.Trying to start....podman rm toolbox-** で実行中の toolbox コンテナーを削除して、**sosreport** プラグインの問題を回避するために、新規の toolbox コンテナーを生成します。

5. **sosreport** アーカイブを収集します。
   a. **sosreport** コマンドを実行して、**crio.all** および **crio.logs** CRI-O コンテナーエンジン **sosreport** プラグインを有効にします。

      ```
      # sosreport -k crio.all=on -k crio.logs=on ①
      ```

      ① **-K** により、デフォルト以外の **sosreport** プラグインパラメーターを定義できます。

   b. プロンプトが表示されたら **Enter** を押して続行します。
   c. Red Hat サポートケース ID を指定します。**sosreport** は ID をアーカイブのファイル名に追加します。
   d. **sosreport** 出力は、アーカイブの場所とチェックサムを提供します。以下の出力参照例は、ケース ID **01234567** を参照します。

      ```
      Your sosreport has been generated and saved in:
        /host/var/tmp/sosreport-my-cluster-node-01234567-2020-05-28-eyjknxt.tar.xz ①

      The checksum is: 382ffc167510fd71b4f12a4f40b97a4e
      ```

① toolbox コンテナーはホストの root ディレクトリーを **/host** にマウントするため、**sosreport** アーカイブのファイルパスは **chroot** 環境外にあります。

6. 以下の方法のいずれかを使用して、解析のために **sosreport** アーカイブを Red Hat サポートに提供します。

   - ファイルを OpenShift Container Platform クラスターから直接既存の Red Hat サポートケースにアップロードします。

     a. toolbox コンテナー内から、**redhat-support-tool** を実行してアーカイブを既存の Red Hat サポートケースに直接割り当てます。この例では、サポートケース ID **01234567** を使用します。

     ```
     # redhat-support-tool addattachment -c 01234567 /host/var/tmp/my-sosreport.tar.xz
     ①
     ```

     ① toolbox コンテナーは、ホストの root ディレクトリーを **/host** にマウントします。**redhat-support-tool** コマンドでアップロードするファイルを指定する場合は、toolbox コンテナーの root ディレクトリー (**/host/** を含む) から絶対パスを参照します。

   - 既存の Red Hat サポートケースにファイルをアップロードします。

     a. **oc debug node/<node_name>** コマンドを実行して **sosreport** アーカイブを連結し、出力をファイルにリダイレクトします。このコマンドは、直前の **oc debug** セッションを終了していることを前提としています。

     ```
     $ oc debug node/my-cluster-node -- bash -c 'cat /host/var/tmp/sosreport-my-cluster-node-01234567-2020-05-28-eyjknxt.tar.xz' > /tmp/sosreport-my-cluster-node-01234567-2020-05-28-eyjknxt.tar.xz ①
     ```

     ① デバッグコンテナーは、ホストの root ディレクトリーを **/host** にマウントします。連結のためにターゲットファイルを指定する際に、デバッグコンテナーの root ディレクトリー (**/host** を含む) から絶対パスを参照します。

     注記

     Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。**scp** を使用してクラスターノードから **sosreport** アーカイブを転送することは推奨されず、ノードには **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この状態では、**scp core@<node>.<cluster_name>.<base_domain>:<file_path> <local_path>** を実行して、ノードから **sosreport** アーカイブをコピーすることができます。

     b. https://access.redhat.com/support/cases/ 内の既存のサポートケースに移動します。

     c. **Attach files** を選択し、プロンプトに従ってファイルをアップロードします。

3.7. ブートストラップノードのジャーナルログのクエリー

ブートストラップ関連の問題が発生した場合、ブートストラップノードから **bootkube.service** の**journald** ユニットログおよびコンテナーログを収集できます。

前提条件

- ブートストラップノードへの SSH アクセスがある。
- ブートストラップノードの完全修飾ドメイン名がある。

手順

1. OpenShift Container Platform のインストール時にブートストラップノードから **bootkube.service** の**journald** ユニットログをクエリーします。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> journalctl -b -f -u bootkube.service
```

注記

ブートストラップノードで **bootkube.service** のログは etcd の **connection refused** エラーを出力し、ブートストラップサーバーがマスターノードの etcd に接続できないことを示します。etcd が各マスターノードで起動し、ノードがクラスターに参加した後には、エラーは発生しなくなるはずです。

2. ブートストラップノードで **podman** を使用してブートストラップノードのコンテナーからログを収集します。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> 'for pod in $(sudo podman ps -a -q); do sudo podman logs
$pod; done'
```

3.8. クラスターノードジャーナルログのクエリー

個別のクラスターノードの **/var/log** 内で **journald** ユニットログおよびその他のログを収集できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。

手順

1. OpenShift Container Platform クラスターノードから **kubelet** の**journald** ユニットログをクエリーします。以下の例では、マスターノードのみがクエリーされます。

```
$ oc adm node-logs --role=master -u kubelet ①
```

–

① 他のユニットログをクエリーするために、**kubelet** を適宜置き換えます。

2. クラスターノードの **/var/log/** の下にある特定のサブディレクトリーからログを収集します。
   a. **/var/log/** サブディレクトリー内に含まれるログの一覧を取得します。以下の例では、すべてのマスターノードの **/var/log/openshift-apiserver/** にあるファイルを一覧表示します。

      ```
      $ oc adm node-logs --role=master --path=openshift-apiserver
      ```

   b. **/var/log/** サブディレクトリー内の特定ログを確認します。以下の例は、すべてのマスターノードから **/var/log/openshift-apiserver/audit.log** コンテンツを出力します。

      ```
      $ oc adm node-logs --role=master --path=openshift-apiserver/audit.log
      ```

   c. API が機能しない場合は、代わりに SSH を使用して各ノードのログを確認します。以下の例は、**/var/log/openshift-apiserver/audit.log** をベースとしています。

      ```
      $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo tail -f
      /var/log/openshift-apiserver/audit.log
      ```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

3.9. OPENSHIFT CONTAINER PLATFORM ノードまたはコンテナーからのネットワークトレースの収集

ネットワーク関連の OpenShift Container Platform の潜在的な問題を調査する際に、Red Hat サポートは特定の OpenShift Container Platform クラスターノードまたは特定のコンテナーからネットワークパケットトレースを要求する可能性があります。OpenShift Container Platform でネットワークトレースをキャプチャーする方法として、デバッグ Pod を使用できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- Red Hat の標準またはプレミアムサブスクリプションがある。
- Red Hat カスタマーポータルのアカウントがある。
- 既存の Red Hat サポートケース ID がある。

- ホストへの SSH アクセスがあること。

手順

1. クラスターノードの一覧を取得します。

```
$ oc get nodes
```

2. ターゲットノードのデバッグセッションに入ります。この手順は、**<node_name>-debug** というデバッグ Pod をインスタンス化します。

```
$ oc debug node/my-cluster-node
```

3. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

```
# chroot /host
```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

4. **chroot** 環境コンソール内から、ノードのインターフェース名を取得します。

```
# ip ad
```

5. **sosreport** を実行するために必要なバイナリーおよびプラグインが含まれる **toolbox** コンテナーを起動します。

```
# toolbox
```

注記

既存の **toolbox** Pod がすでに実行されている場合、**toolbox** コマンドは以下を出力します: **'toolbox-' already exists.Trying to start....tcpdump** の問題が発生するのを回避するには、**podman rm toolbox-** で実行中の toolbox コンテナーを削除し、新規の toolbox コンテナーを生成します。

6. クラスターノードで **tcpdump** セッションを開始し、出力をキャプチャーファイルにリダイレクトします。この例では、**ens5** をインターフェース名として使用します。

```
$ tcpdump -nn -s 0 -i ens5 -w /host/var/tmp/my-cluster-node_$(date +%d_%m_%Y-
%H_%M_%S-%Z).pcap ①
```

① toolbox コンテナーはホストの root ディレクトリーを **/host** にマウントするため、**tcpdump** キャプチャーファイルのパスは **chroot** 環境外にあります。

7. ノード上の特定コンテナーに **tcpdump** キャプチャーが必要な場合は、以下の手順に従います。
   a. ターゲットコンテナー ID を確認します。toolbox コンテナーはホストの root ディレクトリーを **/host** にマウントするため、この手順では、**chroot host** コマンドが **crictl** コマンドの前に実行されます。

      ```
      # chroot /host crictl ps
      ```

   b. コンテナーのプロセス ID を確認します。この例では、コンテナー ID は **a7fe32346b120** です。

      ```
      # chroot /host crictl inspect --output yaml a7fe32346b120 | grep 'pid' | awk '{print $2}'
      ```

   c. コンテナーで **tcpdump** セッションを開始し、出力をキャプチャーファイルにリダイレクトします。この例では、**49628** をコンテナーのプロセス ID として使用し、**ens5** をインターフェース名として使用します。**nsenter** コマンドはターゲットプロセスの namespace に入り、その namespace でコマンドを実行します。この例ではターゲットプロセスがコンテナーのプロセス ID であるため、**tcpdump** コマンドはホストからコンテナーの namespace で実行されます。

      ```
      # nsenter -n -t 49628 -- tcpdump -nn -i ens5 -w /host/var/tmp/my-cluster-node-my-container_$(date +%d_%m_%Y-%H_%M_%S-%Z).pcap.pcap ①
      ```

      ① toolbox コンテナーはホストの root ディレクトリーを **/host** にマウントするため、**tcpdump** キャプチャーファイルのパスは **chroot** 環境外にあります。

8. 以下の方法のいずれかを使用して、分析用に **tcpdump** キャプチャーファイルを Red Hat サポートに提供します。
   - ファイルを OpenShift Container Platform クラスターから直接既存の Red Hat サポートケースにアップロードします。
     a. toolbox コンテナー内から、**redhat-support-tool** を実行してファイルディレクトリーを既存の Red Hat サポートケースに直接割り当てます。この例では、サポートケース ID **01234567** を使用します。

        ```
        # redhat-support-tool addattachment -c 01234567 /host/var/tmp/my-tcpdump-capture-file.pcap ①
        ```

        ① toolbox コンテナーは、ホストの root ディレクトリーを **/host** にマウントします。**redhat-support-tool** コマンドでアップロードするファイルを指定する場合は、toolbox コンテナーの root ディレクトリー (**/host/** を含む) から絶対パスを参照します。

   - 既存の Red Hat サポートケースにファイルをアップロードします。
     a. **oc debug node/<node_name>** コマンドを実行して **sosreport** アーカイブを連結し、出力をファイルにリダイレクトします。このコマンドは、直前の **oc debug** セッションを終了していることを前提としています。

```
$ oc debug node/my-cluster-node -- bash -c 'cat /host/var/tmp/my-tcpdump-capture-file.pcap' > /tmp/my-tcpdump-capture-file.pcap ①
```

① デバッグコンテナーは、ホストの root ディレクトリーを **/host** にマウントします。連結のためにターゲットファイルを指定する際に、デバッグコンテナーの root ディレクトリー (**/host** を含む) から絶対パスを参照します。

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。**scp** を使用してクラスターノードから **tcpdump** キャプチャーファイルを転送することは推奨されず、ノードには **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この状態では、**scp core@<node>.<cluster_name>.<base_domain>:<file_path> <local_path>** を実行して、ノードから **tcpdump** キャプチャーファイルをコピーすることができます。

b. https://access.redhat.com/support/cases/ 内の既存のサポートケースに移動します。

c. **Attach files** を選択し、プロンプトに従ってファイルをアップロードします。

3.10. RED HAT サポートへの診断データの提供

OpenShift Container Platform の問題を調査する際に、Red Hat サポートは診断データをサポートケースにアップロードするよう依頼する可能性があります。ファイルは、Red Hat カスタマーポータルからサポートケースにアップロードするか、または **redhat-support-tool** コマンドを使用して OpenShift Container Platform クラスターから直接アップロードできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- ホストへの SSH アクセスがあること。
- OpenShift CLI (**oc**) がインストールされている。
- Red Hat の標準またはプレミアムサブスクリプションがある。
- Red Hat カスタマーポータルのアカウントがある。
- 既存の Red Hat サポートケース ID がある。

手順

- Red Hat カスタマーポータルから既存の Red Hat サポートケースに診断データをアップロードします。
  1. **oc debug node/<node_name>** コマンドを使用して OpenShift Container Platform ノードで組み込まれている診断ファイルを連結し、出力をファイルにリダイレクトします。以下の例では、**/host/var/tmp/my-diagnostic-data.tar.gz** をデバッグコンテナーから

/var/tmp/my-diagnostic-data.tar.gz にコピーします。

```
$ oc debug node/my-cluster-node -- bash -c 'cat /host/var/tmp/my-diagnostic-data.tar.gz'
> /var/tmp/my-diagnostic-data.tar.gz ①
```

① デバッグコンテナーは、ホストの root ディレクトリーを **/host** にマウントします。連結のためにターゲットファイルを指定する際に、デバッグコンテナーの root ディレクトリー (**/host** を含む) から絶対パスを参照します。

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。**scp** を使用してクラスターノードからファイルを転送することは推奨されず、ノードには **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この状態では、**scp core@<node>.<cluster_name>.<base_domain>:<file_path> <local_path>** を実行してノードから診断ファイルをコピーすることができます。

2. https://access.redhat.com/support/cases/ 内の既存のサポートケースに移動します。
3. **Attach files** を選択し、プロンプトに従ってファイルをアップロードします。

- OpenShift Container Platform クラスターから直接診断データを既存の Red Hat サポートケースにアップロードします。
  1. クラスターノードの一覧を取得します。

```
$ oc get nodes
```

  2. ターゲットノードのデバッグセッションに入ります。この手順は、**<node_name>-debug** というデバッグ Pod をインスタンス化します。

```
$ oc debug node/my-cluster-node
```

  3. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

```
# chroot /host
```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

4. **redhat-support-tool** を実行するために必要なバイナリーを含む **toolbox** コンテナーを起動します。

```
# toolbox
```

注記

既存の **toolbox** Pod がすでに実行されている場合、**toolbox** コマンドは以下を出力します: **'toolbox-' already exists.Trying to start....**問題が発生するのを回避するには、**podman rm toolbox-** で実行中の toolbox コンテナーを削除し、新規の toolbox コンテナーを生成します。

   a. **redhat-support-tool** を実行して、直接デバッグ Pod から既存の Red Hat サポートケースにファイルを添付します。この例では、サポートケース ID '01234567' とサンプルのファイルパス **/host/var/tmp/my-diagnostic-data.tar.gz** を使用します。

```
# redhat-support-tool addattachment -c 01234567 /host/var/tmp/my-diagnostic-data.tar.gz ①
```

① toolbox コンテナーは、ホストの root ディレクトリーを **/host** にマウントします。**redhat-support-tool** コマンドでアップロードするファイルを指定する場合は、toolbox コンテナーの root ディレクトリー (**/host/** を含む) から絶対パスを参照します。

第4章 クラスター仕様の要約

4.1. CLUSTERVERSIONによるクラスター仕様の要約

clusterversion リソースをクエリーすることにより、OpenShift Container Platform クラスター仕様の要約を取得できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. クラスターバージョン、可用性、アップタイム、および一般的なステータスをクエリーします。

   ```
   $ oc get clusterversion
   ```

2. クラスター仕様の詳細な要約、更新の可用性、および更新履歴を取得します。

   ```
   $ oc describe clusterversion
   ```

第5章 トラブルシューティング

5.1. インストールのトラブルシューティング

5.1.1. インストールの問題が発生する場所の判別

OpenShift Container Platform のインストールの問題のトラブルシューティング時に、インストールログを監視して、問題が発生した段階を判別できます。次に、その段階に関連する診断データを取得します。

OpenShift Container Platform インストールは以下の段階に従って実行されます。

1. Ignition 設定ファイルが作成されます。
2. ブートストラップマシンが起動し、マスターマシンの起動に必要なリモートリソースのホスティングを開始します。
3. マスターマシンは、ブートストラップマシンからリモートリソースをフェッチし、起動を終了します。
4. マスターマシンはブートストラップマシンを使用して、etcd クラスターを作成します。
5. ブートストラップマシンは、新規 etcd クラスターを使用して一時的な Kubernetes コントロールプレーンを起動します。
6. 一時的なコントロールプレーンは、マスターマシンに対して実稼働コントロールプレーンをスケジュールします。
7. 一時的なコントロールプレーンはシャットダウンし、コントロールを実稼働コントロールプレーンに渡します。
8. ブートストラップマシンは OpenShift Container Platform コンポーネントを実稼働コントロールプレーンに追加します。
9. インストールプログラムはブートストラップマシンをシャットダウンします。
10. コントロールプレーンはワーカーノードをセットアップします。
11. コントロールプレーンは一連の Operator の形式で追加のサービスをインストールします。
12. クラスターはサポートされる環境でのワーカーマシンの作成など、日常の操作に必要な残りのコンポーネントをダウンロードし、設定します。

5.1.2. ユーザーによってプロビジョニングされるインフラストラクチャーのインストールに関する考慮事項

デフォルトのインストール方法は、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です。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クラスターの場合、OpenShift Container Platform は、オペレーティングシステム自体を含むクラスターのすべての側面を管理します。可能な場合は、この機能を使用してクラスターインフラストラクチャーのプロビジョニングと保守の手間を省くようにしてください。

OpenShift Container Platform 4.5 はユーザーが独自にプロビジョニングするインフラストラクチャーにインストールすることもできます。このインストール方法を使用する場合は、ユーザーによってプロビジョニングされるインフラストラクチャーのインストールドキュメントに注意深く従ってください。

また、インストール前に以下の考慮事項を確認してください。

- Red Hat Enterprise Linux (RHEL) Ecosystem を確認し、選択したサーバーハードウェアまたは仮想化テクノロジー向けに提供されている Red Hat Enterprise Linux CoreOS (RHCOS) サポートのレベルを判別します。
- 多くの仮想化環境およびクラウド環境では、ゲストオペレーティングシステムにエージェントをインストールする必要があります。これらのエージェントがデーモンセット経由でデプロイされるコンテナー化されたワークロードとしてインストールされていることを確認します。
- 動的ストレージ、オンデマンドサービスルーティング、ノードホスト名の Kubernetes ホスト名への解決、クラスターの自動スケーリングなどの機能を有効にする場合は、クラウドプロバイダーの統合をインストールします。

注記

異なるクラウドプロバイダーのリソースを組み合わせた OpenShift Container Platform 環境でのクラウドプロバイダーの統合を有効にしたり、複数の物理または仮想プラットフォームにまたがるクラウドプロバイダーの統合を有効にすることはできません。ノードライフサイクルコントローラーでは、既存プロバイダーの外部にあるノードをクラスターに追加することはできず、複数のクラウドプロバイダーの統合を指定することはできません。

- マシンセットまたは自動スケーリングを使用して OpenShift Container Platform クラスターノードを自動的にプロビジョニングする必要がある場合、プロバイダー固有のマシン API 実装が必要です。
- 選択したクラウドプロバイダーが、初期デプロイメントの一部として Ignition 設定ファイルをホストに挿入する方法を提供するかどうかを確認します。提供しない場合は、HTTP サーバーを使用して Ignition 設定ファイルをホストする必要があります。Ignition 設定ファイルの問題のトラブルシューティングを行う手順は、これらの 2 つの方法のどちらをデプロイするかによって異なります。
- 組み込みコンテナーレジストリー、ElasticSearch、Prometheus などのオプションのフレームワークコンポーネントを利用する必要がある場合は、ストレージを手動でプロビジョニングする必要があります。デフォルトのストレージクラスは、明示的に設定されない限り、ユーザーによってプロビジョニングされるインフラストラクチャーのインストールでは定義されません。
- ロードバランサーは、可用性の高い OpenShift Container Platform 環境にあるすべてのマスターノードに API 要求を分散するために必要です。OpenShift Container Platform DNS ルーティングおよびポートの要件を満たす TCP ベースの負荷分散ソリューションを使用できます。

5.1.3. OpenShift Container Platform インストール前のロードバランサー設定の確認

OpenShift Container Platform インストールを開始する前に、ロードバランサーの設定を確認してください。

前提条件

- OpenShift Container Platform インストールの準備のために、選択した外部ロードバランサーを設定している。以下の例では、HAProxy を使用した Red Hat Enterprise Linux (RHEL) ホストに基づいて、負荷分散サービスをクラスターに提供します。
- OpenShift Container Platform インストールの準備のために DNS を設定している。

- ロードバランサーへの SSH アクセスがある。

手順

1. **haproxy** systemd サービスがアクティブであることを確認します。

   ```
   $ ssh <user_name>@<load_balancer> systemctl status haproxy
   ```

2. ロードバランサーが必要なポートでリッスンしていることを確認します。以下の例では、ポート **80**、**443**、**6443**、および **22623** を参照します。

   - Red Hat Enterprise Linux (RHEL) 6 で実行している HAProxy インスタンスの場合は、**netstat** コマンドを使用して、ポートのステータスを確認します。

     ```
     $ ssh <user_name>@<load_balancer> netstat -nltupe | grep -E ':80|:443|:6443|:22623'
     ```

   - Red Hat Enterprise Linux (RHEL) 7 または 8 で実行している HAProxy インスタンスの場合、**ss** コマンドを使用して、ポートのステータスを確認します。

     ```
     $ ssh <user_name>@<load_balancer> ss -nltupe | grep -E ':80|:443|:6443|:22623'
     ```

     注記

     Red Hat は、Red Hat Enterprise Linux (RHEL) 7 以降の **netstat** ではなく、**ss** コマンドを推奨しています。**ss** は、iproute パッケージで提供されます。**ss** コマンドの詳細は、『Red Hat Enterprise Linux (RHEL) 7 パフォーマンスチューニングガイド』を参照してください。

3. ワイルドカード DNS レコードがロードバランサーに解決されていることを確認します。

   ```
   $ dig <wildcard_fqdn> @<dns_server>
   ```

5.1.4. OpenShift Container Platform インストーラーのログレベルの指定

デフォルトで、OpenShift Container Platform インストーラーのログレベルは **info** に設定されます。失敗した OpenShift Container Platform インストールの診断時により詳細なロギングが必要な場合は、再びインストールを開始する際に **openshift-install** ログレベルを **debug** に引き上げることができます。

前提条件

- インストールホストにアクセスできる。

手順

- インストールを開始する際に、インストールのログレベルを **debug** に設定します。

  ```
  $ ./openshift-install --dir=<installation_directory> wait-for bootstrap-complete --log-
  level=debug ①
  ```

  ① ログレベルには、**info**、**warn**、**error**、 および **debug** が含まれます。

5.1.5. openshift-install コマンド関連の問題のトラブルシューティング

openshift-install コマンドの実行に問題がある場合には、以下を確認してください。

- インストールは Ignition 設定ファイルの作成から 24 時間以内に開始されている。Ignition ファイルは以下のコマンドの実行時に作成されている。

  ```
  $ ./openshift-install create ignition-configs --dir=./install_dir
  ```

- **install-config.yaml** ファイルはインストーラーと同じディレクトリーにある。代替インストールパスが **./openshift-install --dir** オプションを使用して宣言される場合、そのディレクトリーに **install-config.yaml** ファイルが存在することを確認します。

5.1.6. インストールの進捗の監視

OpenShift Container Platform インストールの進捗として、高レベルのインストール、ブートストラップ、およびコントロールプレーンのログをモニターできます。これにより、インストールの進捗をより明確に把握できるようになり、インストールが失敗する段階を特定しやすくなります。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。
- ブートストラップおよびマスターノードの完全修飾ドメイン名がある。

> **注記**
>
> 初期の **kubeadmin** パスワードは、インストールホストの **<install_directory>/auth/kubeadmin-password** にあります。

手順

1. インストールの進捗に応じてインストールログを監視します。

   ```
   $ tail -f ~/<installation_directory>/.openshift_install.log
   ```

2. 起動後にブートストラップノードで **bootkube.service** journald ユニットログを監視します。これにより、最初のコントロールプレーンのブートストラップを可視化できます。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

   ```
   $ ssh core@<bootstrap_fqdn> journalctl -b -f -u bootkube.service
   ```

> **注記**
>
> ブートストラップノードで **bootkube.service** のログは etcd の **connection refused** エラーを出力し、ブートストラップサーバーがマスターノードの etcd に接続できないことを示します。etcd が各マスターノードで起動し、ノードがクラスターに参加した後には、エラーは発生しなくなるはずです。

3. **kubelet.service** journald ユニットログを、起動後のマスターノードで監視します。これにより、マスターノードのエージェントアクティビティーを可視化できます。
    a. **oc** を使用してログを監視します。

    ```
    $ oc adm node-logs --role=master -u kubelet
    ```

    b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> journalctl -b -f -u
    kubelet.service
    ```

4. 起動後のマスターノードで **crio.service** journald ユニットログを監視します。これにより、マスターノードの CRI-O コンテナーランタイムのアクティビティーを可視化できます。
    a. **oc** を使用してログを監視します。

    ```
    $ oc adm node-logs --role=master -u crio
    ```

    b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

    ```
    $ ssh core@master-N.cluster_name.sub_domain.domain journalctl -b -f -u crio.service
    ```

5.1.7. ブートストラップノードの診断データの収集

ブートストラップ関連の問題が発生した場合、ブートストラップノードから **bootkube.service** の**journald** ユニットログおよびコンテナーログを収集できます。

前提条件

- ブートストラップノードへの SSH アクセスがある。
- ブートストラップノードの完全修飾ドメイン名がある。
- HTTP サーバーを使用して Ignition 設定ファイルをホストする場合、HTTP サーバーの完全修飾ドメイン名およびポート番号が必要です。HTTP ホストへの SSH アクセスも必要です。

手順

1. ブートストラップノードのコンソールにアクセスできる場合は、ノードがログインプロンプトに到達するまでコンソールを監視します。
2. Ignition ファイル設定を検証します。
    - HTTP サーバーを使用して Ignition 設定ファイルをホストする場合。
        a. ブートストラップノードの Ignition ファイル URL を確認します。**<http_server_fqdn>** を HTTP サーバーの完全修飾ドメイン名に置き換えます。

        ```
        $ curl -I http://<http_server_fqdn>:<port>/bootstrap.ign ①
        ```

① **-I** オプションはヘッダーのみを返します。Ignition ファイルが指定された URL で利用可能な場合、コマンドは **200 OK** ステータスを返します。これが利用できない場合は、コマンドは **404 file not found** を返します。

b. Ignition ファイルがブートストラップノードで受信されたことを確認するには、提供側ホストの HTTP サーバーログをクエリーします。たとえば、Apache Web サーバーを使用して Ignition ファイルを提供する場合は、以下のコマンドを入力します。

```
$ grep -is 'bootstrap.ign' /var/log/httpd/access_log
```

ブートストラップ Ignition ファイルが受信される場合、関連付けられた **HTTP GET** ログメッセージには要求が成功したことを示す **200 OK** の成功ステータスが含まれます。

c. Ignition ファイルが受信されていない場合には、Ignition ファイルが存在し、それらに提供側ホストの適切なファイルおよび Web サーバーパーミッションがあることを直接確認します。

- クラウドプロバイダーのメカニズムを使用して Ignition 設定ファイルを初期デプロイメントの一部としてホストに挿入する場合。

    a. ブートストラップノードのコンソールを確認し、ブートストラップノードの Ignition ファイルを正しく挿入するメカニズムが機能しているかどうかを確認します。

3. ブートストラップノードの割り当てられたストレージデバイスの可用性を確認します。

4. ブートストラップノードに DHCP サーバーから IP アドレスが割り当てられていることを確認します。

5. ブートストラップノードから **bootkube.service** journald ユニットログを収集します。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> journalctl -b -f -u bootkube.service
```

注記

ブートストラップノードで **bootkube.service** のログは etcd の **connection refused** エラーを出力し、ブートストラップサーバーがマスターノードの etcd に接続できないことを示します。etcd が各マスターノードで起動し、ノードがクラスターに参加した後には、エラーは発生しなくなるはずです。

6. ブートストラップノードコンテナーからログを収集します。

    a. ブートストラップノードで **podman** を使用してログを収集します。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> 'for pod in $(sudo podman ps -a -q); do sudo podman
logs $pod; done'
```

7. ブートストラッププロセスに失敗した場合は、以下を確認します。

- インストールホストから **api.<cluster_name>.<base_domain>** を解決できます。

- ロードバランサーはブートストラップおよびマスターノードへのポート 6443 接続をプロキシーします。プロキシー設定が OpenShift Container Platform のインストール要件を満たしていることを確認します。

5.1.8. マスターノードのインストール関連の問題の調査

マスターノードのインストールに問題がある場合には、マスターノード、OpenShift Container Platform ソフトウェア定義ネットワーク (SDN)、およびネットワーク Operator のステータスを判別します。マスターノードエージェント、CRI-O コンテナーランタイム、および Pod アクティビティーを可視化できるように **kubelet.service**、**crio.service** journald ユニットログ、およびマスターノードコンテナーログを収集します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。
- ブートストラップおよびマスターノードの完全修飾ドメイン名がある。
- HTTP サーバーを使用して Ignition 設定ファイルをホストする場合、HTTP サーバーの完全修飾ドメイン名およびポート番号が必要です。HTTP ホストへの SSH アクセスも必要です。

> **注記**
>
> 初期の **kubeadmin** パスワードは、インストールホストの **<install_directory>/auth/kubeadmin-password** にあります。

手順

1. マスターノードのコンソールにアクセスできる場合は、ノードがログインプロンプトに到達するまでコンソールを監視します。インストール時に、Ignition ログメッセージはコンソールに出力されます。
2. Ignition ファイル設定を確認します。
   - HTTP サーバーを使用して Ignition 設定ファイルをホストする場合。
     a. マスターノードの Ignition ファイル URL を確認します。**<http_server_fqdn>** を HTTP サーバーの完全修飾ドメイン名に置き換えます。

     ```
     $ curl -I http://<http_server_fqdn>:<port>/master.ign ①
     ```

     ① **-I** オプションはヘッダーのみを返します。Ignition ファイルが指定された URL で利用可能な場合、コマンドは **200 OK** ステータスを返します。これが利用できない場合は、コマンドは **404 file not found** を返します。

     b. Ignition ファイルがマスターノードで受信されたことを確認するには、提供側ホストの HTTP サーバーログをクエリーします。たとえば、Apache Web サーバーを使用して Ignition ファイルを提供する場合は、以下を考慮してください。

     ```
     $ grep -is 'master.ign' /var/log/httpd/access_log
     ```

    マスター Ignition ファイルが受信される場合、関連付けられた **HTTP GET** ログメッセージには要求が成功したことを示す **200 OK** の成功ステータスが含まれます。

    c. Ignition ファイルが受信されなかった場合、これが提供側ホストに存在することを直接確認します。適切なファイルおよび Web サーバーのパーミッションが適用されていることを確認します。

  - クラウドプロバイダーのメカニズムを使用して Ignition 設定ファイルを初期デプロイメントの一部としてホストに挿入する場合。

    a. マスターノードのコンソールを確認し、マスターノードの Ignition ファイルを正しく挿入するメカニズムが機能しているかどうかを確認します。

3. マスターノードの割り当てられたストレージデバイスの可用性を確認します。

4. マスターノードに DHCP サーバーから IP アドレスが割り当てられていることを確認します。

5. マスターノードのステータスを判別します。

   a. マスターノードのステータスをクエリーします。

   ```
   $ oc get nodes
   ```

   b. マスターノードのいずれかが **Ready** ステータスに達していない場合は、詳細なノードの説明を取得します。

   ```
   $ oc describe node <master_node>
   ```

   注記

   インストールの問題により OpenShift Container Platform API が実行できなくなったり、kubelet が各ノードでまだ実行されていない場合、**oc** コマンドを実行することはできません。

6. OpenShift Container Platform SDN のステータスを判別します。

   a. **openshift-sdn** namespace で、**sdn-controller**、**sdn**、および **ovs** デーモンセットのステータスを確認します。

   ```
   $ oc get daemonsets -n openshift-sdn
   ```

   b. これらのリソースが **Not found** として一覧表示されている場合には、**openshift-sdn** namespace の Pod を確認します。

   ```
   $ oc get pods -n openshift-sdn
   ```

   c. **openshift-sdn** namespace で失敗した OpenShift Container Platform SDN Pod に関連するログを確認します。

   ```
   $ oc logs <sdn_pod> -n openshift-sdn
   ```

7. クラスターのネットワーク設定のステータスを確認します。

   a. クラスターのネットワーク設定が存在するかどうかを確認します。

```
$ oc get network.config.openshift.io cluster -o yaml
```

b. インストーラーがネットワーク設定の作成に失敗した場合、Kubernetes マニフェストを再度生成し、メッセージの出力を確認します。

```
$ ./openshift-install create manifests
```

c. **openshift-network-operator** namespace で Pod のステータスを確認し、Cluster Network Operator (CNO) が実行されているかどうかを判別します。

```
$ oc get pods -n openshift-network-operator
```

d. **openshift-network-operator** namespace からネットワーク Operator Pod ログを収集します。

```
$ oc logs pod/<network_operator_pod_name> -n openshift-network-operator
```

8. **kubelet.service** journald ユニットログを、起動後のマスターノードで監視します。これにより、マスターノードのエージェントアクティビティーを可視化できます。

a. **oc** を使用してログを取得します。

```
$ oc adm node-logs --role=master -u kubelet
```

b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> journalctl -b -f -u
kubelet.service
```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

9. 起動後のマスターノードで **crio.service** journald ユニットログを取得します。これにより、マスターノードの CRI-O コンテナーランタイムのアクティビティーを可視化できます。

a. **oc** を使用してログを取得します。

```
$ oc adm node-logs --role=master -u crio
```

b. API が機能しない場合は、代わりに SSH を使用してログを確認します。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> journalctl -b -f -u
crio.service
```

10. マスターノードの **/var/log/** の下にある特定のサブディレクトリーからログを収集します。
    a. **/var/log/** サブディレクトリー内に含まれるログの一覧を取得します。以下の例では、すべてのマスターノードの **/var/log/openshift-apiserver/** にあるファイルを一覧表示します。

    ```
    $ oc adm node-logs --role=master --path=openshift-apiserver
    ```

    b. **/var/log/** サブディレクトリー内の特定ログを確認します。以下の例は、すべてのマスターノードから **/var/log/openshift-apiserver/audit.log** コンテンツを出力します。

    ```
    $ oc adm node-logs --role=master --path=openshift-apiserver/audit.log
    ```

    c. API が機能しない場合は、代わりに SSH を使用して各ノードのログを確認します。以下の例は、**/var/log/openshift-apiserver/audit.log** をベースとしています。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo tail -f
    /var/log/openshift-apiserver/audit.log
    ```

11. SSH を使用してマスターノードのコンテナーログを確認します。
    a. コンテナーを一覧表示します。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl ps -a
    ```

    b. **crictl** を使用してコンテナーのログを取得します。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl logs -f
    <container_id>
    ```

12. マスターノードの設定に問題がある場合には、MCO、MCO エンドポイント、および DNS レコードが機能していることを確認します。Machine Config Operator (MCO) は、インストール時にオペレーティングシステムの設定を管理します。システムクロックの精度と証明書の有効性も確認します。
    a. MCO エンドポイントが利用可能かどうかをテストします。**<cluster_name>** を適切な値に置き換えます。

    ```
    $ curl https://api-int.<cluster_name>:22623/config/master
    ```

    b. エンドポイントが応答しない場合は、ロードバランサーの設定を確認します。エンドポイントがポート 22623 で実行されるよう設定されていることを確認します。
    c. MCO エンドポイントの DNS レコードが設定され、ロードバランサーに対して解決していることを確認します。
        i. 定義された MCO エンドポイント名の DNS ルックアップを実行します。

        ```
        $ dig api-int.<cluster_name> @<dns_server>
        ```

      ii. ロードバランサーの割り当てられた MCO IP アドレスに対して逆引き参照を実行します。

```
$ dig -x <load_balancer_mco_ip_address> @<dns_server>
```

   d. MCO がブートストラップノードから直接機能していることを確認します。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> curl https://api-int.<cluster_name>:22623/config/master
```

   e. システムクロックは、ブートストラップ、マスター、およびワーカーノード間で同期される必要があります。各ノードのシステムクロックの参照時間と時刻同期の統計を確認します。

```
$ ssh core@<node>.<cluster_name>.<base_domain> chronyc tracking
```

   f. 証明書の有効性を確認します。

```
$ openssl s_client -connect api-int.<cluster_name>:22623 | openssl x509 -noout -text
```

5.1.9. etcd インストールの問題の調査

インストール時に etcd の問題が発生した場合には、etcd Pod のステータスを確認し、etcd Pod ログを収集できます。etcd DNS レコードを確認し、マスターノードで DNS の可用性を確認することもできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。
- マスターノードの完全修飾ドメイン名がある。

手順

1. etcd Pod のステータスを確認します。
   a. **openshift-etcd** namespace の Pod のステータスを確認します。

```
$ oc get pods -n openshift-etcd
```

   b. **openshift-etcd-operator** namespace の Pod のステータスを確認します。

```
$ oc get pods -n openshift-etcd-operator
```

2. 直前のコマンドで一覧表示される Pod のいずれかに **Running** または **Completed** ステータスが表示されない場合は、Pod の診断情報を収集します。
   a. Pod のイベントを確認します。

```
$ oc describe pod/<pod_name> -n <namespace>
```

b. Pod のログを検査します。

```
$ oc logs pod/<pod_name> -n <namespace>
```

c. Pod に複数のコンテナーがある場合、前述のコマンドでエラーが作成され、コンテナー名はエラーメッセージに指定されます。各コンテナーのログを検査します。

```
$ oc logs pod/<pod_name> -c <container_name> -n <namespace>
```

3. API が機能しない場合には、代わりに SSH を使用して各マスターノードで etcd Pod およびコンテナーログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

   a. 各マスターノードで etcd Pod を一覧表示します。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl pods --
name=etcd-
```

   b. **Ready** ステータスが表示されない Pod について、Pod のステータスの詳細を検査します。**<pod_id>** を前述のコマンドの出力に一覧表示されている Pod の ID に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl inspectp
<pod_id>
```

   c. Pod に関連するコンテナーを一覧表示します。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl ps | grep
'<pod_id>'
```

   d. **Ready** ステータスが表示されていないコンテナーの場合は、コンテナーのステータスの詳細を検査します。**<container_id>** を前述のコマンドの出力に一覧表示されているコンテナー ID に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl inspect
<container_id>
```

   e. **Ready** ステータスが表示されていないコンテナーのログを確認します。**<container_id>** を前述のコマンドの出力に一覧表示されているコンテナー ID に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl logs -f
<container_id>
```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

4. マスターノードからプライマリーおよびセカンダリー DNS サーバー接続を検証します。

5.1.10. マスターノードの kubelet および API サーバーの問題の調査

インストール時にマスターノードの kubelet および API サーバーの問題を調査するには、DNS、DHCP、およびロードバランサーの機能を確認してください。また、証明書の有効期限が切れていないことを確認します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。
- マスターノードの完全修飾ドメイン名がある。

手順

1. API サーバーの DNS レコードがマスターノードの kubelet を **https://api-int.<cluster_name>.<base_domain>:6443** にダイレクトすることを確認します。レコードがロードバランサーを参照することを確認します。
2. ロードバランサーのポート 6443 定義が各マスターノードを参照することを確認します。
3. DHCP によって固有のマスターノードのホスト名が指定されていることを確認します。
4. 各マスターノードで **kubelet.service** journald ユニットログを検査します。
    a. **oc** を使用してログを取得します。

    ```
    $ oc adm node-logs --role=master -u kubelet
    ```

    b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> journalctl -b -f -u
    kubelet.service
    ```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

5. マスターノードの kubelet ログで証明書の有効期限のメッセージの有無を確認します。
    a. **oc** を使用してログを取得します。

    ```
    $ oc adm node-logs --role=master -u kubelet | grep -is 'x509: certificate has expired'
    ```

    b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

    ```
    $ ssh core@<master-node>.<cluster_name>.<base_domain> journalctl -b -f -u
    kubelet.service  | grep -is 'x509: certificate has expired'
    ```

5.1.11. ワーカーノードのインストールに関連する問題の調査

ワーカーノードのインストールに問題がある場合には、ワーカーノードのステータスを確認できます。**kubelet.service**、**crio.service** journald ユニットログ、およびワーカーノードコンテナーログを収集し、ワーカーノードエージェント、CRI-O コンテナーランタイム、および Pod アクティビティーを可視化します。さらに、Ignition ファイルおよびマシン API Operator の機能を確認することもできます。ワーカーノードのインストール後の設定が失敗する場合は、Machine Config Operator (MCO) および DNS 機能を確認します。また、ブートストラップ、マスター、およびワーカーノード間のシステムクロックの同期を確認し、証明書を検証することもできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。
- ブートストラップおよびワーカーノードの完全修飾ドメイン名がある。
- HTTP サーバーを使用して Ignition 設定ファイルをホストする場合、HTTP サーバーの完全修飾ドメイン名およびポート番号が必要です。HTTP ホストへの SSH アクセスも必要です。

注記

初期の **kubeadmin** パスワードは、インストールホストの **<install_directory>/auth/kubeadmin-password** にあります。

手順

1. ワーカーノードのコンソールにアクセスできる場合は、ノードがログインプロンプトに到達するまでコンソールを監視します。インストール時に、Ignition ログメッセージはコンソールに出力されます。

2. Ignition ファイル設定を確認します。

   - HTTP サーバーを使用して Ignition 設定ファイルをホストする場合。

     a. ワーカーノードの Ignition ファイル URL を確認します。**<http_server_fqdn>** を HTTP サーバーの完全修飾ドメイン名に置き換えます。

     ```
     $ curl -I http://<http_server_fqdn>:<port>/worker.ign ①
     ```

     ① **-I** オプションはヘッダーのみを返します。Ignition ファイルが指定された URL で利用可能な場合、コマンドは **200 OK** ステータスを返します。これが利用できない場合は、コマンドは **404 file not found** を返します。

     b. Ignition ファイルがワーカーノードで受信されたことを確認するには、HTTP ホストの HTTP サーバーログをクエリーします。たとえば、Apache Web サーバーを使用して Ignition ファイルを提供する場合は、以下を考慮してください。

     ```
     $ grep -is 'worker.ign' /var/log/httpd/access_log
     ```

     ワーカー Ignition ファイルが受信される場合、関連付けられた **HTTP GET** ログメッセージには要求が成功したことを示す **200 OK** の成功ステータスが含まれます。

     c. Ignition ファイルが受信されなかった場合、これが提供側ホストに存在することを直接確認します。適切なファイルおよび Web サーバーのパーミッションが適用されていることを確認します。

   - クラウドプロバイダーのメカニズムを使用して Ignition 設定ファイルを初期デプロイメントの一部としてホストに挿入する場合。

     a. ワーカーノードのコンソールを確認し、ワーカーノードの Ignition ファイルを正しく挿入するメカニズムが機能しているかどうかを確認します。

3. ワーカーノードの割り当てられたストレージデバイスの可用性を確認します。

4. ワーカーノードに DHCP サーバーから IP アドレスが割り当てられていることを確認します。

5. ワーカーノードのステータスを判別します。

   a. ノードのステータスをクエリーします。

   ```
   $ oc get nodes
   ```

   b. **Ready** ステータスが表示されないワーカーノードの詳細なノードの説明を取得します。

   ```
   $ oc describe node <worker_node>
   ```

   注記

   インストールの問題により OpenShift Container Platform API が実行できなくなったり、kubelet が各ノードでまだ実行されていない場合、**oc** コマンドを実行することはできません。

6. マスターノードとは異なり、ワーカーノードは Machine API Operator を使用してデプロイされ、スケーリングされます。Machine API Operator のステータスを確認します。

   a. Machine API Operator Pod のステータスを確認します。

   ```
   $ oc get pods -n openshift-machine-api
   ```

   b. Machine API Operator Pod のステータスが **Ready** ではない場合は、Pod のイベントを詳細に作成します。

   ```
   $ oc describe pod/<machine_api_operator_pod_name> -n openshift-machine-api
   ```

   c. **machine-api-operator** コンテナーログを検査します。コンテナーは **machine-api-operator** Pod 内で実行されます。

   ```
   $ oc logs pod/<machine_api_operator_pod_name> -n openshift-machine-api -c machine-api-operator
   ```

   d. また、**kube-rbac-proxy** コンテナーログも検査します。コンテナーは **machine-api-operator** Pod 内でも実行されます。

   ```
   $ oc logs pod/<machine_api_operator_pod_name> -n openshift-machine-api -c kube-rbac-proxy
   ```

7. **kubelet.service** journald ユニットログを、起動後のワーカーノードでモニターします。これにより、ワーカーノードエージェントのアクティビティーを可視化できます。

   a. **oc** を使用してログを取得します。

   ```
   $ oc adm node-logs --role=worker -u kubelet
   ```

   b. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<worker-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

   ```
   $ ssh core@<worker-node>.<cluster_name>.<base_domain> journalctl -b -f -u kubelet.service
   ```

   注記

   Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

8. 起動後のワーカーノードで **crio.service** journald ユニットログを取得します。これにより、ワーカーノードの CRI-O コンテナーランタイムのアクティビティーを可視化できます。

   a. **oc** を使用してログを取得します。

      ```
      $ oc adm node-logs --role=worker -u crio
      ```

   b. API が機能しない場合は、代わりに SSH を使用してログを確認します。

      ```
      $ ssh core@<worker-node>.<cluster_name>.<base_domain> journalctl -b -f -u
      crio.service
      ```

9. ワーカーノードの **/var/log/** の下にある特定のサブディレクトリーからログを収集します。

   a. **/var/log/** サブディレクトリー内に含まれるログの一覧を取得します。以下の例は、すべてのワーカーノードの **/var/log/sssd/** にあるファイルを一覧表示します。

      ```
      $ oc adm node-logs --role=worker --path=sssd
      ```

   b. **/var/log/** サブディレクトリー内の特定ログを確認します。以下の例では、すべてのワーカーノードから **/var/log/sssd/audit.log** コンテンツを出力します。

      ```
      $ oc adm node-logs --role=worker --path=sssd/sssd.log
      ```

   c. API が機能しない場合は、代わりに SSH を使用して各ノードのログを確認します。以下の例は、**/var/log/sssd/sssd.log** をベースとしています。

      ```
      $ ssh core@<worker-node>.<cluster_name>.<base_domain> sudo tail -f
      /var/log/sssd/sssd.log
      ```

10. SSH を使用してワーカーノードのコンテナーログを確認します。

    a. コンテナーを一覧表示します。

       ```
       $ ssh core@<worker-node>.<cluster_name>.<base_domain> sudo crictl ps -a
       ```

    b. **crictl** を使用してコンテナーのログを取得します。

       ```
       $ ssh core@<worker-node>.<cluster_name>.<base_domain> sudo crictl logs -f
       <container_id>
       ```

11. ワーカーノードの設定に問題がある場合には、MCO、MCO エンドポイント、および DNS レコードが機能していることを確認します。Machine Config Operator (MCO) は、インストール時にオペレーティングシステムの設定を管理します。システムクロックの精度と証明書の有効性も確認します。

    a. MCO エンドポイントが利用可能かどうかをテストします。**<cluster_name>** を適切な値に置き換えます。

       ```
       $ curl https://api-int.<cluster_name>:22623/config/worker
       ```

    b. エンドポイントが応答しない場合は、ロードバランサーの設定を確認します。エンドポイントがポート 22623 で実行されるよう設定されていることを確認します。

    c. MCO エンドポイントの DNS レコードが設定され、ロードバランサーに対して解決していることを確認します。

i. 定義された MCO エンドポイント名の DNS ルックアップを実行します。

```
$ dig api-int.<cluster_name> @<dns_server>
```

ii. ロードバランサーの割り当てられた MCO IP アドレスに対して逆引き参照を実行します。

```
$ dig -x <load_balancer_mco_ip_address> @<dns_server>
```

d. MCO がブートストラップノードから直接機能していることを確認します。**<bootstrap_fqdn>** をブートストラップノードの完全修飾ドメイン名に置き換えます。

```
$ ssh core@<bootstrap_fqdn> curl https://api-int.<cluster_name>:22623/config/worker
```

e. システムクロックは、ブートストラップ、マスター、およびワーカーノード間で同期される必要があります。各ノードのシステムクロックの参照時間と時刻同期の統計を確認します。

```
$ ssh core@<node>.<cluster_name>.<base_domain> chronyc tracking
```

f. 証明書の有効性を確認します。

```
$ openssl s_client -connect api-int.<cluster_name>:22623 | openssl x509 -noout -text
```

5.1.12. インストール後の Operator ステータスのクエリー

インストールの終わりに Operator のステータスを確認できます。利用できない Operator の診断データを取得します。**Pending** と一覧表示されているか、またはエラーステータスのある Operator Pod のログを確認します。問題のある Pod によって使用されるベースイメージを検証します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. クラスター Operator がすべてインストールの終わりに利用可能な状態であることを確認します。

```
$ oc get clusteroperators
```

2. Operator が利用可能にならない場合は、Operator イベントを表示します。

```
$ oc describe clusteroperator <operator_name>
```

3. Operator の namespace 内で Operator Pod のステータスを確認します。

```
$ oc get pods -n <operator_namespace>
```

4. **Running** ステータスを持たない Pod についての詳細な説明を取得します。

```
$ oc describe pod/<operator_pod_name> -n <operator_namespace>
```

5. Pod ログを検査します。

```
$ oc logs pod/<operator_pod_name> -n <operator_namespace>
```

6. Pod ベースイメージに関連する問題が発生した場合には、ベースイメージのステータスを確認します。

   a. 問題のある Pod で使用されるベースイメージの詳細を取得します。

   ```
   $ oc get pod -o "jsonpath={range .status.containerStatuses[*]}{.name}{'\t'}{.state}{'\t'}
   {.image}{'\n'}{end}" <operator_pod_name> -n <operator_namespace>
   ```

   b. ベースイメージのリリース情報を一覧表示します。

   ```
   $ oc adm release info <image_path>:<tag> --commits
   ```

5.1.13. 失敗したインストールのログの収集

SSH キーをインストールプログラムに指定している場合、失敗したインストールについてのデータを収集することができます。

注記

実行中のクラスターからログを収集する場合とは異なるコマンドを使用して失敗したインストールについてのログを収集します。実行中のクラスターからログを収集する必要がある場合は、**oc adm must-gather** コマンドを使用します。

前提条件

- OpenShift Container Platform のインストールがブートストラッププロセスの終了前に失敗している。ブートストラップノードは実行中であり、SSH でアクセスできる。
- **ssh-agent** プロセスはコンピューター上でアクティブであり、**ssh-agent** プロセスとインストールプログラムの両方に同じ SSH キーを提供している。
- 独自にプロビジョニングしたインフラストラクチャーにクラスターのインストールを試行した場合には、ブートストラップおよびマスターノードの完全修飾ドメイン名がある。

手順

1. ブートストラップおよびコントロールプレーンマシンからインストールログを収集するために必要なコマンドを生成します。

   - 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使用している場合は、以下のコマンドを実行します。

   ```
   $ ./openshift-install gather bootstrap --dir=<installation_directory> 1
   ```

   ① **installation_directory** は、**./openshift-install create cluster** を実行した際に指定したディレクトリーです。このディレクトリーには、インストールプログラムが作成する OpenShift Container Platform 定義ファイルが含まれます。

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の場合、インストールプログラムは、ホスト名または IP アドレスを指定しなくてもよいようにクラスターについての情報を保存します。

- 独自にプロビジョニングしたインフラストラクチャーを使用している場合は、以下のコマンドを実行します。

```
$ ./openshift-install gather bootstrap --dir=<installation_directory> \ ①
  --bootstrap <bootstrap_address> \ ②
  --master <master_1_address> \ ③
  --master <master_2_address> \ ④
  --master <master_3_address>" ⑤
```

① **installation_directory** には、**./openshift-install create cluster** を実行した際に指定したのと同じディレクトリーを指定します。このディレクトリーには、インストールプログラムが作成する OpenShift Container Platform 定義ファイルが含まれます。

② **<bootstrap_address>** は、クラスターのブートストラップマシンの完全修飾ドメイン名または IP アドレスです。

③ ④ ⑤ クラスター内のそれぞれのコントロールプレーン、またはマスター、マシンについては、**<master_*_address>** をその完全修飾ドメイン名または IP アドレスに置き換えます。

注記

デフォルトクラスターには 3 つのコントロールプレーンマシンが含まれます。クラスターが使用する数にかかわらず、表示されるようにすべてのコントロールプレーンマシンを一覧表示します。

出力例

```
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "<installation_directory>/log-bundle-
<timestamp>.tar.gz"
```

インストールの失敗についての Red Hat サポートケースを作成する場合は、圧縮したログをケースに含めるようにしてください。

5.1.14. 追加リソース

- OpenShift Container Platform のインストールタイプおよびプロセスについての詳細は、「インストールプロセス」を参照してください。

5.2. ノードの正常性の確認

5.2.1. ノードのステータス、リソースの使用状況および設定の確認

クラスターノードの正常性ステータス、リソース消費統計およびノードログを確認します。さらに、個別のノードで **kubelet** ステータスをクエリーします。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. クラスターのすべてのノードの名前、ステータスおよびロールを一覧表示します。

   ```
   $ oc get nodes
   ```

2. クラスター内の各ノードの CPU およびメモリーの使用状況を要約します。

   ```
   $ oc adm top nodes
   ```

3. 特定のノードの CPU およびメモリーの使用状況を要約します。

   ```
   $ oc adm top node -l my-node
   ```

5.2.2. ノードにおける kubelet ステータスのクエリー

クラスターノードの正常性ステータス、リソース消費統計およびノードログを確認できます。さらに、個別のノードで **kubelet** ステータスをクエリーできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. kubelet は各ノードの systemd サービスを使用して管理されます。デバッグ Pod 内で **kubelet** systemd サービスをクエリーし、kubelet のステータスを確認します。
   a. ノードのデバッグ Pod を起動します。

      ```
      $ oc debug node/my-node
      ```

   b. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

      ```
      # chroot /host
      ```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、**kubelet** がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

c. **kubelet** systemd サービスがノードでアクティブかどうかを確認します。

```
# systemctl is-active kubelet
```

d. より詳細な **kubelet.service** ステータスの要約を出力します。

```
# systemctl status kubelet
```

5.2.3. クラスターノードジャーナルログのクエリー

個別のクラスターノードの **/var/log** 内で **journald** ユニットログおよびその他のログを収集できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。
- ホストへの SSH アクセスがあること。

手順

1. OpenShift Container Platform クラスターノードから **kubelet** の**journald** ユニットログをクエリーします。以下の例では、マスターノードのみがクエリーされます。

```
$ oc adm node-logs --role=master -u kubelet ①
```

① 他のユニットログをクエリーするために、**kubelet** を適宜置き換えます。

2. クラスターノードの **/var/log/** の下にある特定のサブディレクトリーからログを収集します。

   a. **/var/log/** サブディレクトリー内に含まれるログの一覧を取得します。以下の例では、すべてのマスターノードの **/var/log/openshift-apiserver/** にあるファイルを一覧表示します。

```
$ oc adm node-logs --role=master --path=openshift-apiserver
```

   b. **/var/log/** サブディレクトリー内の特定ログを確認します。以下の例は、すべてのマスターノードから **/var/log/openshift-apiserver/audit.log** コンテンツを出力します。

```
$ oc adm node-logs --role=master --path=openshift-apiserver/audit.log
```

c. API が機能しない場合は、代わりに SSH を使用して各ノードのログを確認します。以下の例は、**/var/log/openshift-apiserver/audit.log** をベースとしています。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo tail -f
/var/log/openshift-apiserver/audit.log
```

> **注記**
>
> Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

5.3. CRI-O コンテナーランタイムの問題のトラブルシューティング

5.3.1. CRI-O コンテナーランタイムエンジンについて

CRI-O は Kubernetes ネイティブコンテナーランタイム実装です。これはオペレーティングシステムに密接に統合し、Kubernetes の効率的で最適化されたエクスペリエンスを提供します。CRI-O は、コンテナーを実行、停止および再起動を実行するための機能を提供します。

CRI-O コンテナーランタイムエンジンは、各 OpenShift Container Platform クラスターノードで systemd サービスを使用して管理されます。コンテナーランタイムの問題が発生する場合は、各ノードの **crio** systemd サービスのステータスを確認します。マニフェストコンテナーランタイムの問題のあるノードから CRI-O の journald ユニットログを収集します。

5.3.2. CRI-O ランタイムエンジンのステータスの確認

各クラスターノードで CRI-O コンテナーランタイムエンジンのステータスを確認できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. デバッグ Pod 内で、ノードの **crio** systemd サービスをクエリーして CRI-O ステータスを確認します。
   a. ノードのデバッグ Pod を起動します。

```
$ oc debug node/my-node
```

b. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

```
# chroot /host
```

> **注記**
>
> Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

c. **crio** systemd サービスがノードでアクティブかどうかを確認します。

```
# systemctl is-active crio
```

d. より詳細な **kubelet.service** ステータスの要約を出力します。

```
# systemctl status crio
```

5.3.3. CRI-O の journald ユニットログの収集

CRI-O の問題が発生した場合には、ノードから CRI-O journald ユニットログを取得できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。
- コントロールプレーンまたはマスターマシンの完全修飾ドメイン名がある。

手順

1. CRI-O journald ユニットログを収集します。以下の例は、クラスター内のすべてのマスターノードからログを収集します。

```
$ oc adm node-logs --role=master -u crio
```

2. 特定のノードから CRI-O journald ユニットログを収集します。

```
$ oc adm node-logs <node_name> -u crio
```

3. API が機能しない場合は、代わりに SSH を使用してログを確認します。**<node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

```
$ ssh core@<node>.<cluster_name>.<base_domain> journalctl -b -f -u crio.service
```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

5.4. OPERATOR 関連の問題のトラブルシューティング

Operator は、OpenShift Container Platform アプリケーションをパッケージ化し、デプロイし、管理する方法です。Operator はソフトウェアベンダーのエンジニアリングチームの拡張機能のように動作し、OpenShift Container Platform 環境を監視し、その最新状態に基づいてリアルタイムの意思決定を行います。Operator はアップグレードをシームレスに実行し、障害に自動的に対応するように設計されており、時間の節約のためにソフトウェアのバックアッププロセスを省略するなどのショートカットを実行することはありません。

OpenShift Container Platform 4.5 には、クラスターの正常な機能に必要なデフォルトの Operator セットが含まれます。これらのデフォルト Operator は Cluster Version Operator (CVO) によって管理されます。

クラスター管理者は、OpenShift Container Platform Web コンソールまたは CLI を使用して OperatorHub からアプリケーション Operator をインストールできます。その後、Operator を 1 つまたは複数の namespace にサブスクライブし、クラスター上で開発者が使用できるようにできます。アプリケーション Operator は Operator Lifecycle Manager (OLM) によって管理されます。

Operator に問題が発生した場合には、Operator Subscription のステータスを確認します。クラスター全体で Operator Pod の正常性を確認し、診断用に Operator ログを収集します。

5.4.1. Operator サブスクリプションの状態のタイプ

サブスクリプションは状態についての以下のタイプを報告します。

表5.1 サブスクリプションの状態のタイプ

状態	説明
CatalogSourcesUnhealthy	解決に使用される一部のまたはすべてのカタログソースは正常ではありません。
InstallPlanMissing	サブスクリプションのインストール計画がありません。
InstallPlanPending	サブスクリプションのインストール計画はインストールの保留中です。
InstallPlanFailed	サブスクリプションのインストール計画が失敗しました。

注記

デフォルトの OpenShift Container Platform クラスター Operator は Cluster Version Operator (CVO) によって管理され、これらの Operator には **Subscription** オブジェクトがありません。アプリケーション Operator は Operator Lifecycle Manager (OLM) によって管理され、それらには **Subscription** オブジェクトがあります。

5.4.2. CLI を使用した Operator サブスクリプションステータスの表示

CLI を使用して Operator サブスクリプションステータスを表示できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. Operator サブスクリプションを一覧表示します。

   ```
   $ oc get subs -n <operator_namespace>
   ```

2. **oc describe** コマンドを使用して、**Subscription** リソースを検査します。

   ```
   $ oc describe sub <subscription_name> -n <operator_namespace>
   ```

3. コマンド出力で、**Conditions** セクションで Operator サブスクリプションの状態タイプのステータスを確認します。以下の例では、利用可能なすべてのカタログソースが正常であるため、**CatalogSourcesUnhealthy** 状態タイプのステータスは **false** になります。

   出力例

   ```
   Conditions:
     Last Transition Time:  2019-07-29T13:42:57Z
     Message:               all available catalogsources are healthy
     Reason:                AllCatalogSourcesHealthy
     Status:                False
     Type:                  CatalogSourcesUnhealthy
   ```

注記

デフォルトの OpenShift Container Platform クラスター Operator は Cluster Version Operator (CVO) によって管理され、これらの Operator には **Subscription** オブジェクトがありません。アプリケーション Operator は Operator Lifecycle Manager (OLM) によって管理され、それらには **Subscription** オブジェクトはありません。

5.4.3. Operator Pod ステータスのクエリー

クラスター内の Operator Pod およびそれらのステータスを一覧表示できます。詳細な Operator Pod の要約を収集することもできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. クラスターで実行されている Operator を一覧表示します。出力には、Operator バージョン、可用性、およびアップタイムの情報が含まれます。

   ```
   $ oc get clusteroperators
   ```

2. Operator の namespace で実行されている Operator Pod を一覧表示し、Pod のステータス、再起動、および経過時間を一覧表示します。

   ```
   $ oc get pod -n <operator_namespace>
   ```

3. 詳細な Operator Pod の要約を出力します。

   ```
   $ oc describe pod <operator_pod_name> -n <operator_namespace>
   ```

4. Operator の問題がノード固有の問題である場合、そのノードで Operator コンテナーのステータスをクエリーします。

   a. ノードのデバッグ Pod を起動します。

      ```
      $ oc debug node/my-node
      ```

   b. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

      ```
      # chroot /host
      ```

      注記

      Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

   c. 状態および関連付けられた Pod ID を含む、ノードのコンテナーについての詳細を一覧表示します。

      ```
      # crictl ps
      ```

   d. ノード上の特定の Operator コンテナーについての情報を一覧表示します。以下の例では、**network-operator** コンテナーに関する情報を一覧表示します。

```
# crictl ps --name network-operator
```

e. デバッグシェルを終了します。

5.4.4. Operator ログの収集

Operator の問題が発生した場合、Operator Pod ログから詳細な診断情報を収集できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。
- コントロールプレーンまたはマスターマシンの完全修飾ドメイン名がある。

手順

1. Operator の namespace で実行されている Operator Pod、Pod のステータス、再起動、および経過時間を一覧表示します。

   ```
   $ oc get pods -n <operator_namespace>
   ```

2. Operator Pod のログを確認します。

   ```
   $ oc logs pod/<pod_name> -n <operator_namespace>
   ```

   Operator Pod に複数のコンテナーがある場合、前述のコマンドにより各コンテナーの名前が含まれるエラーが生成されます。個別のコンテナーからログをクエリーします。

   ```
   $ oc logs pod/<operator_pod_name> -c <container_name> -n <operator_namespace>
   ```

3. API が機能しない場合には、代わりに SSH を使用して各マスターノードで Operator Pod およびコンテナーログを確認します。**<master-node>.<cluster_name>.<base_domain>** を適切な値に置き換えます。

   a. 各マスターノードの Pod を一覧表示します。

   ```
   $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl pods
   ```

   b. Operator Pod で **Ready** ステータスが表示されない場合は、Pod のステータスを詳細に検査します。**<operator_pod_id>** を直前のコマンドの出力に一覧表示されている Operator Pod の ID に置き換えます。

   ```
   $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl inspectp
   <operator_pod_id>
   ```

   c. Operator Pod に関連するコンテナーを一覧表示します。

   ```
   $ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl ps --pod=
   <operator_pod_id>
   ```

d. **Ready** ステータスが Operator コンテナーに表示されない場合は、コンテナーのステータスを詳細に検査します。**<container_id>** を前述のコマンドの出力に一覧表示されているコンテナー ID に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl inspect
<container_id>
```

e. **Ready** ステータスが表示されない Operator コンテナーのログを確認します。**<container_id>** を前述のコマンドの出力に一覧表示されているコンテナー ID に置き換えます。

```
$ ssh core@<master-node>.<cluster_name>.<base_domain> sudo crictl logs -f
<container_id>
```

> **注記**
>
> Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。SSH 経由で診断データの収集を試行する前に、**oc adm must gather** およびその他の **oc** コマンドを実行して収集されるデータが十分であるかどうかを確認してください。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

5.4.5. Machine Config Operator の自動再起動の無効化

設定変更が Machine Config Operator によって行われる場合、Red Hat Enterprise Linux CoreOS (RHCOS) を再起動して変更を反映する必要があります。設定変更が **kube-apiserver-to-kubelet-signer** CA がローテーションされる場合などのように自動であるか、またはレジストリーまたは SSH キーが更新された場合などのように手動で実行されるかどうかにかかわらず、RHCOS ノードは一時停止されない限り自動的に再起動します。

不要な中断を防ぐために、マシン設定プール (MCP) を変更して、Operator がマシン設定を変更した後に自動再起動を防ぐことができます。

> **注記**
>
> マシン設定プールを停止すると、すべてのシステム再起動プロセスが一時停止し、すべての設定の変更が適用されなくなります。

5.4.5.1. コンソールの使用による Machine Config Operator の自動再起動の無効化

Machine Config Operator (MCO) の変更から不要な中断を防ぐには、OpenShift Container Platform Web コンソールを使用してマシン設定プール (MCP) を変更し、MCO がそのプール内のノードに変更を加えられないようにすることができます。これにより、通常 MCO 更新プロセスの一部として実行される再起動ができなくなります。

注記

MCP を一時停止にすることにより、オペレーティングシステム、セキュリティー、証明書およびマシン設定に関連するその他の更新を含む、RHCOS ノードへのすべての更新が停止します。一時停止は短期間にのみ実行する必要があります。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

自動 MCO 更新の再起動の一時停止または一時停止を解除するには、以下を実行します。

- 自動再起動プロセスを一時停止します。
  1. **cluster-admin** ロールを持つユーザーとして OpenShift Container Platform Web コンソールにログインします。
  2. **Compute** → **Machine Config Pools** をクリックします。
  3. **Machine Config Pools** ページで、再起動を一時停止するノードに応じて **master** または **worker** のいずれかをクリックします。
  4. **master** または **worker** ページで、**YAML** をクリックします。
  5. YAML で、**spec.paused** フィールドを **true** に更新します。

     MachineConfigPool オブジェクトのサンプル

     ```
     apiVersion: machineconfiguration.openshift.io/v1
     kind: MachineConfigPool
      ...
     spec:
      ...
       paused: true 1
     ```

     ① **spec.paused** フィールドを **true** に更新し、再起動を一時停止します。

  6. MCP が一時停止されていることを確認するには、**Machine Config Pools** ページに戻ります。
     Machine Config Pools ページで、**Paused** 列では、変更した MCP について **True** が報告されます。

     MCP が一時停止中に保留中の変更がある場合は、**Updated** 列は **False** であり、**Updating** は **False** になります。**Updated** が **True** であり、**Updating** が **False** の場合、保留中の変更はありません。

     重要

     保留中の変更がある場合 (**Updated** および **Updating** 列の両方が **False** の場合)、できるだけ早期に再起動のメンテナンス期間をスケジュールすることが推奨されます。自動再起動プロセスの一時停止を解除して、最後に再起動してからキューに追加された変更を適用するには、以下の手順に従います。

- 自動再起動プロセスの一時停止を解除するには、以下を実行します。

  1. **cluster-admin** ロールを持つユーザーとして OpenShift Container Platform Web コンソールにログインします。
  2. **Compute** → **Machine Config Pools** をクリックします。
  3. **Machine Config Pools** ページで、再起動を一時停止するノードに応じて **master** または **worker** のいずれかをクリックします。
  4. **master** または **worker** ページで、**YAML** をクリックします。
  5. YAML で、**spec.paused** フィールドを **false** に更新します。

     MachineConfigPool オブジェクトのサンプル

     ```
     apiVersion: machineconfiguration.openshift.io/v1
     kind: MachineConfigPool
      ...
     spec:
      ...
       paused: false ①
     ```

     ① **spec.paused** フィールドを **false** に更新し、再起動を許可します。

     注記

     MCP の一時停止を解除すると、MCO は一時停止したすべての変更を適用し、必要に応じて Red Hat Enterprise Linux CoreOS (RHCOS) を再起動します。

  6. MCP が一時停止されていることを確認するには、**Machine Config Pools** ページに戻ります。
     Machine Config Pools ページで、**Paused** 列では、変更した MCP について **False** が報告されます。

     MCP が保留中の変更を適用する場合、 **Updated** 列は **False** になり、**Updating** 列は **True** になります。**Updated** が **True** であり、**Updating** が **False** の場合、追加の変更は加えられません。

5.4.5.2. CLI の使用による Machine Config Operator の自動再起動の無効化

Machine Config Operator (MCO) によって加えられる変更から生じる不要な中断を防ぐには、OpenShift CLI (oc) を使用してマシン設定プール (MCP) を変更し、MCO がそのプール内のノードに変更を加えられないようにすることができます。これにより、通常 MCO 更新プロセスの一部として実行される再起動ができなくなります。

注記

MCP を一時停止にすることにより、オペレーティングシステム、セキュリティー、証明書およびマシン設定に関連するその他の更新を含む、RHCOS ノードへのすべての更新が停止します。一時停止は短期間にのみ実行する必要があります。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

自動 MCO 更新の再起動の一時停止または一時停止を解除するには、以下を実行します。

- 自動再起動プロセスを一時停止します。
  1. **MachineConfigPool** カスタムリソースを、**spec.paused** フィールドを **true** に設定するように更新します。

     コントロールプレーン (マスター) ノード

     ```
     $ oc patch --type=merge --patch='{"spec":{"paused":true}}' machineconfigpool/master
     ```

     ワーカーノード

     ```
     $ oc patch --type=merge --patch='{"spec":{"paused":true}}' machineconfigpool/worker
     ```

  2. MCP が一時停止されていることを確認します。

     コントロールプレーン (マスター) ノード

     ```
     $ oc get machineconfigpool/master --template='{{.spec.paused}}'
     ```

     ワーカーノード

     ```
     $ oc get machineconfigpool/worker --template='{{.spec.paused}}'
     ```

     出力例

     ```
     true
     ```

     spec.paused フィールドは **true** であり、MCP は一時停止されます。

  3. MCP に保留中の変更があるかどうかを判別します。

     ```
     # oc get machineconfigpool
     ```

     出力例

     ```
     NAME     CONFIG                                             UPDATED   UPDATING
     master   rendered-master-33cf0a1254318755d7b48002c597bf91   True      False
     worker   rendered-worker-e405a5bdb0db1295acea08bcca33fa60   False     False
     ```

     UPDATED 列が **False** であり、**UPDATING** が **False** の場合は、保留中の変更があります。**UPDATED** が **True** であり、**UPDATING** が **False** の場合、保留中の変更はありません。この例では、ワーカーノードに保留中の変更があります。マスターノードには保留中の変更がありません。

重要

保留中の変更がある場合 (**Updated** および **Updating** 列の両方が **False** の場合)、できるだけ早期に再起動のメンテナンス期間をスケジュールすることが推奨されます。自動再起動プロセスの一時停止を解除して、最後に再起動してからキューに追加された変更を適用するには、以下の手順に従います。

- 自動再起動プロセスの一時停止を解除するには、以下を実行します。

  1. **MachineConfigPool** カスタムリソースを、**spec.paused** フィールドを **false** に設定するように更新します。

     コントロールプレーン (マスター) ノード

     ```
     $ oc patch --type=merge --patch='{"spec":{"paused":false}}' machineconfigpool/master
     ```

     ワーカーノード

     ```
     $ oc patch --type=merge --patch='{"spec":{"paused":false}}' machineconfigpool/worker
     ```

     注記

     MCP の一時停止を解除すると、MCO は一時停止したすべての変更を適用し、必要に応じて Red Hat Enterprise Linux CoreOS (RHCOS) を再起動します。

  2. MCP の一時停止が解除されていることを確認します。

     コントロールプレーン (マスター) ノード

     ```
     $ oc get machineconfigpool/master --template='{{.spec.paused}}'
     ```

     ワーカーノード

     ```
     $ oc get machineconfigpool/worker --template='{{.spec.paused}}'
     ```

     出力例

     ```
     false
     ```

     spec.paused フィールドは **false** であり、マシン設定プールの一時停止は解除されます。

  3. MCP に保留中の変更があるかどうかを判別します。

     ```
     $ oc get machineconfigpool
     ```

     出力例

     ```
     NAME     CONFIG                                     UPDATED  UPDATING
     master   rendered-master-546383f80705bd5aeaba93   True     False
     worker   rendered-worker-b4c51bb33ccaae6fc4a6a5   False    True
     ```

MCP が保留中の変更を適用する場合、**UPDATED** 列は **False** で、**UPDATING** 列は **True** になります。**UPDATED** が **True** であり、**UPDATING** が **False** の場合、追加の変更は加えられません。直前の例では、MCO はワーカーノードを更新しています。

5.5. POD の問題の調査

OpenShift Container Platform は、ホスト上に共にデプロイされる 1 つ以上のコンテナーである Pod の Kubernetes の概念を活用しています。Pod は、OpenShift Container Platform 4.5 で定義され、デプロイされ、管理される最小のコンピュート単位です。

Pod が定義されると、コンテナーが終了するまで、またはコンテナーが削除されるまでノードで実行されるように割り当てられます。ポリシーおよび終了コードに応じて、Pod は終了または保持後に削除され、それらのログがアクセスできるようにします。

Pod の問題が発生した場合には、まず Pod のステータスをチェックします。Pod の明示的な障害が発生した場合には、Pod のエラー状態をチェックして、特定のイメージ、コンテナー、または Pod ネットワークの問題を特定してください。エラー状態に基づく診断データの収集を行います。Pod イベントメッセージおよび Pod およびコンテナーのログ情報を確認します。コマンドライン上で実行中の Pod にアクセスするか、または問題のある Pod のデプロイメント設定に基づいて root アクセスでデバッグ Pod を起動して問題を動的に診断します。

5.5.1. Pod のエラー状態について

Pod の障害により、**oc get Pods** の出力の **status** フィールドで確認できる明示的なエラー状態が返されます。Pod のエラー状態は、イメージ、コンテナー、およびコンテナーネットワークに関連する障害についての状態を示します。

以下の表は、Pod のエラー状態の一覧をそれらの説明を記載しています。

表5.2 Pod のエラー状態

Pod のエラー状態	説明
ErrImagePull	一般的なイメージの取得エラー。
ErrImagePullBackOff	イメージの取得に失敗し、取り消されました。
ErrInvalidImageName	指定されたイメージ名は無効です。
ErrImageInspect	イメージの検査に失敗しました。
ErrImageNeverPull	**PullPolicy** は **NeverPullImage** に設定され、ターゲットイメージはホスト上でローカルに見つかりません。
ErrRegistryUnavailable	レジストリーからイメージの取得を試みる際に、HTTP エラーが発生しました。
ErrContainerNotFound	指定されたコンテナーが宣言された Pod 内にないか、または kubelet によって管理されていません。

Pod のエラー状態	説明
ErrRunInitContainer	コンテナーの初期化に失敗しました。
ErrRunContainer	Pod のコンテナーのいずれも正常に起動しませんでした。
ErrKillContainer	Pod のコンテナーのいずれも正常に強制終了されませんでした。
ErrCrashLoopBackOff	コンテナーが終了しました。kubelet は再起動を試行しません。
ErrVerifyNonRoot	コンテナーまたはイメージが root 権限で実行を試行しました。
ErrCreatePodSandbox	Pod サンドボックスの作成が成功しませんでした。
ErrConfigPodSandbox	Pod サンドボックス設定を取得できませんでした。
ErrKillPodSandbox	Pod サンドボックスは正常に停止しませんでした。
ErrSetupNetwork	ネットワークの初期化に失敗しました。
ErrTeardownNetwork	ネットワークの終了に失敗しました。

5.5.2. Pod ステータスの確認

Pod のステータスおよびエラー状態をクエリーできます。Pod に関連するデプロイメント設定をクエリーし、ベースイメージの可用性を確認することもできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされている。
- **skopeo** がインストールされている。

手順

1. プロジェクトに切り替えます。

```
$ oc project <project_name>
```

2. namespace 内で実行されている Pod、Pod のステータス、エラーの状態、再起動、および経過時間を一覧表示します。

   ```
   $ oc get pods
   ```

3. namespace がデプロイメント設定で管理されているかどうかを判別します。

   ```
   $ oc status
   ```

   namespace がデプロイメント設定で管理される場合、出力には、デプロイメント設定名とベースイメージの参照が含まれます。

4. 前述のコマンドの出力で参照されているベースイメージを検査します。

   ```
   $ skopeo inspect docker://<image_reference>
   ```

5. ベースイメージの参照が正しくない場合は、デプロイメント設定の参照を更新します。

   ```
   $ oc edit deployment/my-deployment
   ```

6. デプロイメント設定が終了時に変更されると、設定が自動的に再デプロイされます。デプロイメントの進行中に Pod ステータスを確認し、問題が解決されているかどうかを判別します。

   ```
   $ oc get pods -w
   ```

7. Pod の失敗に関連する診断情報については、namespace 内でイベントを確認します。

   ```
   $ oc get events
   ```

5.5.3. Pod およびコンテナーログの検査

明示的な Pod の失敗に関連する警告およびエラーメッセージの有無について Pod およびコンテナーログを検査できます。ポリシーおよび終了コードによっては、Pod およびコンテナーログは Pod の終了後も利用可能のままになります。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. 特定の Pod のログをクエリーします。

   ```
   $ oc logs <pod_name>
   ```

2. Pod 内の特定コンテナーのログをクエリーします。

   ```
   $ oc logs <pod_name> -c <container_name>
   ```

前述の **oc logs** コマンドを使用して取得されるログは、Pod またはコンテナー内の標準出力 (stdout) に送信されるメッセージで構成されます。

3. Pod 内の **/var/log**/ に含まれるログを検査します。
    a. Pod 内の **/var/log** に含まれるファイルおよびサブディレクトリーを一覧表示します。

    ```
    $ oc exec <pod_name> ls -alh /var/log
    ```

    b. Pod 内の **/var/log** に含まれる特定のログファイルをクエリーします。

    ```
    $ oc exec <pod_name> cat /var/log/<path_to_log>
    ```

    c. 特定のコンテナー内の **/var/log** に含まれるログファイルおよびサブディレクトリーを一覧表示します。

    ```
    $ oc exec <pod_name> -c <container_name> ls /var/log
    ```

    d. 特定のコンテナー内の **/var/log** に含まれる特定のログファイルをクエリーします。

    ```
    $ oc exec <pod_name> -c <container_name> cat /var/log/<path_to_log>
    ```

5.5.4. 実行中の Pod へのアクセス

Pod 内でシェルを開くか、またはポート転送によりネットワークアクセスを取得して、実行中の Pod を動的に確認することができます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. アクセスする Pod が含まれるプロジェクトに切り替えます。これは、**oc rsh** コマンドが **-n** namespace オプションを受け入れないために必要です。

    ```
    $ oc project <namespace>
    ```

2. リモートシェルを Pod で起動します。

    ```
    $ oc rsh <pod_name> ①
    ```

    ① Pod に複数のコンテナーがある場合、**oc rsh** は **-c <container_name>** が指定されていない限り最初のコンテナーにデフォルト設定されます。

3. Pod 内の特定のコンテナーでリモートシェルを起動します。

    ```
    $ oc rsh -c <container_name> pod/<pod_name>
    ```

4. Pod のポートへのポート転送セッションを作成します。

```
$ oc port-forward <pod_name> <host_port>:<pod_port> ①
```

① ポート転送セッションをキャンセルするには、**Ctrl+C** を入力します。

5.5.5. root アクセスでのデバッグ Pod の起動

問題のある Pod のデプロイメントまたはデプロイメント設定に基づいて、root アクセスでデバッグ Pod を起動できます。通常、Pod ユーザーは root 以外の権限で実行しますが、問題を調査するために一時的な root 権限で Pod のトラブルシューティングを実行することは役に立ちます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. デプロイメントに基づいて、root アクセスでデバッグ Pod を起動します。
   a. プロジェクトのデプロイメント名を取得します。

      ```
      $ oc get deployment -n <project_name>
      ```

   b. デプロイメントに基づいて、root 権限でデバッグ Pod を起動します。

      ```
      $ oc debug deployment/my-deployment --as-root -n <project_name>
      ```

2. デプロイメント設定に基づいて、root アクセスでデバッグ Pod を起動します。
   a. プロジェクトのデプロイメント設定名を取得します。

      ```
      $ oc get deploymentconfigs -n <project_name>
      ```

   b. デプロイメント設定に基づいて、root 権限でデバッグ Pod を起動します。

      ```
      $ oc debug deploymentconfig/my-deployment-configuration --as-root -n <project_name>
      ```

注記

インタラクティブなシェルを実行する代わりに、**-- <command>** を前述の **oc debug** コマンドに追加し、デバッグ Pod 内で個々のコマンドを実行することができます。

5.5.6. Pod およびコンテナーへの/からのファイルのコピー

Pod に/からファイルをコピーして、設定変更をテストしたり、診断情報を収集したりできます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. ファイルを Pod にコピーします。

```
$ oc cp <local_path> <pod_name>:/<path> -c <container_name> ①
```

① **-c** オプションが指定されていない場合、Pod の最初のコンテナーが選択されます。

2. Pod からファイルをコピーします。

```
$ oc cp <pod_name>:/<path>  -c <container_name><local_path> ①
```

① **-c** オプションが指定されていない場合、Pod の最初のコンテナーが選択されます。

注記

oc cp が機能するには、**tar** バイナリーがコンテナー内で利用可能である必要があります。

5.6. SOURCE-TO-IMAGE (S2I) プロセスのトラブルシューティング

5.6.1. Source-to-Image (S2I) のトラブルシューティングのストラテジー

Source-to-Image (S2I) を使用して、再現可能な Docker 形式のコンテナーイメージをビルドします。アプリケーションソースコードをコンテナーイメージに挿入し、新規イメージをアセンブルして実行可能なイメージを作成できます。新規イメージには、ベースイメージ (ビルダー) およびビルドされたソースが組み込まれています。

S2I プロセスで障害が発生する場所を特定するには、以下の各 S2I ステージに関連する Pod の状態を確認できます。

1. **ビルド設定の段階** で、ビルド Pod はベースイメージおよびアプリケーションのソースコードからアプリケーションコンテナーイメージを作成するために使用されます。
2. **デプロイメント設定の段階** で、デプロイメント Pod はビルド設定段階でビルドされたアプリケーションコンテナーイメージからアプリケーション Pod をデプロイするために使用されます。デプロイメント Pod は、サービスやルートなどの他のリソースもデプロイします。デプロイメント設定は、ビルド設定が成功すると開始されます。
3. **デプロイメント Pod のアプリケーション Pod の起動後** に、アプリケーションの障害が実行中のアプリケーション Pod 内で発生する可能性があります。たとえば、アプリケーション Pod が **Running** 状態であっても、アプリケーションは予想通りに動作しない可能性があります。このシナリオでは、実行中のアプリケーション Pod にアクセスして、Pod 内のアプリケーションの障害を調査できます。

S2I の問題のトラブルシューティングを行う際には、以下のストラテジーに従います。

1. ビルド、デプロイメント、およびアプリケーション Pod ステータスの監視
2. 問題が発生した S2I プロセスのステージの判別
3. 失敗したステージに対応するログの確認

5.6.2. Source-to-Image 診断データの収集

S2I ツールは、ビルド Pod とデプロイメント Pod を順番に実行します。デプロイメント Pod は、ビルドステージで作成されたアプリケーションコンテナーイメージに基づいてアプリケーション Pod をデプロイします。S2I プロセスで障害が発生する場所を判別するために、ビルド、デプロイメント、およびアプリケーション Pod のステータスを監視します。次に、これに応じて診断データを収集します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- API サービスが機能している。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. S2I プロセス全体での Pod のステータスを確認し、障害が発生するステージを判別します。

   ```
   $ oc get pods -w ①
   ```

   ① **-w** を使用して、**Ctrl+C** を使用してコマンドを終了するまで Pod で変更の有無を監視します。

2. 障害のある Pod のログでエラーの有無を確認します。
   - **ビルド Pod が失敗する場合**、ビルド Pod のログを確認します。

     ```
     $ oc logs -f pod/<application_name>-<build_number>-build
     ```

     注記

     または、**oc logs -f bc/<application_name>** を使用して、ビルド設定のログを確認できます。ビルド設定のログには、ビルド Pod からのログが含まれます。

   - **デプロイメント Pod が失敗する場合**、デプロイメント Pod のログを確認します。

     ```
     $ oc logs -f pod/<application_name>-<build_number>-deploy
     ```

> **注記**
>
> または、**oc logs -f dc/<application_name>** を使用して、デプロイメント設定のログを確認できます。これにより、デプロイメント Pod が正常に実行されるまで、デプロイメント Pod からログが出力されます。デプロイメント Pod の完了後に実行すると、コマンドはアプリケーション Pod からログを出力します。デプロイメント Pod の完了後も、**oc logs -f pod/<application_name>-<build_number>-deploy** を実行してログにアクセスできます。

- **アプリケーション Pod が失敗した場合や、アプリケーションが実行中のアプリケーション Pod 内で予想通りに動作しない場合**、アプリケーション Pod のログを確認します。

  ```
  $ oc logs -f pod/<application_name>-<build_number>-<random_string>
  ```

5.6.3. アプリケーションの障害を調査するためのアプリケーション診断データの収集

アプリケーションの障害は実行中のアプリケーション Pod 内で発生する可能性があります。このような状態では、以下のストラテジーを使用して診断情報を取得できます。

- アプリケーション Pod に関連するイベントを確認します。
- アプリケーション Pod からログを確認します。これには、OpenShift Container Platform のロギングフレームワークによって収集されないアプリケーション固有のログファイルが含まれます。
- アプリケーション機能を対話的にテストし、アプリケーションコンテナーで診断ツールを実行します。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. 特定のアプリケーション Pod に関連するイベントを一覧表示します。以下の例では、**my-app-1-akdlg** という名前のアプリケーション Pod のイベントを取得します。

   ```
   $ oc describe pod/my-app-1-akdlg
   ```

2. アプリケーション Pod からのログを確認します。

   ```
   $ oc logs -f pod/my-app-1-akdlg
   ```

3. 実行中のアプリケーション Pod 内で特定のログをクエリーします。標準出力(stdout) に送信されるログは OpenShift Container Platform のロギングフレームワークによって収集され、これは前述のコマンドの出力に含まれます。以下のクエリーは、標準出力 (stdout) に送信されないログにのみ必要です。

   a. Pod 内で root 権限なしにアプリケーションログにアクセスできる場合は、以下のようにログファイルを連結します。

```
$ oc exec my-app-1-akdlg -- cat /var/log/my-application.log
```

b. アプリケーションログの表示に root アクセスが必要な場合は、root 権限でデバッグコンテナーを起動し、コンテナー内でログファイルを表示できます。プロジェクトの **DeploymentConfig** オブジェクトからデバッグコンテナーを起動します。通常、Pod ユーザーは root 以外の権限で実行しますが、問題を調査するために一時的な root 権限で Pod のトラブルシューティングを実行することは役に立ちます。

```
$ oc debug dc/my-deployment-configuration --as-root -- cat /var/log/my-application.log
```

注記

oc debug dc/<deployment_configuration> --as-root を **-- <command>** を追加せずに実行する場合、デバッグ Pod 内で root アクセスでインタラクティブなシェルにアクセスできます。

4. インタラクティブなシェルでアプリケーション機能を対話的にテストし、アプリケーションコンテナーで診断ツールを実行します。

   a. アプリケーションコンテナーでインタラクティブなシェルを起動します。

   ```
   $ oc exec -it my-app-1-akdlg /bin/bash
   ```

   b. シェルからアプリケーションの機能を対話的にテストします。たとえば、コンテナーのエントリーポイントコマンドを実行して、結果を確認することができます。次に、ソースコードを更新し、S2I プロセスでアプリケーションコンテナーを再ビルドする前に、コマンドラインから直接に変更をテストします。

   c. コンテナー内で利用可能な診断バイナリーを実行します。

   注記

   一部の診断バイナリーを実行するには、root 権限が必要です。このような状況では、**oc debug dc/<deployment_configuration> --as-root** を実行して、問題のある Pod の **DeploymentConfig** オブジェクトに基づいて、root 権限でデバッグ Pod を起動できます。次に、デバッグ Pod 内から診断バイナリーを root として実行できます。

5. 診断バイナリーがコンテナー内で利用できない場合は、**nsenter** を使用して、コンテナーの namespace 内でホストの診断バイナリーを実行できます。以下の例では、ホストの **ip** バイナリーを使用して、コンテナーの namespace 内で **ip ad** を実行します。

   a. ターゲットノードのデバッグセッションに入ります。この手順は、**<node_name>-debug** というデバッグ Pod をインスタンス化します。

   ```
   $ oc debug node/my-cluster-node
   ```

   b. **/host** をデバッグシェル内の root ディレクトリーとして設定します。デバッグ Pod は、Pod 内の **/host** にホストの root ファイルシステムをマウントします。root ディレクトリーを **/host** に変更すると、ホストの実行パスに含まれるバイナリーを実行できます。

   ```
   # chroot /host
   ```

注記

Red Hat Enterprise Linux CoreOS (RHCOS) を実行する OpenShift Container Platform 4.5 クラスターノードは変更できず、Operator を使用してクラスターの変更を適用します。SSH を使用したクラスターノードへのアクセスは推奨されず、ノードは **accessed** のテイントのマークが付けられます。ただし、OpenShift Container Platform API が利用できない場合や、kubelet がターゲットノードで適切に機能しない場合、**oc** 操作がその影響を受けます。この場合は、代わりに **ssh core@<node>.<cluster_name>.<base_domain>** を使用してノードにアクセスできます。

c. ターゲットコンテナー ID を判別します。

```
# crictl ps
```

d. コンテナーのプロセス ID を確認します。この例では、ターゲットコンテナー ID は **a7fe32346b120** です。

```
# crictl inspect a7fe32346b120 --output yaml | grep 'pid:' | awk '{print $2}'
```

e. ホストの **ip** バイナリーを使用して、コンテナーの namespace 内で **ip ad** を実行します。この例では、**31150** をコンテナーのプロセス ID として使用します。**nsenter** コマンドは、ターゲットプロセスの namespace を入力し、その namespace でコマンドを実行します。この例のターゲットプロセスはコンテナーのプロセス ID であるため、**ip ad** コマンドは、ホストからコンテナーの namespace で実行されます。

```
# nsenter -n -t 31150 -- ip ad
```

注記

デバッグノードなどの特権付きコンテナーを使用している場合のみ、コンテナーの namespace 内でホストの診断バイナリーを実行できます。

5.6.4. 追加リソース

- S2I ビルドストラテジーの詳細は、「Source-to-Image (S2I) ビルド」を参照してください。

5.7. ストレージの問題のトラブルシューティング

5.7.1. 複数割り当てエラーの解決

ノードが予期せずにクラッシュまたはシャットダウンすると、割り当てられた ReadWriteOnce (RWO) ボリュームがノードからアンマウントされ、その後は別のノードでスケジュールされる Pod で使用可能になることが予想されます。

ただし、障害が発生したノードは割り当てられたボリュームをアンマウントできないため、新規ノードにマウントすることはできません。

複数割り当てのエラーが報告されます。

出力例

```
Unable to attach or mount volumes: unmounted volumes=[sso-mysql-pvol], unattached volumes=
[sso-mysql-pvol default-token-x4rzc]: timed out waiting for the condition
Multi-Attach error for volume "pvc-8837384d-69d7-40b2-b2e6-5df86943eef9" Volume is already used
by pod(s) sso-mysql-1-ns6b4
```

手順

複数割り当ての問題を解決するには、以下のソリューションのいずれかを使用します。

- RWX ボリュームを使用して、複数割り当てを有効にします。
  ほとんどのストレージソリューションでは、ReadWriteMany (RWX) ボリュームを使用して、複数割り当てエラーを防ぐことができます。

- RWO ボリュームの使用時に障害が発生したノードを回復するか、または削除します。
  VMware vSphere などの RWX をサポートしないストレージの場合、RWO ボリュームが代わりに使用される必要があります。ただし、RWO ボリュームは複数のノードにマウントできません。

  複数割り当てのエラーメッセージが RWO ボリュームと共に表示される場合には、シャットダウンまたはクラッシュしたノードで Pod を強制的に削除し、動的永続ボリュームの割り当て時などの重要なワークロードでのデータ損失を回避します。

  ```
  $ oc delete pod <old_pod> --force=true --grace-period=0s
  ```

  このコマンドは、シャットダウンまたはクラッシュしたノードで停止したボリュームを 6 分後に削除します。

5.8. OPENSHIFT CLI (oc) 関連の問題の診断

5.8.1. OpenShift CLI (oc) ログレベルについて

OpenShift CLI (**oc**) を使用すると、ターミナルからアプリケーションを作成し、OpenShift Container Platform プロジェクトを管理できます。

oc コマンド固有の問題が発生した場合は、**oc** のログレベルを引き上げ、コマンドで生成される API 要求、API 応答、および **curl** 要求の詳細を出力します。これにより、特定の **oc** コマンドの基礎となる操作の詳細ビューが得られます。これにより、障害の性質についての洞察が得られる可能性があります。

oc ログレベルは、1 から 10 まであります。以下の表は、 **oc** ログレベルの一覧とそれらの説明を示しています。

表5.3 OpenShift CLI (oc) ログレベル

ログレベル	説明
1 - 5	標準エラー (stderr) への追加のロギングはありません。
6	標準エラー (stderr) に API 要求のログを記録します。
7	標準エラー (stderr) に API 要求およびヘッダーのログを記録します。

ログレベル	説明
8	標準エラー (stderr) に API 要求、ヘッダーおよび本体、ならびに API 応答ヘッダーおよび本体のログを記録します。
9	標準エラー (stderr) に API 要求、ヘッダーおよび本体、API 応答ヘッダーおよび本体、**curl** 要求のログを記録します。
10	標準エラー (stderr) に API 要求、ヘッダーおよび本体、API 応答ヘッダーおよび本体、**curl** 要求のログを詳細に記録します。

5.8.2. OpenShift CLI (oc) ログレベルの指定

コマンドのログレベルを引き上げて、OpenShift CLI (**oc**) の問題を調査できます。

前提条件

- OpenShift CLI (**oc**) がインストールされている。

手順

1. **oc** コマンドの実行時に **oc** ログレベルを指定します。

```
$ oc <options> --loglevel <log_level>
```

2. 通常、OpenShift Container Platform ユーザーの現行セッショントークンは、必要に応じてログに記録される **curl** 要求に含まれます。また、手順に従って **oc** コマンドの基礎となるプロセスをテストする際に使用するために、現行ユーザーのセッショントークンを手動で取得することもできます。

```
$ oc whoami -t
```