



OpenShift Container Platform 4.9

备份和恢复

备份和恢复 OpenShift Container Platform 集群

OpenShift Container Platform 4.9 备份和恢复

备份和恢复 OpenShift Container Platform 集群

法律通告

Copyright © 2023 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.

摘要

本文档提供了备份集群数据以及从各种灾难场景中恢复的步骤。

目录

第 1 章 备份和恢复	3
1.1. CONTROL PLANE 备份和恢复操作	3
1.2. 应用程序备份和恢复操作	3
第 2 章 安全地关闭集群	5
2.1. 先决条件	5
2.2. 关闭集群	5
2.3. 其他资源	6
第 3 章 正常重启集群	7
3.1. 先决条件	7
3.2. 重启集群	7
第 4 章 应用程序备份和恢复	10
4.1. OADP 发行注册	10
4.2. OADP 功能和插件	11
4.3. 安装和配置 OADP	13
4.4. 备份和恢复	53
4.5. 故障排除	68
4.6. 与 OADP 一起使用的 API	77
4.7. 高级 OADP 特性和功能	81
第 5 章 CONTROL PLANE 备份和恢复	84
5.1. 备份 ETCD	84
5.2. 替换不健康的 ETCD 成员	86
5.3. 灾难恢复	110

第 1 章 备份和恢复

1.1. CONTROL PLANE 备份和恢复操作

作为集群管理员，您可能需要在一段时间内停止 OpenShift Container Platform 集群，并在以后重启集群。重启集群的一些原因是您需要对集群执行维护或希望降低资源成本。在 OpenShift Container Platform 中，您可以[对集群执行安全关闭](#)，以便在以后轻松重启集群。

您必须在关闭集群前[备份 etcd 数据](#)；etcd 是 OpenShift Container Platform 的键值存储，它会保留所有资源对象的状态。etcd 备份在灾难恢复中扮演着关键角色。在 OpenShift Container Platform 中，您还可以[替换不健康的 etcd 成员](#)。

当您希望集群再次运行时，请[安全地重启集群](#)。



注意

集群的证书在安装日期后一年后过期。您可以关闭集群，并在证书仍有效时安全地重启集群。虽然集群自动检索过期的 control plane 证书，但您仍需要[批准证书签名请求\(CSR\)](#)。

您可能会遇到 OpenShift Container Platform 无法按预期工作的一些情况，例如：

- 您有一个在重启后无法正常工作的集群，因为意外状况（如节点故障或网络连接问题）无法正常工作。
- 您已错误地删除了集群中的某些关键内容。
- 您丢失了大多数 control plane 主机，从而导致 etcd 仲裁丢失。

通过使用保存的 etcd 快照，始终可以通过将[集群恢复到之前的状态](#)来从灾难中恢复。

1.2. 应用程序备份和恢复操作

作为集群管理员，您可以使用 OpenShift API 进行数据保护(OADP)来备份和恢复在 OpenShift Container Platform 上运行的应用程序。

根据[下载 Velero CLI 工具](#)中的表，按照命名空间粒度来备份和恢复 Kubernetes 资源和内部镜像。OADP 使用快照或 Restic 来备份和恢复持久性卷(PV)。详情请查看[OADP 功能](#)。

1.2.1. OADP 要求

OADP 有以下要求：

- 您必须以具有 **cluster-admin** 角色的用户身份登录。
- 您必须具有用于存储备份的对象存储，比如以下存储类型之一：
 - OpenShift Data Foundation
 - Amazon Web Services
 - Microsoft Azure
 - Google Cloud Platform

- S3 兼容对象存储



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

- 要使用快照备份 PV，您必须有具有原生快照 API 的云存储，或者支持 Container Storage Interface(CSI)快照，如以下供应商：
 - Amazon Web Services
 - Microsoft Azure
 - Google Cloud Platform
 - 支持 CSI 快照的云存储，如 Ceph RBD 或 Ceph FS



注意

如果您不想使用快照备份 PV，可以使用 [Restic](#)，这由 OADP Operator 安装。

1.2.2. 备份和恢复应用程序

您可以通过创建 [Backup](#) 自定义资源(CR) 来备份应用程序。您可以配置以下备份选项：

- [Backup hooks](#)，在备份操作之前或之后运行命令
- [Scheduled backups](#)
- [Restic backups](#)

您可以通过创建一个 [Restore](#) CR 来恢复应用程序。您可以配置 [restore hook](#)，以便在 init 容器或应用程序容器中运行命令。

第 2 章 安全地关闭集群

本文档描述了安全关闭集群的过程。出于维护或者节约资源成本的原因，您可能需要临时关闭集群。

2.1. 先决条件

- 在关闭集群前进行 [etcd 备份](#)。

2.2. 关闭集群

您可以以安全的方式关闭集群，以便稍后重启集群。



注意

您可以在安装日期起的一年内关闭集群，并期望它可以正常重启。安装日期起一年后，集群证书会过期。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 已进行 etcd 备份。



重要

执行此流程前务必要进行 etcd 备份，以便在重启集群遇到任何问题时可以恢复集群。

例如，以下条件可能会导致重启的集群失败：

- 关机过程中的 etcd 数据崩溃
- 因硬件原因造成节点故障
- 网络连接问题

如果集群无法恢复，请按照以下步骤恢复到以前的集群状态。

流程

1. 如果您要长时间关闭集群，请确定证书过期的日期。

```
$ oc -n openshift-kube-apiserver-operator get secret kube-apiserver-to-kubelet-signer -o jsonpath='{.metadata.annotations.auth\.openshift\.io/certificate-not-after}'
```

输出示例

```
2022-08-05T14:37:50Zuser@user:~ $ 1
```

- 1** 为确保集群可以正常重启，请计划在指定的日期或之前重启集群。当集群重启时，可能需要您手动批准待处理的证书签名请求 (CSR) 来恢复 kubelet 证书。

2. 关闭集群中的所有节点。您可以从云供应商的 Web 控制台完成此操作，或者运行以下循环：

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do oc debug
node/${node} -- chroot /host shutdown -h 1; done
```

- 1 **-h 1** 表示此过程在 control-plane 节点关闭前可以持续的时间（以分钟为单位）。对于具有 10 个或更多节点的大型集群，请将它设置为 10 分钟或更长时间，以确保所有计算节点都已关闭。

输出示例

```
Starting pod/ip-10-0-130-169us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:17 UTC, use 'shutdown -c' to cancel.

Removing debug pod ...
Starting pod/ip-10-0-150-116us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:29 UTC, use 'shutdown -c' to cancel.
```

使用以下方法关闭节点可让 pod 安全终止，从而减少数据崩溃的可能性。



注意

为大规模集群调整关闭时间：

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do oc
debug node/${node} -- chroot /host shutdown -h 10; done
```



注意

在关闭前，不需要排空 OpenShift Container Platform 中附带的标准 pod 的 control plane 节点。

集群管理员负责确保在集群重启后，彻底重启自己的工作负载。如果因为自定义工作负载的原因已在关闭前排空 control plane 节点，您必须在重启后将 control plane 节点标记为可调度，然后集群才可以重新正常工作。

3. 关闭不再需要的集群依赖项，如外部存储或 LDAP 服务器。在进行操作前请务必查阅您的厂商文档。



重要

如果您在云供应商平台上部署了集群，请不要关闭、挂起或删除关联的云资源。如果您删除挂起的虚拟机的云资源，OpenShift Container Platform 可能无法成功恢复。

2.3. 其他资源

- [正常重启集群](#)
- [恢复到以前的集群状态](#)

第 3 章 正常重启集群

本文档论述了在安全关闭后重启集群的过程。

尽管在重启后集群应该可以正常工作，但可能会因为意外状况集群可能无法恢复，例如：

- 关机过程中的 etcd 数据崩溃
- 因硬件原因造成节点故障
- 网络连接问题

如果集群无法恢复，请按照以下步骤[恢复到以前的集群状态](#)。

3.1. 先决条件

- 已[安全关闭集群](#)。

3.2. 重启集群

您可以在集群被安全关闭后重启它。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 此流程假设您安全关闭集群。

流程

1. 启动所有依赖设备，如外部存储或 LDAP 服务器。
2. 启动所有集群机器。
使用适合您的云环境的方法启动机器，例如从云供应商的 Web 控制台启动机器。

等待大约 10 分钟，然后继续检查 control plane 节点的状态。

3. 验证所有 control plane 节点都已就绪。

```
$ oc get nodes -l node-role.kubernetes.io/master
```

如果状态为 **Ready**，如以下输出中所示，则代表 control plane 节点已就绪：

```
NAME                                STATUS ROLES  AGE  VERSION
ip-10-0-168-251.ec2.internal        Ready  master    75m  v1.22.1
ip-10-0-170-223.ec2.internal        Ready  master    75m  v1.22.1
ip-10-0-211-16.ec2.internal         Ready  master    75m  v1.22.1
```

4. 如果 control plane 节点没有就绪，请检查是否有待批准的证书签名请求 (CSR)。
 - a. 获取当前 CSR 列表：

```
$ oc get csr
```

- b. 查看一个 CSR 的详细信息以验证其是否有效：

```
$ oc describe csr <csr_name> ❶
```

❶ <csr_name> 是当前 CSR 列表中 CSR 的名称。

- c. 批准每个有效的 CSR：

```
$ oc adm certificate approve <csr_name>
```

5. 在 control plane 节点就绪后，验证所有 worker 节点是否已就绪。

```
$ oc get nodes -l node-role.kubernetes.io/worker
```

如果状态为 **Ready**，如下所示，则代表 worker 节点已就绪：

```
NAME                                STATUS ROLES  AGE  VERSION
ip-10-0-179-95.ec2.internal        Ready  worker  64m  v1.22.1
ip-10-0-182-134.ec2.internal        Ready  worker  64m  v1.22.1
ip-10-0-250-100.ec2.internal        Ready  worker  64m  v1.22.1
```

6. 如果 worker 节点未就绪，请检查是否有待批准的证书签名请求(CSR)。

- a. 获取当前 CSR 列表：

```
$ oc get csr
```

- b. 查看一个 CSR 的详细信息以验证其是否有效：

```
$ oc describe csr <csr_name> ❶
```

❶ <csr_name> 是当前 CSR 列表中 CSR 的名称。

- c. 批准每个有效的 CSR：

```
$ oc adm certificate approve <csr_name>
```

7. 验证集群是否已正确启动。

- a. 检查是否有降级的集群 Operator。

```
$ oc get clusteroperators
```

确定没有 **DEGRADED** 条件为 **True** 的集群 Operator。

```
NAME                                VERSION AVAILABLE PROGRESSING DEGRADED
SINCE
authentication                       4.9.0 True      False      False      59m
cloud-credential                       4.9.0 True      False      False      85m
cluster-autoscaler                     4.9.0 True      False      False      73m
config-operator                        4.9.0 True      False      False      73m
```

```

console                4.9.0  True   False   False   62m
csi-snapshot-controller 4.9.0  True   False   False   66m
dns                    4.9.0  True   False   False   76m
etcd                   4.9.0  True   False   False   76m
...

```

b. 检查所有节点是否处于 **Ready** 状态：

```
$ oc get nodes
```

检查所有节点的状态是否为 **Ready**。

```

NAME                                STATUS ROLES  AGE  VERSION
ip-10-0-168-251.ec2.internal        Ready  master  82m  v1.22.1
ip-10-0-170-223.ec2.internal        Ready  master  82m  v1.22.1
ip-10-0-179-95.ec2.internal         Ready  worker  70m  v1.22.1
ip-10-0-182-134.ec2.internal        Ready  worker  70m  v1.22.1
ip-10-0-211-16.ec2.internal         Ready  master  82m  v1.22.1
ip-10-0-250-100.ec2.internal        Ready  worker  69m  v1.22.1

```

如果集群无法正确启动，您可能需要使用 etcd 备份来恢复集群。

其他资源

- 如果集群重启后无法恢复，请参阅[恢复到以前的集群状态](#)。

第 4 章 应用程序备份和恢复

4.1. OADP 发行注记

OpenShift API for Data Protection (OADP) 的发行注记介绍了新的功能和增强功能、已弃用的功能、产品建议、已知问题和解决问题。

4.1.1. OADP 1.1.2 发行注记

OADP 1.1.2 发行注记包括产品建议、修复的错误列表和已知问题的描述。

4.1.1.1. 产品建议

VolSync

要准备从 VolSync 0.5.1 升级到 VolSync **stable** 频道中的最新版本，您必须运行以下命令在 **openshift-adp** 命名空间中添加此注解：

```
$ oc annotate --overwrite namespace/openshift-adp volsync.backube/privileged-movers='true'
```

Velero

在这个发行版本中，Velero 已从 1.9.2 升级到 [1.9.5](#) 版本。

Restic

在本发行版本中，Restic 从 0.13.1 升级到 [0.14.0](#) 版本。

4.1.1.2. 修复了的错误

这个版本解决了以下程序错误：

- [OADP-1150](#)
- [OADP-290](#)
- [OADP-1056](#)

4.1.1.3. 已知问题

这个版本有以下已知问题：

- OADP 目前不支持使用 Velero ([OADP-778](#)) 中的 restic 备份和恢复 AWS EFS 卷。
- CSI 备份可能会因为每个 PVC 的 **VolumeSnapshotContent** 快照限制而失败。您可以创建同一持久性卷声明(PVC)的许多快照，但无法调度定期创建快照：
 - 对于 CephFS，您可以为每个 PVC 创建最多 100 个快照。([OADP-804](#))
 - 对于 RADOS 块设备 (RBD)，您可以为每个 PVC 创建最多 512 个快照。([OADP-975](#))

如需更多信息，请参阅[卷快照](#)。

4.1.2. OADP 1.1.1 发行注记

OADP 1.1.1 发行注记包括产品建议和已知问题的描述。

4.1.2.1. 产品建议

在安装 OADP 1.1.1 前，建议安装 VolSync 0.5.1 或升级到它。

4.1.2.2. 已知问题

这个版本有以下已知问题：

- OADP 目前不支持使用 Velero ([OADP-778](#)) 中的 restic 备份和恢复 AWS EFS 卷。
- CSI 备份可能会因为每个 PVC 的 **VolumeSnapshotContent** 快照限制而失败。您可以创建同一持久性卷声明(PVC)的许多快照，但无法调度定期创建快照：
 - 对于 CephFS，您可以为每个 PVC 创建最多 100 个快照。
 - 对于 RADOS 块设备 (RBD)，您可以为每个 PVC 创建最多 512 个快照。([OADP-804](#)) 和 ([OADP-975](#))
如需更多信息，请参阅[卷快照](#)。

4.2. OADP 功能和插件

OpenShift API 用于数据保护(OADP)功能，提供用于备份和恢复应用的选项。

默认插件使 Velero 能够与某些云供应商集成，并备份和恢复 OpenShift Container Platform 资源。

4.2.1. OADP 功能

OpenShift API 用于数据保护(OADP)支持以下功能：

Backup

您可以备份集群中的所有资源，或者您可以按类型、命名空间或标签过滤资源。

OADP 通过将 Kubernetes 对象和内部镜像保存为对象存储上的存档文件来备份 Kubernetes 对象和内部镜像。OADP 使用原生云快照 API 或通过容器存储接口(CSI)创建快照来备份持久性卷(PV)。对于不支持快照的云供应商，OADP 使用 Restic 备份资源和 PV 数据。

恢复

您可以从备份中恢复资源和 PV。您可以恢复备份中的所有对象，或者根据命名空间、PV 或标签过滤恢复的对象。

调度

您可以通过指定的间隔调度备份。

钩子

您可以使用 hook 在 pod 上的容器中运行命令，如 **fsfreeze** 以冻结文件系统。您可以将 hook 配置为在备份或恢复之前或之后运行。恢复 hook 可以在 init 容器或应用程序容器中运行。

4.2.2. OADP 插件

用于数据保护(OADP)的 OpenShift API 提供了与存储供应商集成的默认 Velero 插件，以支持备份和恢复操作。您可以根据 Velero 插件创建[自定义插件](#)。

OADP 还为 OpenShift Container Platform 资源备份和 Container Storage Interface(CSI)快照提供了插件。

表 4.1. OADP 插件

OADP 插件	功能	存储位置
aws	使用对象存储备份和恢复 Kubernetes 对象。	AWS S3
	使用快照备份和恢复卷。	AWS EBS
azure	使用对象存储备份和恢复 Kubernetes 对象。	Microsoft Azure Blob 存储
	使用快照备份和恢复卷。	Microsoft Azure 管理的磁盘
gcp	使用对象存储备份和恢复 Kubernetes 对象。	Google Cloud Storage
	使用快照备份和恢复卷。	Google Compute Engine 磁盘
openshift	使用对象存储备份和恢复 OpenShift Container Platform 资源。 [1]	对象存储
csi	使用 CSI 快照备份和恢复卷。 [2]	支持 CSI 快照的云存储

1. 必需。
2. **csi** 插件使用 [Velero CSI beta 快照 API](#)。

4.2.3. 关于 OADP Velero 插件

安装 Velero 时，您可以配置两种类型的插件：

- 默认云供应商插件
- 自定义插件

两种类型的插件都是可选的，但大多数用户都会至少配置一个云供应商插件。

4.2.3.1. 默认 Velero 云供应商插件

当您在部署过程中配置 `oadp_v1alpha1_dpa.yaml` 文件时，您可以安装以下默认 Velero 云供应商插件：

- **aws** (Amazon Web Services)
- **gcp** (Google Cloud Platform)
- **azure** (Microsoft Azure)

- **openshift** (OpenShift Velero plugin)
- **csi** (Container Storage Interface)
- **kubevirt** (KubeVirt)

在部署过程中，您可以在 `oadp_v1alpha1_dpa.yaml` 文件中指定所需的默认插件。

示例文件

以下 `.yaml` 文件会安装 **openshift**、**aws**、**azure** 和 **gcp** 插件：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - azure
        - gcp
```

4.2.3.2. 自定义 Velero 插件

您可在部署期间配置 `oadp_v1alpha1_dpa.yaml` 文件时，通过指定插件 **镜像**和**名称**来安装自定义 Velero 插件。

在部署过程中，您可以在 `oadp_v1alpha1_dpa.yaml` 文件中指定所需的自定义插件。

示例文件

以下 `.yaml` 文件会安装默认的 **openshift**、**azure** 和 **gcp** 插件，以及一个自定义插件，其名称为 **custom-plugin-example** 和镜像 `quay.io/example-repo/custom-velero-plugin`：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - azure
        - gcp
      customPlugins:
        - name: custom-plugin-example
          image: quay.io/example-repo/custom-velero-plugin
```

4.3. 安装和配置 OADP

4.3.1. 关于安装 OADP

作为集群管理员，您可以通过安装 OADP Operator 来为数据保护(OADP)安装 OpenShift API。OADP Operator 会安装 [Velero 1.7](#)。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。

要备份 Kubernetes 资源和内部镜像，必须将对象存储用作备份位置，如以下存储类型之一：

- [Amazon Web Services](#)
- [Microsoft Azure](#)
- [Google Cloud Platform](#)
- [多云对象网关](#)
- S3 兼容对象存储，如 Noobaa 或 Minio



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

您可以使用快照或 Restic 备份持久性卷(PV)。

要使用快照备份 PV，您必须有一个支持原生快照 API 或 Container Storage Interface(CSI)快照的云供应商，如以下云供应商之一：

- [Amazon Web Services](#)
- [Microsoft Azure](#)
- [Google Cloud Platform](#)
- 支持 CSI 快照的云供应商，如 [OpenShift Container Storage](#)

如果您的云供应商不支持快照，或者您的存储是 NFS，您可以使用 [Restic](#) 备份应用程序。

您可以为存储供应商凭证创建一个 **Secret** 对象，然后安装数据保护应用程序。

4.3.1.1. 配置 NooBaa 以便在 OpenShift Container Storage 上进行灾难恢复

如果您在 OpenShift Container Storage 上为 NooBaa bucket **backupStorageLocation** 使用集群存储，请将 NooBaa 配置为外部对象存储。



警告

将 NooBaa 配置为外部对象存储可能会导致备份不可用。

流程

- 将 NooBaa 配置为外部对象存储，如[为混合或多云添加存储资源](#)中所述。

其他资源

- [Velero 文档](#)中的备份位置和快照位置概述。

/// 模块包含在以下 assemblies 中：

4.3.1.2. 关于 OADP 更新频道

安装 OADP Operator 时，您可以选择 **更新频道**。这个频道决定到您接收到的 OADP Operator 和 Velero 的哪些升级。您可以随时切换频道。

有三个更新频道：

- **stable** 频道包含 OADP ClusterServiceVersion' 的最新次要更新 (y-stream 更新) 和补丁 (z-stream 更新)。当发布每个新版本时，OADP Operator 的可用 **ClusterServiceVersion** 将附加最新的可用次要补丁。
- **stable-1.0** 频道包含 **oadp.v1.0.z**，它是最新的 OADP 1.0 **ClusterServiceVersion**。
- **stable-1.1** 频道包含 **oadp.v1.1.z**，它是最新的 OADP 1.1 **ClusterServiceVersion**。

哪个更新频道适合您？

- 选择 **stable** 更新频道来安装最新的 stable OADP 版本，并接收次要更新和补丁。如果您选择此频道，您将收到版本 x.y.z 的所有 y-stream 和 z-stream 的更新。
- 选择 **stable-1.y** 更新频道来安装 OADP 1.y，并继续为其接受补丁。如果您选择此频道，您将收到版本 1.y.z 的所有 z-stream。

何时需要切换更新频道？

- 如果您安装了 OADP 1.y，并且只想接收那个 y-stream 的补丁，则必须从 **stable** 更新频道切换到 **stable-1.y** 更新频道。然后，您将收到版本 1.y.z 的所有 z-stream 补丁。
- 如果您安装了 OADP 1.0，希望升级到 OADP 1.1，然后只接收 OADP 1.1 的补丁，则必须从 **stable-1.0** 更新频道切换到 **stable-1.1** 更新频道。然后，您将收到版本 1.1.z 的所有 z-stream 补丁。
- 如果您安装了 OADP 1.y，且 y 大于 0，并且希望切换到 OADP 1.0，则必须 **卸载** OADP Operator，然后使用 **stable-1.0** 更新频道重新安装。然后，您将收到 1.0.z 版本的所有 z-stream 补丁。



注意

您无法通过切换更新频道从 OADP 1.y 切换到 OADP 1.0。您必须卸载 Operator，然后重新安装它。

其他资源

- [集群服务版本](#)

4.3.2. 为 Amazon Web Services 进行数据保护安装和配置 OpenShift API

您可以通过安装 OADP Operator、为 Velero 配置 AWS，然后安装数据保护应用程序，安装带有 Amazon Web Services(AWS)的数据保护(OADP)的 OpenShift API。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

要在受限网络环境中安装 OADP Operator，您必须首先禁用默认的 OperatorHub 源并镜像 Operator 目录。详情请参阅[在受限网络中使用 Operator Lifecycle Manager](#)。

4.3.2.1. 安装 OADP Operator

您可以使用 Operator Lifecycle Manager(OLM)在 OpenShift Container Platform 4.9 上安装 Data Protection(OADP)Operator 的 OpenShift API。

OADP Operator 会安装 [Velero 1.7](#)。

先决条件

- 您必须以具有 **cluster-admin** 权限的用户身份登录。

流程

1. 在 OpenShift Container Platform Web 控制台中，点击 **Operators** → **OperatorHub**。
2. 使用 **Filter by keyword** 字段查找 **OADP Operator**。
3. 选择 **OADP Operator** 并点 **Install**。
4. 点 **Install** 在 **openshift-adp** 项目中安装 Operator。
5. 点 **Operators** → **Installed Operators** 来验证安装。

4.3.2.2. 配置 Amazon Web Services

您可以为 OpenShift API 配置 Amazon Web Services(AWS)以进行数据保护(OADP)。

先决条件

- 已安装 [AWS CLI](#)。

流程

1. 设置 **BUCKET** 变量：

```
$ BUCKET=<your_bucket>
```

2. 设置 **REGION** 变量：

```
$ REGION=<your_region>
```

3. 创建 AWS S3 存储桶：

```
$ aws s3api create-bucket \
  --bucket $BUCKET \
  --region $REGION \
  --create-bucket-configuration LocationConstraint=$REGION 1
```

- 1** **us-east-1** 不支持 **LocationConstraint**。如果您的区域是 **us-east-1**，忽略 **--create-bucket-configuration LocationConstraint=\$REGION**。

4. 创建一个 IAM 用户：

```
$ aws iam create-user --user-name velero 1
```

- 1** 如果要使用 Velero 备份具有多个 S3 存储桶的集群，请为每个集群创建一个唯一用户名。

5. 创建 **velero-policy.json** 文件：

```
$ cat > velero-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:DescribeSnapshots",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:CreateSnapshot",
        "ec2>DeleteSnapshot"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```

    "Action": [
      "s3:GetObject",
      "s3:DeleteObject",
      "s3:PutObject",
      "s3:AbortMultipartUpload",
      "s3:ListMultipartUploadParts"
    ],
    "Resource": [
      "arn:aws:s3:::${BUCKET}/*"
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:ListBucket",
      "s3:GetBucketLocation",
      "s3:ListBucketMultipartUploads"
    ],
    "Resource": [
      "arn:aws:s3:::${BUCKET}"
    ]
  }
]
}
EOF

```

6. 附加策略，为 **velero** 用户提供所需的最低权限：

```

$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero \
  --policy-document file://velero-policy.json

```

7. 为 **velero** 用户创建访问密钥：

```

$ aws iam create-access-key --user-name velero

```

输出示例

```

{
  "AccessKey": {
    "UserName": "velero",
    "Status": "Active",
    "CreateDate": "2017-07-31T22:24:41.576Z",
    "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>,
    "AccessKeyId": <AWS_ACCESS_KEY_ID>
  }
}

```

8. 创建 **credentials-velero** 文件：

```

$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>

```

```
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

在安装数据保护应用程序前，您可以使用 **credentials-velero** 文件为 AWS 创建 **Secret** 对象。

4.3.2.3. 创建用于备份和快照位置的 secret

如果使用相同的凭证，您可以为备份和快照位置创建一个 **Secret** 对象。

Secret 的默认名称为 **cloud-credentials**。

先决条件

- 对象存储和云存储必须使用相同的凭证。
- 您必须为 Velero 配置对象存储。
- 您必须以适当的格式为对象存储创建一个 **credentials-velero** 文件。



注意

DataProtectionApplication 自定义资源(CR)需要一个 **Secret** 进行安装。如果没有指定 **spec.backupLocations.credential.name** 值，则会使用默认名称。

如果您不想指定备份位置或快照位置，则必须使用空 **credentials-velero** 文件创建带有默认名称的 **Secret**。

流程

- 使用默认名称创建 **Secret**：

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

在安装 Data Protection Application 时，**secret** 会在 **DataProtectionApplication** CR 的 **spec.backupLocations.credential** 块中引用。

4.3.2.3.1. 为不同的备份和恢复位置凭证配置 secret

如果您的备份和快照位置使用不同的凭证，您可以在 **credentials-velero** 文件中创建单独的配置集。

然后，您可以创建一个 **Secret** 对象并在 **DataProtectionApplication** 自定义资源(CR)中指定配置集。

流程

1. 使用备份和快照位置的独立配置集创建一个 **credentials-velero** 文件，如下例所示：

```
[backupStorage]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>

[volumeSnapshot]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. 使用 **credentials-velero** 文件创建 **Secret** 对象：

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero 1
```

3. 在 **DataProtectionApplication** CR 中添加配置集，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
  - name: default
    velero:
      provider: aws
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: <prefix>
      config:
        region: us-east-1
        profile: "backupStorage"
      credential:
        key: cloud
        name: cloud-credentials
  snapshotLocations:
  - name: default
    velero:
      provider: aws
      config:
        region: us-west-2
        profile: "volumeSnapshot"
```

4.3.2.4. 配置数据保护应用程序

您可以配置 Velero 资源分配并启用自签名 CA 证书。

4.3.2.4.1. 设置 Velero CPU 和内存分配

您可以通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为 **Velero** pod 设置 CPU 和内存分配。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.configuration.velero.podConfig.ResourceAllocations** 块中的值，如下例所示：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> ❶
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi

```

❶ ❶ 指定要向 Velero podSpec 提供的节点选择器

4.3.2.4.2. 启用自签名 CA 证书

您必须通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为对象存储启用自签名 CA 证书，以防止由未知颁发机构签名的证书。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.backupLocations.velero.objectStorage.caCert** 参数和 **spec.backupLocations.velero.config** 参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> ❶
        config:
          insecureSkipTLSVerify: "false" ❷
  ...

```

- 1 指定 Base46 编码的 CA 证书字符串。
- 2 `insecureSkipTLSVerify` 配置可以设置为 `"true"` 或 `"false"`。如果设置为 `"true"`，则禁用 SSL/TLS 安全性。如果设置为 `"false"`，则启用 SSL/TLS 安全性。

4.3.2.5. 安装数据保护应用程序

您可以通过创建 `DataProtectionApplication` API 的实例来安装数据保护应用程序(DPA)。

先决条件

- 您必须安装 OADP Operator。
- 您必须将对象存储配置为备份位置。
- 如果使用快照来备份 PV，云供应商必须支持原生快照 API 或 Container Storage Interface(CSI) 快照。
- 如果备份和快照位置使用相同的凭证，您必须创建带有默认名称 `cloud-credentials` 的 `Secret`。
- 如果备份和快照位置使用不同的凭证，则必须使用默认名称 `cloud-credentials` 创建一个 `Secret`，其中包含备份和快照位置凭证的独立配置集。



注意

如果您不想在安装过程中指定备份或快照位置，您可以使用空 `credentials-velero` 文件创建默认 `Secret`。如果没有默认 `Secret`，安装将失败。

流程

1. 点 `Operators` → `Installed Operators` 并选择 `OADP Operator`。
2. 在 `Provided APIs` 下，点 `DataProtectionApplication` 框中的 `Create 实例`。
3. 点 `YAML View` 并更新 `DataProtectionApplication` 清单的参数：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift 1
        - aws
    restic:
      enable: true 2
      podConfig:
        nodeSelector: <node selector> 3
  backupLocations:
    - name: default
      velero:
```

```

provider: aws
default: true
objectStorage:
  bucket: <bucket_name> 4
  prefix: <prefix> 5
config:
  region: <region>
  profile: "default"
credential:
  key: cloud
  name: cloud-credentials 6
snapshotLocations: 7
- name: default
velero:
  provider: aws
  config:
    region: <region> 8
    profile: "default"

```

- 1 **openshift** 插件是必须的，才能在 OpenShift Container Platform 集群上备份和恢复命名空间。
- 2 如果要禁用 Restic 安装，设置为 **false**。Restic 部署一个守护进程集，这意味着每个 worker 节点都运行有 **Restic** pod。您可以通过在 **Backup** CR 中添加 **spec.defaultVolumesToRestic: true** 来配置 Restic 以进行备份。
- 3 指定要提供给 Restic podSpec 的节点选择器。
- 4 指定存储桶作为备份存储位置。如果存储桶不是 Velero 备份的专用存储桶，您必须指定一个前缀。
- 5 如果存储桶用于多个目的，请为 Velero 备份指定一个前缀，如 **velero**。
- 6 指定您创建的 **Secret** 对象的名称。如果没有指定这个值，则使用默认值 **cloud-credentials**。如果您指定了自定义名称，则使用自定义名称进行备份位置。
- 7 如果您使用 CSI 快照或 Restic 备份 PV，则不需要指定快照位置。
- 8 快照位置必须与 PV 位于同一区域。

4. 点 **Create**。

5. 通过查看 OADP 资源来验证安装：

```
$ oc get all -n openshift-adp
```

输出示例

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/restic-9cq4q	1/1	Running	0	94s
pod/restic-m4lts	1/1	Running	0	94s
pod/restic-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

```

NAME                                TYPE      CLUSTER-IP      EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service ClusterIP 172.30.70.140
<none>    8443/TCP 2m8s

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/restic 3      3      3      3          3          <none>    96s

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager 1/1    1          1          2m9s
deployment.apps/velero                          1/1    1          1          96s

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47 1      1      1      2m9s
replicaset.apps/velero-588db7f655                        1      1      1      96s

```

4.3.2.5.1. 在 DataProtectionApplication CR 中启用 CSI

您可以在 **DataProtectionApplication** 自定义资源(CR)中启用 Container Storage Interface(CSI)来备份持久性卷，以使用 CSI 快照备份持久性卷。

先决条件

- 云供应商必须支持 CSI 快照。

流程

- 编辑 **DataProtectionApplication** CR，如下例所示：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi ❶

```

- ❶ 添加 **csi** 默认插件。

4.3.3. 为 Microsoft Azure 的数据保护安装和配置 OpenShift API

您可以通过安装 OADP Operator、为 Velero 配置 Azure，然后安装数据保护应用程序，使用 Microsoft Azure 安装 OpenShift API for Data ProtectionP。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

要在受限网络环境中安装 OADP Operator，您必须首先禁用默认的 OperatorHub 源并镜像 Operator 目录。详情请参阅[在受限网络中使用 Operator Lifecycle Manager](#)。

4.3.3.1. 安装 OADP Operator

您可以使用 Operator Lifecycle Manager(OLM)在 OpenShift Container Platform 4.9 上安装 Data Protection(OADP)Operator 的 OpenShift API。

OADP Operator 会安装 [Velero 1.7](#)。

先决条件

- 您必须以具有 **cluster-admin** 权限的用户身份登录。

流程

1. 在 OpenShift Container Platform Web 控制台中，点击 **Operators** → **OperatorHub**。
2. 使用 **Filter by keyword** 字段查找 **OADP Operator**。
3. 选择 **OADP Operator** 并点 **Install**。
4. 点 **Install** 在 **openshift-adp** 项目中安装 Operator。
5. 点 **Operators** → **Installed Operators** 来验证安装。

4.3.3.2. 配置 Microsoft Azure

您可以为 OpenShift API 配置 Microsoft Azure 以进行数据保护(OADP)。

先决条件

- 已安装 [Azure CLI](#)。

流程

1. 登录到 Azure:

```
$ az login
```

2. 设置 **AZURE_RESOURCE_GROUP** 变量 :

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

3. 创建 Azure 资源组 :

```
$ az group create -n $AZURE_RESOURCE_GROUP --location CentralUS 1
```

1 指定位置。

4. 设置 **AZURE_STORAGE_ACCOUNT_ID** 变量：

```
$ AZURE_STORAGE_ACCOUNT_ID="velero$(uuidgen | cut -d '-' -f5 | tr '[A-Z]' '[a-z]')
```

5. 创建 Azure 存储帐户：

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

6. 设置 **BLOB_CONTAINER** 变量：

```
$ BLOB_CONTAINER=velero
```

7. 创建 Azure Blob 存储容器：

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

8. 获取存储帐户访问密钥：

```
$ AZURE_STORAGE_ACCOUNT_ACCESS_KEY=`az storage account keys list \
  --account-name $AZURE_STORAGE_ACCOUNT_ID \
  --query "[?keyName == 'key1'].value" -o tsv`
```

9. 创建具有最低所需权限的自定义角色：

```
AZURE_ROLE=Velero
az role definition create --role-definition '{
  "Name": "$AZURE_ROLE",
  "Description": "Velero related permissions to perform backups, restores and deletions",
  "Actions": [
    "Microsoft.Compute/disks/read",
    "Microsoft.Compute/disks/write",
    "Microsoft.Compute/disks/endGetAccess/action",
    "Microsoft.Compute/disks/beginGetAccess/action",
    "Microsoft.Compute/snapshots/read",
    "Microsoft.Compute/snapshots/write",
    "Microsoft.Compute/snapshots/delete",
    "Microsoft.Storage/storageAccounts/listkeys/action",
    "Microsoft.Storage/storageAccounts/regeneratekey/action"
```

```

    ],
    "AssignableScopes": ["/subscriptions/${AZURE_SUBSCRIPTION_ID}"]
  }'

```

10. 创建 **credentials-velero** 文件：

```

$ cat << EOF > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_STORAGE_ACCOUNT_ACCESS_KEY=${AZURE_STORAGE_ACCOUNT_ACCESS_KEY} 1
AZURE_CLOUD_NAME=AzurePublicCloud
EOF

```

1 必需。如果 **credentials-velero** 文件只包含服务主体凭证，则无法备份内部镜像。

在安装 Data Protection 应用前，您可以使用 **credentials-velero** 文件为 Azure 创建 **Secret** 对象。

4.3.3.3. 创建用于备份和快照位置的 secret

如果使用相同的凭证，您可以为备份和快照位置创建一个 **Secret** 对象。

Secret 的默认名称为 **cloud-credentials-azure**。

先决条件

- 对象存储和云存储必须使用相同的凭证。
- 您必须为 Velero 配置对象存储。
- 您必须以适当的格式为对象存储创建一个 **credentials-velero** 文件。



注意

DataProtectionApplication 自定义资源(CR)需要一个 **Secret** 进行安装。如果没有指定 **spec.backupLocations.credential.name** 值，则会使用默认名称。

如果您不想指定备份位置或快照位置，则必须使用空 **credentials-velero** 文件创建带有默认名称的 **Secret**。

流程

- 使用默认名称创建 **Secret**：

```

$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file
cloud=credentials-velero

```

在安装 Data Protection Application 时，**secret** 会在 **DataProtectionApplication** CR 的 **spec.backupLocations.credential** 块中引用。

4.3.3.3.1. 为不同的备份和恢复位置凭证配置 secret

如果您的备份和恢复位置使用不同的凭证，请创建两个 **Secret** 对象：

- 具有自定义名称的备份位置 **Secret**。自定义名称在 **DataProtectionApplication** 自定义资源(CR)的 **spec.backupLocations** 块中指定。
- 带有默认名称 **cloud-credentials-azure** 的快照位置 **Secret**。此 **Secret** 不在 **DataProtectionApplication** CR 中指定。

流程

1. 为您的云供应商为快照位置创建一个 **credentials-velero** 文件。
2. 使用默认名称为快照位置创建 **Secret**：

```
$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file cloud=credentials-velero
```

3. 为您的对象存储创建一个用于备份位置的 **credentials-velero** 文件。

4. 使用自定义名称为备份位置创建 **Secret**：

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. 将带有自定义名称的 **Secret** 添加到 **DataProtectionApplication** CR 中，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
      config:
        resourceGroup: <azure_resource_group>
        storageAccount: <azure_storage_account_id>
        subscriptionId: <azure_subscription_id>
        storageAccountKeyEnvVar: AZURE_STORAGE_ACCOUNT_ACCESS_KEY
      credential:
        key: cloud
        name: <custom_secret> ①
      provider: azure
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: <prefix>
  snapshotLocations:
    - velero:
      config:
        resourceGroup: <azure_resource_group>
        subscriptionId: <azure_subscription_id>
```

```
incremental: "true"
name: default
provider: azure
```

- 1 具有自定义名称的备份位置 **Secret**。

4.3.3.4. 配置数据保护应用程序

您可以配置 Velero 资源分配并启用自签名 CA 证书。

4.3.3.4.1. 设置 Velero CPU 和内存分配

您可以通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为 **Velero** pod 设置 CPU 和内存分配。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.configuration.velero.podConfig.ResourceAllocations** 块中的值，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

- 1 指定要向 Velero podSpec 提供的节点选择器

4.3.3.4.2. 启用自签名 CA 证书

您必须通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为对象存储启用自签名 CA 证书，以防止由未知颁发机构签名的证书。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.backupLocations.velero.objectStorage.caCert** 参数和 **spec.backupLocations.velero.config** 参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  backupLocations:
  - name: default
    velero:
      provider: aws
      default: true
      objectStorage:
        bucket: <bucket>
        prefix: <prefix>
        caCert: <base64_encoded_cert_string> ❶
      config:
        insecureSkipTLSVerify: "false" ❷
  ...

```

- ❶ 指定 Base46 编码的 CA 证书字符串。
- ❷ **insecureSkipTLSVerify** 配置可以设置为 **"true"** 或 **"false"**。如果设置为 **"true"**，则禁用 SSL/TLS 安全性。如果设置为 **"false"**，则启用 SSL/TLS 安全性。

4.3.3.5. 安装数据保护应用程序

您可以通过创建 **DataProtectionApplication** API 的实例来安装数据保护应用程序(DPA)。

先决条件

- 您必须安装 OADP Operator。
- 您必须将对象存储配置为备份位置。
- 如果使用快照来备份 PV，云供应商必须支持原生快照 API 或 Container Storage Interface(CSI) 快照。
- 如果备份和快照位置使用相同的凭证，您必须创建带有默认名称 **cloud-credentials-azure** 的 **Secret**。
- 如果备份和快照位置使用不同的凭证，您必须创建两个 **Secret**：
 - 带有备份位置的自定义名称的 **secret**。您可以将此 **Secret** 添加到 **DataProtectionApplication** CR 中。
 - 带有默认名称 **cloud-credentials-azure** 的 **secret**，用于快照位置。这个 **Secret** 不在 **DataProtectionApplication** CR 中被引用。



注意

如果您不想在安装过程中指定备份或快照位置，您可以使用空 **credentials-velero** 文件创建默认 **Secret**。如果没有默认 **Secret**，安装将失败。

流程

1. 点 **Operators** → **Installed Operators** 并选择 **OADP Operator**。
2. 在 **Provided APIs** 下，点 **DataProtectionApplication** 框中的 **Create 实例**。
3. 点 **YAML View** 并更新 **DataProtectionApplication** 清单的参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - azure
        - openshift ①
    restic:
      enable: true ②
      podConfig:
        nodeSelector: <node selector> ③
  backupLocations:
    - velero:
      config:
        resourceGroup: <azure_resource_group> ④
        storageAccount: <azure_storage_account_id> ⑤
        subscriptionId: <azure_subscription_id> ⑥
        storageAccountKeyEnvVar: AZURE_STORAGE_ACCOUNT_ACCESS_KEY
      credential:
        key: cloud
        name: cloud-credentials-azure ⑦
      provider: azure
      default: true
      objectStorage:
        bucket: <bucket_name> ⑧
        prefix: <prefix> ⑨
  snapshotLocations: ⑩
    - velero:
      config:
        resourceGroup: <azure_resource_group>
        subscriptionId: <azure_subscription_id>
        incremental: "true"
      name: default
      provider: azure
  
```

- ① **openshift** 插件是必须的，才能在 OpenShift Container Platform 集群上备份和恢复命名空间。

- 2 如果要禁用 Restic 安装，设置为 **false**。Restic 部署一个守护进程集，这意味着每个 worker 节点都运行有 **Restic** pod。您可以通过在 **Backup** CR 中添加
- 3 指定要提供给 Restic podSpec 的节点选择器。
- 4 指定 Azure 资源组。
- 5 指定 Azure 存储帐户 ID。
- 6 指定 Azure 订阅 ID。
- 7 如果没有指定这个值，则使用默认值 **cloud-credentials-azure**。如果您指定了自定义名称，则使用自定义名称进行备份位置。
- 8 指定存储桶作为备份存储位置。如果存储桶不是 Velero 备份的专用存储桶，您必须指定一个前缀。
- 9 如果存储桶用于多个目的，请为 Velero 备份指定一个前缀，如 **velero**。
- 10 如果您使用 CSI 快照或 Restic 备份 PV，则不需要指定快照位置。

4. 点 **Create**。

5. 通过查看 OADP 资源来验证安装：

```
$ oc get all -n openshift-adp
```

输出示例

```

NAME                                READY STATUS  RESTARTS  AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running  0         2m8s
pod/restic-9cq4q                                1/1   Running  0         94s
pod/restic-m4lts                                1/1   Running  0         94s
pod/restic-pv4kr                                1/1   Running  0         95s
pod/velero-588db7f655-n842v                    1/1   Running  0         95s

NAME                                TYPE          CLUSTER-IP      EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>    8443/TCP    2m8s

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/restic  3        3        3      3           3          <none>    96s

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1            1          2m9s
deployment.apps/velero                          1/1    1            1          96s

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47  1        1        1      2m9s
replicaset.apps/velero-588db7f655                          1        1        1      96s

```

4.3.3.5.1. 在 DataProtectionApplication CR 中启用 CSI

您可以在 **DataProtectionApplication** 自定义资源(CR)中启用 Container Storage Interface(CSI)来备份持久性卷，以使用 CSI 快照备份持久性卷。

先决条件

- 云供应商必须支持 CSI 快照。

流程

- 编辑 **DataProtectionApplication** CR，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi 1
```

- 1** 添加 **csi** 默认插件。

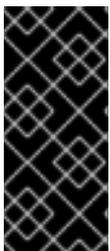
4.3.4. 安装和配置 OpenShift API 以进行 Google Cloud Platform 的数据保护

您可以通过安装 OADP Operator、为 Velero 配置 GCP，然后安装数据保护应用程序，使用 Google Cloud Platform(GCP)安装 OpenShift API for Data Protection(GCP)。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

要在受限网络环境中安装 OADP Operator，您必须首先禁用默认的 OperatorHub 源并镜像 Operator 目录。详情请参阅[在受限网络中使用 Operator Lifecycle Manager](#)。

4.3.4.1. 安装 OADP Operator

您可以使用 Operator Lifecycle Manager(OLM)在 OpenShift Container Platform 4.9 上安装 Data Protection(OADP)Operator 的 OpenShift API。

OADP Operator 会安装 [Velero 1.7](#)。

先决条件

- 您必须以具有 **cluster-admin** 权限的用户身份登录。

流程

1. 在 OpenShift Container Platform Web 控制台中，点击 **Operators** → **OperatorHub**。
2. 使用 **Filter by keyword** 字段查找 **OADP Operator**。
3. 选择 **OADP Operator** 并点 **Install**。
4. 点 **Install** 在 **openshift-adp** 项目中安装 Operator。
5. 点 **Operators** → **Installed Operators** 来验证安装。

4.3.4.2. 配置 Google Cloud Platform

对于数据保护(OADP)，您可以为 OpenShift API 配置 Google Cloud Platform(GCP)。

先决条件

- 您必须安装了 **gcloud** 和 **gsutil** CLI 工具。详情请查看 [Google 云文档](#)。

流程

1. 登录到 GCP:

```
$ gcloud auth login
```

2. 设置 **BUCKET** 变量 :

```
$ BUCKET=<bucket> 1
```

1 指定存储桶名称。

3. 创建存储桶 :

```
$ gsutil mb gs://$BUCKET/
```

4. 将 **PROJECT_ID** 变量设置为您的活跃项目 :

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. 创建服务帐户 :

```
$ gcloud iam service-accounts create velero \  
--display-name "Velero service account"
```

6. 列出服务帐户 :

```
$ gcloud iam service-accounts list
```

7. 设置 **SERVICE_ACCOUNT_EMAIL** 变量，使其与 **email** 值匹配 :

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
  --filter="displayName:Velero service account" \
  --format 'value(email)')
```

8. 附加策略，为 **velero** 用户提供所需的最低权限：

```
$ ROLE_PERMISSIONS=(
  compute.disks.get
  compute.disks.create
  compute.disks.createSnapshot
  compute.snapshots.get
  compute.snapshots.create
  compute.snapshots.useReadOnly
  compute.snapshots.delete
  compute.zones.get
)
```

9. 创建 **velero.server** 自定义角色：

```
$ gcloud iam roles create velero.server \
  --project $PROJECT_ID \
  --title "Velero Server" \
  --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"
```

10. 为项目添加 IAM 策略绑定：

```
$ gcloud projects add-iam-policy-binding $PROJECT_ID \
  --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
  --role projects/$PROJECT_ID/roles/velero.server
```

11. 更新 IAM 服务帐户：

```
$ gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

12. 将 IAM 服务帐户的密钥保存到当前目录中的 **credentials-velero** 文件中：

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

在安装 Data Protection Application 前，您可以使用 **credentials-velero** 文件为 GCP 创建 **Secret** 对象。

4.3.4.3. 创建用于备份和快照位置的 secret

如果使用相同的凭证，您可以为备份和快照位置创建一个 **Secret** 对象。

Secret 的默认名称为 **cloud-credentials-gcp**。

先决条件

- 对象存储和云存储必须使用相同的凭证。
- 您必须为 Velero 配置对象存储。

- 您必须以适当的格式为对象存储创建一个 **credentials-velero** 文件。

流程

- 使用默认名称创建 **Secret** :

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

在安装 Data Protection Application 时，**secret** 会在 **DataProtectionApplication** CR 的 **spec.backupLocations.credential** 块中引用。

4.3.4.3.1. 为不同的备份和恢复位置凭证配置 secret

如果您的备份和恢复位置使用不同的凭证，请创建两个 **Secret** 对象：

- 具有自定义名称的备份位置 **Secret**。自定义名称在 **DataProtectionApplication** 自定义资源(CR) 的 **spec.backupLocations** 块中指定。
- 带有默认名称 **cloud-credentials-gcp** 的快照位置 **Secret**。此 **Secret** 不在 **DataProtectionApplication** CR 中指定。

流程

1. 为您的云供应商为快照位置创建一个 **credentials-velero** 文件。
2. 使用默认名称为快照位置创建 **Secret** :

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

3. 为您的对象存储创建一个用于备份位置的 **credentials-velero** 文件。
4. 使用自定义名称为备份位置创建 **Secret** :

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-
velero
```

5. 将带有自定义名称的 **Secret** 添加到 **DataProtectionApplication** CR 中，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
      provider: gcp
      default: true
      credential:
        key: cloud
        name: <custom_secret> 1
```

```

objectStorage:
  bucket: <bucket_name>
  prefix: <prefix>
snapshotLocations:
- velero:
  provider: gcp
  default: true
  config:
    project: <project>
    snapshotLocation: us-west1

```

- 1 具有自定义名称的备份位置 **Secret**。

4.3.4.4. 配置数据保护应用程序

您可以配置 Velero 资源分配并启用自签名 CA 证书。

4.3.4.4.1. 设置 Velero CPU 和内存分配

您可以通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为 **Velero** pod 设置 CPU 和内存分配。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.configuration.velero.podConfig.ResourceAllocations** 块中的值，如下例所示：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi

```

- 1 指定要向 Velero podSpec 提供的节点选择器

4.3.4.4.2. 启用自签名 CA 证书

您必须通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为对象存储启用自签名 CA 证书，以防止由未知颁发机构签名的证书。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.backupLocations.velero.objectStorage.caCert** 参数和 **spec.backupLocations.velero.config** 参数：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> ①
        config:
          insecureSkipTLSVerify: "false" ②
  ...
```

① 指定 Base46 编码的 CA 证书字符串。

② **insecureSkipTLSVerify** 配置可以设置为 **"true"** 或 **"false"**。如果设置为 **"true"**，则禁用 SSL/TLS 安全性。如果设置为 **"false"**，则启用 SSL/TLS 安全性。

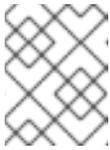
4.3.4.5. 安装数据保护应用程序

您可以通过创建 **DataProtectionApplication** API 的实例来安装数据保护应用程序(DPA)。

先决条件

- 您必须安装 OADP Operator。
- 您必须将对象存储配置为备份位置。
- 如果使用快照来备份 PV，云供应商必须支持原生快照 API 或 Container Storage Interface(CSI) 快照。
- 如果备份和快照位置使用相同的凭证，您必须创建带有默认名称 **cloud-credentials-gcp** 的 **Secret**。
- 如果备份和快照位置使用不同的凭证，您必须创建两个 **Secret**：

- 带有备份位置的自定义名称的 **secret**。您可以将此 **Secret** 添加到 **DataProtectionApplication** CR 中。
- 带有默认名称 **cloud-credentials-gcp** 的 **secret**，用于快照位置。这个 **Secret** 不在 **DataProtectionApplication** CR 中被引用。



注意

如果您不想在安装过程中指定备份或快照位置，您可以使用空 **credentials-velero** 文件创建默认 **Secret**。如果没有默认 **Secret**，安装将失败。

流程

1. 点 **Operators** → **Installed Operators** 并选择 **OADP Operator**。
2. 在 **Provided APIs** 下，点 **DataProtectionApplication** 框中的 **Create 实例**。
3. 点 **YAML View** 并更新 **DataProtectionApplication** 清单的参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - gcp
        - openshift ❶
    restic:
      enable: true ❷
    podConfig:
      nodeSelector: <node selector> ❸
  backupLocations:
    - velero:
        provider: gcp
        default: true
        credential:
          key: cloud
          name: cloud-credentials-gcp ❹
        objectStorage:
          bucket: <bucket_name> ❺
          prefix: <prefix> ❻
  snapshotLocations: ❼
    - velero:
        provider: gcp
        default: true
        config:
          project: <project>
          snapshotLocation: us-west1 ❽

```

- ❶ **openshift** 插件是必须的，才能在 OpenShift Container Platform 集群上备份和恢复命名空间。

- 2 如果要禁用 Restic 安装，设置为 **false**。Restic 部署一个守护进程集，这意味着每个 worker 节点都运行有 **Restic** pod。您可以通过在 **Backup** CR 中添加
- 3 指定要提供给 Restic podSpec 的节点选择器。
- 4 如果没有指定这个值，则使用默认值 **cloud-credentials-gcp**。如果您指定了自定义名称，则使用自定义名称进行备份位置。
- 5 指定存储桶作为备份存储位置。如果存储桶不是 Velero 备份的专用存储桶，您必须指定一个前缀。
- 6 如果存储桶用于多个目的，请为 Velero 备份指定一个前缀，如 **velero**。
- 7 如果您使用 CSI 快照或 Restic 备份 PV，则不需要指定快照位置。
- 8 快照位置必须与 PV 位于同一区域。

4. 点 **Create**。

5. 通过查看 OADP 资源来验证安装：

```
$ oc get all -n openshift-adp
```

输出示例

```
NAME                                READY STATUS RESTARTS AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running 0      2m8s
pod/restic-9cq4q                               1/1   Running 0      94s
pod/restic-m4lts                               1/1   Running 0      94s
pod/restic-pv4kr                               1/1   Running 0      95s
pod/velero-588db7f655-n842v                  1/1   Running 0      95s

NAME                                TYPE          CLUSTER-IP      EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>    8443/TCP  2m8s

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/restic  3        3        3      3           3          <none>    96s

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1           1          2m9s
deployment.apps/velero                          1/1    1           1          96s

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47  1        1        1      2m9s
replicaset.apps/velero-588db7f655                        1        1        1      96s
```

4.3.4.5.1. 在 DataProtectionApplication CR 中启用 CSI

您可以在 **DataProtectionApplication** 自定义资源(CR)中启用 Container Storage Interface(CSI)来备份持久性卷，以使用 CSI 快照备份持久性卷。

元设计

- 云供应商必须支持 CSI 快照。

流程

- 编辑 **DataProtectionApplication** CR，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi 1
```

- 1** 添加 **csi** 默认插件。

4.3.5. 安装和配置 OpenShift API 以进行 Google Cloud Platform 的数据保护

您可以通过安装 OADP Operator、创建 **Secret** 对象，然后安装数据保护应用程序，安装带有 Multicloud Object Gateway(MCG)的数据保护(OADP)的 OpenShift API。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。

MCG 是 OpenShift Container Storage(OCS)的一个组件。您可以将 MCG 配置为 **DataProtectionApplication** 自定义资源(CR)中的备份位置。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

如果您的云供应商具有原生快照 API，请配置快照位置。如果您的云供应商不支持快照，或者存储是 NFS，您可以使用 Restic 创建备份。

您不需要在 Restic 或 Container Storage Interface(CSI)快照的 **DataProtectionApplication** CR 中指定快照位置。

要在受限网络环境中安装 OADP Operator，您必须首先禁用默认的 OperatorHub 源并镜像 Operator 目录。详情请参阅[在受限网络中使用 Operator Lifecycle Manager](#)。

4.3.5.1. 安装 OADP Operator

您可以使用 Operator Lifecycle Manager(OLM)在 OpenShift Container Platform 4.9 上安装 Data Protection(OADP)Operator 的 OpenShift API。

OADP Operator 会安装 [Velero 1.7](#)。

先决条件

- 您必须以具有 **cluster-admin** 权限的用户身份登录。

流程

1. 在 OpenShift Container Platform Web 控制台中，点击 **Operators → OperatorHub**。
2. 使用 **Filter by keyword** 字段查找 **OADP Operator**。
3. 选择 **OADP Operator** 并点 **Install**。
4. 点 **Install** 在 **openshift-adp** 项目中安装 Operator。
5. 点 **Operators → Installed Operators** 来验证安装。

4.3.5.2. 检索多云对象网关凭证

您必须检索 Multicloud Object Gateway(MCG)凭证，以便为 OpenShift API 创建用于数据保护(OADP)的 **Secret** 自定义资源(CR)。

MCG 是 OpenShift Container Storage 的一个组件。

先决条件

- 您需要根据 [OpenShift Container Storage 部署指南](#) 来部署 OpenShift Container Storage。

流程

1. 通过对 **NooBaa** 自定义资源运行 **describe** 命令，获取 S3 端点、**AWS_ACCESS_KEY_ID** 和 **AWS_SECRET_ACCESS_KEY**。
2. 创建 **credentials-velero** 文件：

```
$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

在安装 Data Protection Application 时，您可以使用 **credentials-velero** 文件创建 **Secret** 对象。

4.3.5.3. 创建用于备份和快照位置的 secret

如果使用相同的凭证，您可以为备份和快照位置创建一个 **Secret** 对象。

Secret 的默认名称为 **cloud-credentials**。

先决条件

- 对象存储和云存储必须使用相同的凭证。
- 您必须为 Velero 配置对象存储。
- 您必须以适当的格式为对象存储创建一个 **credentials-velero** 文件。

流程

- 使用默认名称创建 **Secret** :

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

在安装 Data Protection Application 时，**secret** 会在 **DataProtectionApplication** CR 的 **spec.backupLocations.credential** 块中引用。

4.3.5.3.1. 为不同的备份和恢复位置凭证配置 secret

如果您的备份和恢复位置使用不同的凭证，请创建两个 **Secret** 对象：

- 具有自定义名称的备份位置 **Secret**。自定义名称在 **DataProtectionApplication** 自定义资源(CR) 的 **spec.backupLocations** 块中指定。
- 带有默认名称 **cloud-credentials** 的快照位置 **Secret**。此 **Secret** 不在 **DataProtectionApplication** CR 中指定。

流程

1. 为您的云供应商为快照位置创建一个 **credentials-velero** 文件。
2. 使用默认名称为快照位置创建 **Secret** :

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

3. 为您的对象存储创建一个用于备份位置的 **credentials-velero** 文件。
4. 使用自定义名称为备份位置创建 **Secret** :

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. 将带有自定义名称的 **Secret** 添加到 **DataProtectionApplication** CR 中，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - aws
```

```

- openshift
restic:
  enable: true
backupLocations:
- velero:
  config:
    profile: "default"
    region: minio
    s3Url: <url>
    insecureSkipTLSVerify: "true"
    s3ForcePathStyle: "true"
  provider: aws
  default: true
  credential:
    key: cloud
    name: <custom_secret> ❶
  objectStorage:
    bucket: <bucket_name>
    prefix: <prefix>

```

❶ 具有自定义名称的备份位置 **Secret**。

4.3.5.4. 配置数据保护应用程序

您可以配置 Velero 资源分配并启用自签名 CA 证书。

4.3.5.4.1. 设置 Velero CPU 和内存分配

您可以通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为 **Velero** pod 设置 CPU 和内存分配。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.configuration.velero.podConfig.ResourceAllocations** 块中的值，如下例所示：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> ❶
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi

```

```
requests:
  cpu: 500m
  memory: 256Mi
```

- 1 指定要向 Velero podSpec 提供的节点选择器

4.3.5.4.2. 启用自签名 CA 证书

您必须通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为对象存储启用自签名 CA 证书，以防止由未知颁发机构签名的证书。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.backupLocations.velero.objectStorage.caCert** 参数和 **spec.backupLocations.velero.config** 参数：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
  ...
```

- 1 指定 Base46 编码的 CA 证书字符串。

- 2 **insecureSkipTLSVerify** 配置可以设置为 **"true"** 或 **"false"**。如果设置为 **"true"**，则禁用 SSL/TLS 安全性。如果设置为 **"false"**，则启用 SSL/TLS 安全性。

4.3.5.5. 安装数据保护应用程序

您可以通过创建 **DataProtectionApplication** API 的实例来安装数据保护应用程序(DPA)。

先决条件

- 您必须安装 OADP Operator。

- 您必须将对象存储配置为备份位置。
- 如果使用快照来备份 PV，云供应商必须支持原生快照 API 或 Container Storage Interface(CSI) 快照。
- 如果备份和快照位置使用相同的凭证，您必须创建带有默认名称 **cloud-credentials** 的 **Secret**。
- 如果备份和快照位置使用不同的凭证，您必须创建两个 **Secret** :
 - 带有备份位置的自定义名称的 **secret**。您可以将此 **Secret** 添加到 **DataProtectionApplication** CR 中。
 - 带有默认名称 **cloud-credentials** 的 **secret**，用于快照位置。这个 **Secret** 没有在 **DataProtectionApplication** CR 中被引用。



注意

如果您不想在安装过程中指定备份或快照位置，您可以使用空 **credentials-velero** 文件创建默认 **Secret**。如果没有默认 **Secret**，安装将失败。

流程

1. 点 **Operators** → **Installed Operators** 并选择 **OADP Operator**。
2. 在 **Provided APIs** 下，点 **DataProtectionApplication** 框中的 **Create 实例**。
3. 点 **YAML View** 并更新 **DataProtectionApplication** 清单的参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - aws
        - openshift ①
    restic:
      enable: true ②
    podConfig:
      nodeSelector: <node selector> ③
  backupLocations:
    - velero:
      config:
        profile: "default"
        region: minio
        s3Url: <url> ④
        insecureSkipTLSVerify: "true"
        s3ForcePathStyle: "true"
      provider: aws
      default: true
      credential:
        key: cloud
  
```

```

name: cloud-credentials 5
objectStorage:
  bucket: <bucket_name> 6
  prefix: <prefix> 7

```

- 1** **openshift** 插件是必须的，才能在 OpenShift Container Platform 集群上备份和恢复命名空间。
- 2** 如果要禁用 Restic 安装，设置为 **false**。Restic 部署一个守护进程集，这意味着每个 worker 节点都运行有 **Restic** pod。您可以通过在 **Backup** CR 中添加 **spec.defaultVolumesToRestic: true** 来配置 Restic 以进行备份。
- 3** 指定要提供给 Restic podSpec 的节点选择器。
- 4** 指定 S3 端点的 URL。
- 5** 如果没有指定这个值，则使用默认值 **cloud-credentials**。如果您指定了自定义名称，则使用自定义名称进行备份位置。
- 6** 指定存储桶作为备份存储位置。如果存储桶不是 Velero 备份的专用存储桶，您必须指定一个前缀。
- 7** 如果存储桶用于多个目的，请为 Velero 备份指定一个前缀，如 **velero**。

4. 点 Create。

5. 通过查看 OADP 资源来验证安装：

```
$ oc get all -n openshift-adp
```

输出示例

```

NAME                                READY STATUS RESTARTS AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running 0      2m8s
pod/restic-9cq4q                                1/1   Running 0      94s
pod/restic-m4lts                                1/1   Running 0      94s
pod/restic-pv4kr                                1/1   Running 0      95s
pod/velero-588db7f655-n842v                    1/1   Running 0      95s

```

```

NAME                                TYPE          CLUSTER-IP      EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>    8443/TCP    2m8s

```

```

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/restic  3        3        3      3           3          <none>    96s

```

```

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1            1          2m9s
deployment.apps/velero                          1/1    1            1          96s

```

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47	1	1	1	2m9s
replicaset.apps/velero-588db7f655	1	1	1	96s

4.3.5.5.1. 在 DataProtectionApplication CR 中启用 CSI

您可以在 **DataProtectionApplication** 自定义资源(CR)中启用 Container Storage Interface(CSI)来备份持久性卷，以使用 CSI 快照备份持久性卷。

先决条件

- 云供应商必须支持 CSI 快照。

流程

- 编辑 **DataProtectionApplication** CR，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi 1
```

- 添加 **csi** 默认插件。

4.3.6. 为 OpenShift Container Storage 的数据保护安装和配置 OpenShift API

您可以通过安装 OADP Operator 并配置备份位置和快照位置，通过安装 OpenShift Container Storage(OCS)安装 OpenShift API for Data Protection(OADP)。然后，您要安装数据保护应用程序。



注意

从 OADP 1.0.4 开始，所有 OADP 1.0.z 版本都只能用作 MTC Operator 的依赖项，且不适用于独立 Operator。

您可以在 **DataProtectionApplication** 自定义资源(CR)中将 [Multicloud Object Gateway](#) 或任何 S3 兼容对象存储配置为备份位置。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

如果云供应商具有原生快照 API，您可以在 **DataProtectionApplication** CR 中将云存储配置为快照位置。您不需要为 Restic 或 Container Storage Interface(CSI)快照指定快照。

要在受限网络环境中安装 OADP Operator，您必须首先禁用默认的 OperatorHub 源并镜像 Operator 目录。详情请参阅[在受限网络中使用 Operator Lifecycle Manager](#)。

4.3.6.1. 安装 OADP Operator

您可以使用 Operator Lifecycle Manager(OLM)在 OpenShift Container Platform 4.9 上安装 Data Protection(OADP)Operator 的 OpenShift API。

OADP Operator 会安装 [Velero 1.7](#)。

先决条件

- 您必须以具有 **cluster-admin** 权限的用户身份登录。

流程

1. 在 OpenShift Container Platform Web 控制台中，点击 **Operators → OperatorHub**。
2. 使用 **Filter by keyword** 字段查找 **OADP Operator**。
3. 选择 **OADP Operator** 并点 **Install**。
4. 点 **Install** 在 **openshift-adp** 项目中安装 Operator。
5. 点 **Operators → Installed Operators** 来验证安装。



注意

安装 OADP Operator 后，如果云供应商支持原生快照 API，将对象存储配置为备份位置，云存储为快照位置。

如果云供应商不支持快照，或者存储是 NFS，您可以使用 [Restic](#) 创建备份。Restic 不需要快照位置。

4.3.6.2. 创建用于备份和快照位置的 secret

如果使用相同的凭证，您可以为备份和快照位置创建一个 **Secret** 对象。

先决条件

- 对象存储和云存储必须使用相同的凭证。
- 您必须为 Velero 配置对象存储。
- 您必须以适当的格式为对象存储创建一个 **credentials-velero** 文件。

流程

- 使用默认名称创建 **Secret**：

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

在安装 Data Protection Application 时，**secret** 会在 **DataProtectionApplication** CR 的 **spec.backupLocations.credential** 块中引用。

4.3.6.2.1. 为不同的备份和恢复位置凭证配置 secret

4.3.6.3. 配置数据保护应用程序

您可以配置 Velero 资源分配并启用自签名 CA 证书。

4.3.6.3.1. 设置 Velero CPU 和内存分配

您可以通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为 **Velero** pod 设置 CPU 和内存分配。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.configuration.velero.podConfig.ResourceAllocations** 块中的值，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

- 1** 指定要向 Velero podSpec 提供的节点选择器

4.3.6.3.2. 启用自签名 CA 证书

您必须通过编辑 **DataProtectionApplication** 自定义资源(CR)清单来为对象存储启用自签名 CA 证书，以防止由未知颁发机构签名的证书。

先决条件

- 您必须安装了 OpenShift API for Data Protection(OADP)Operator。

流程

- 编辑 **DataProtectionApplication** CR 清单的 **spec.backupLocations.velero.objectStorage.caCert** 参数和 **spec.backupLocations.velero.config** 参数：

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> ❶
        config:
          insecureSkipTLSVerify: "false" ❷
  ...

```

- ❶ 指定 Base46 编码的 CA 证书字符串。
- ❷ **insecureSkipTLSVerify** 配置可以设置为 **"true"** 或 **"false"**。如果设置为 **"true"**，则禁用 SSL/TLS 安全性。如果设置为 **"false"**，则启用 SSL/TLS 安全性。

4.3.6.4. 安装数据保护应用程序

您可以通过创建 **DataProtectionApplication** API 的实例来安装数据保护应用程序(DPA)。

先决条件

- 您必须安装 OADP Operator。
- 您必须将对象存储配置为备份位置。
- 如果使用快照来备份 PV，云供应商必须支持原生快照 API 或 Container Storage Interface(CSI) 快照。
- 如果备份和快照位置使用相同的凭证，您必须创建带有默认名称 **cloud-credentials** 的 **Secret**。



注意

如果您不想在安装过程中指定备份或快照位置，您可以使用空 **credentials-velero** 文件创建默认 **Secret**。如果没有默认 **Secret**，安装将失败。

流程

1. 点 **Operators** → **Installed Operators** 并选择 OADP Operator。
2. 在 **Provided APIs** 下，点 **DataProtectionApplication** 框中的 **Create 实例**。

3. 点 **YAML View** 并更新 **DataProtectionApplication** 清单的参数：
4. 点 **Create**。
5. 通过查看 OADP 资源来验证安装：

```
$ oc get all -n openshift-adp
```

输出示例

```

NAME                                READY STATUS  RESTARTS  AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running  0         2m8s
pod/restic-9cq4q                                1/1   Running  0         94s
pod/restic-m4lts                                1/1   Running  0         94s
pod/restic-pv4kr                                1/1   Running  0         95s
pod/velero-588db7f655-n842v                    1/1   Running  0         95s

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>      8443/TCP    2m8s

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/restic  3        3        3      3           3          <none>    96s

NAME          READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1           1          2m9s
deployment.apps/velero                          1/1    1           1          96s

NAME          DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47  1      1      1      2m9s
replicaset.apps/velero-588db7f655                          1      1      1      96s

```

4.3.6.4.1. 配置 NooBaa 以便在 OpenShift Container Storage 上进行灾难恢复

如果您在 OpenShift Container Storage 上为 NooBaa bucket **backupStorageLocation** 使用集群存储，请将 NooBaa 配置为外部对象存储。



警告

将 NooBaa 配置为外部对象存储可能会导致备份不可用。

流程

- 将 NooBaa 配置为外部对象存储，如[为混合或多云添加存储资源](#)中所述。

4.3.6.4.2. 在 DataProtectionApplication CR 中启用 CSI

您可以在 **DataProtectionApplication** 自定义资源(CR)中启用 Container Storage Interface(CSI)来备份持久性卷，以使用 CSI 快照备份持久性卷。

先决条件

- 云供应商必须支持 CSI 快照。

流程

- 编辑 **DataProtectionApplication** CR，如下例所示：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi 1
```

- 1** 添加 **csi** 默认插件。

4.3.7. 为数据保护卸载 OpenShift API

您可以通过删除 OADP Operator 来卸载 OpenShift API for Data Protection(OADP)。详情请参阅[从集群中删除 Operator](#)。

4.4. 备份和恢复

4.4.1. 备份应用程序

您可以通过创建 **Backup** 自定义资源(CR) 来备份应用程序。

Backup CR 为 Kubernetes 资源和内部镜像 (S3 对象存储) 和持久性卷(PV)创建备份文件，如果云供应商使用原生快照 API 或 [Container Storage Interface\(CSI\)](#) 来创建快照，如 OpenShift Container Storage 4。如需更多信息，请参阅 [CSI 卷快照](#)。



重要

S3 存储的 **CloudStorage** API 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

如果您的云供应商具有原生快照 API 或支持 [Container Storage Interface\(CSI\)快照](#)，则 **Backup** CR 通过创建快照来备份持久性卷。如需更多信息，请参阅 OpenShift Container Platform 文档中的 [CSI 卷快照概述](#)。

如果您的云供应商不支持快照，或者应用程序位于 NFS 数据卷中，您可以使用 [Restic](#) 创建备份。

您可以创建 [backup hook](#)，以便在备份操作之前或之后运行命令。

您可以通过创建一个 **Schedule CR** 而不是 **Backup CR** 来调度备份。

4.4.1.1. 创建备份 CR

您可以通过创建 **Backup** 备份自定义资源(CR)来备份 Kubernetes 镜像、内部镜像和持久性卷(PV)。

先决条件

- 您必须安装用于数据保护(OADP)Operator 的 OpenShift API。
- **DataProtectionApplication** CR 必须处于 **Ready** 状态。
- 备份位置先决条件：
 - 您必须为 Velero 配置 S3 对象存储。
 - 您必须在 **DataProtectionApplication** CR 中配置了一个备份位置。
- 快照位置先决条件：
 - 您的云供应商必须具有原生快照 API 或支持 Container Storage Interface(CSI)快照。
 - 对于 CSI 快照，您必须创建一个 **VolumeSnapshotClass** CR 来注册 CSI 驱动程序。
 - 您必须在 **DataProtectionApplication** CR 中配置了一个卷位置。

流程

1. 输入以下命令来检索 **backupStorageLocations** CR：

```
$ oc get backupStorageLocations
```

输出示例

```
NAME           PHASE    LAST VALIDATED  AGE  DEFAULT
velero-sample-1 Available  11s             31m
```

2. 创建一个 **Backup** CR，如下例所示：

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  hooks: {}
  includedNamespaces:
  - <namespace> 1
  includedResources: [] 2
  excludedResources: [] 3
  storageLocation: <velero-sample-1> 4
  ttl: 720h0m0s
```

```

labelSelector: 5
- matchLabels:
  app=<label_1>
- matchLabels:
  app=<label_2>
- matchLabels:
  app=<label_3>
orlabelSelectors: 6
- matchLabels:
  app=<label_1>
- matchLabels:
  app=<label_2>
- matchLabels:
  app=<label_3>

```

- 1 指定要备份的命名空间数组。
- 2 可选：指定一个要包含在备份中的资源的数组。资源可以是缩写方式（例如，'po' 代表 'pods'）或完全限定的方式。如果未指定，则会包含所有资源。
- 3 可选：指定要从备份中排除的资源数组。资源可以是缩写方式（例如，'po' 代表 'pods'）或完全限定的方式。
- 4 指定 **backupStorageLocations** CR 的名称。
- 5 备份具有所有指定标签的资源。
- 6 备份具有一个或多个指定标签的资源。

3. 验证 **Backup** CR 的状态是否为 **Completed**：

```
$ oc get backup -n openshift-adp <backup> -o jsonpath='{.status.phase}'
```

4.4.1.2. 使用 CSI 快照备份持久性卷

在创建 **Backup** CR 前，您可以通过创建 **VolumeSnapshotClass** 自定义资源(CR)来注册 CSI 驱动程序，使用 Container Storage Interface(CSI)快照备份持久性卷。

先决条件

- 云供应商必须支持 CSI 快照。
- 您必须在 **DataProtectionApplication** CR 中启用 CSI。

流程

- 创建 **VolumeSnapshotClass** CR，如下例所示：

Ceph RBD

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
deletionPolicy: Retain
metadata:

```

```

name: <volume_snapshot_class_name>
labels:
  velero.io/csi-volumesnapshot-class: "true"
  snapshotter: openshift-storage.rbd.csi.ceph.com
driver: openshift-storage.rbd.csi.ceph.com
parameters:
  clusterID: openshift-storage
  csi.storage.k8s.io/snapshotter-secret-name: rook-csi-rbd-provisioner
  csi.storage.k8s.io/snapshotter-secret-namespace: openshift-storage

```

Ceph FS

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: <volume_snapshot_class_name>
  labels:
    velero.io/csi-volumesnapshot-class: "true"
driver: openshift-storage.cephfs.csi.ceph.com
deletionPolicy: Retain
parameters:
  clusterID: openshift-storage
  csi.storage.k8s.io/snapshotter-secret-name: rook-csi-cephfs-provisioner
  csi.storage.k8s.io/snapshotter-secret-namespace: openshift-storage

```

其他云供应商

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: <volume_snapshot_class_name>
  labels:
    velero.io/csi-volumesnapshot-class: "true"
driver: <csi_driver>
deletionPolicy: Retain

```

现在，您可以创建一个 **Backup** CR。

4.4.1.3. 使用 Restic 备份应用程序

您可以通过编辑备份自定义资源(CR)来使用 Restic **Backup** 资源、内部镜像和持久性卷备份 Kubernetes 资源。

您不需要在 **DataProtectionApplication** CR 中指定快照位置。



重要

Restic 不支持备份 **hostPath** 卷。如需更多信息，请参阅[额外的 Rustic 限制](#)。

先决条件

- 您必须安装用于数据保护(OADP)Operator 的 OpenShift API。

- 您不能将 **DataProtectionApplication** CR 中的 **spec.configuration.restic.enable** 设置为 **false** 来禁用默认的 Restic 安装。
- **DataProtectionApplication** CR 必须处于 **Ready** 状态。

流程

- 编辑 **Backup** CR，如下例所示：

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  defaultVolumesToRestic: true 1
  ...
```

- 1** 将 **defaultVolumesToRestic: true** 添加到 **spec** 块中。

4.4.1.4. 对 CSI 快照使用 Data Mover



重要

CSI 快照的数据仅是一项技术预览功能。技术预览功能不受红帽产品服务等级协议 (SLA) 支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

OADP 11.0 数据管理可让客户将容器存储接口 (CSI) 卷快照备份到远程对象存储。启用 Data Mover 时，如果出现问题、意外删除或损坏集群，您可以从存储中恢复有状态的应用程序。OADP 11.0 Data Mover 解决方案使用 VolSync 的 Restic 选项。



注意

数据 Mover 支持 CSI 卷快照的备份和恢复。

目前，Data Mover 不支持 Google Cloud Storage (GCS) 存储桶。

先决条件

- 已确认 **StorageClass** 和 **VolumeSnapshotClass** 自定义资源 (CR) 支持 CSI。
- 您已确认只有一个 **volumeSnapshotClass** CR 带有注解 **snapshot.storage.kubernetes.io/is-default-class: true**。
- 已确认只有一个 **storageClass** CR 带有注解 **storageclass.kubernetes.io/is-default-class: true**。
- 您已在 **VolumeSnapshotClass** CR 中包含标签 **velero.io/csi-volumesnapshot-class: 'true'**。

- 已使用 Operator Lifecycle Manager (OLM) 安装 VolSync Operator。



注意

VolSync Operator 只需要与技术预览数据一起使用。使用 OADP 的生产环境功能不需要 Operator。

- 已使用 OLM 安装 OADP operator。

流程

1. 通过创建一个 **.yaml** 文件来配置 Restic secret，如下所示：

```
apiVersion: v1
kind: Secret
metadata:
  name: <secret_name>
  namespace: openshift-adp
type: Opaque
stringData:
  RESTIC_PASSWORD: <secure_restic_password>
```



注意

默认情况下，Operator 会查找名为 **dm-credential** 的 secret。如果您使用其他名称，您需要使用 **dpa.spec.features.dataMover.credentialName** 通过 Data Protection Application (DPA) CR 指定名称。

2. 创建类似以下示例的 DPA CR。默认插件包括 CSI。

数据保护应用程序 (DPA) CR 示例

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: velero-sample
  namespace: openshift-adp
spec:
  features:
    dataMover:
      enable: true
      credentialName: <secret_name> 1
  backupLocations:
    - velero:
        config:
          profile: default
          region: us-east-1
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
```

```

    prefix: <bucket_prefix>
    provider: aws
  configuration:
    restic:
      enable: <true_or_false>
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi

```

- 1 添加上一步中的 Restic secret 名称。如果没有这样做，则使用默认 secret 名称 **dm-credential**。

OADP Operator 安装两个自定义资源定义 (CRD)、**VolumeSnapshotBackup** 和 **VolumeSnapshotRestore**。

VolumeSnapshotBackup CRD 示例

```

apiVersion: datamover.oadp.openshift.io/v1alpha1
kind: VolumeSnapshotBackup
metadata:
  name: <vsb_name>
  namespace: <namespace_name> 1
spec:
  volumeSnapshotContent:
    name: <snapcontent_name>
  protectedNamespace: <adp_namespace>
  resticSecretRef:
    name: <restic_secret_name>

```

- 1 指定卷快照所在的命名空间。

VolumeSnapshotRestore CRD 示例

```

apiVersion: datamover.oadp.openshift.io/v1alpha1
kind: VolumeSnapshotRestore
metadata:
  name: <vsr_name>
  namespace: <namespace_name> 1
spec:
  protectedNamespace: <protected_ns> 2
  resticSecretRef:
    name: <restic_secret_name>
  volumeSnapshotMoverBackupRef:
    sourcePVCData:
      name: <source_pvc_name>
      size: <source_pvc_size>
    resticrepository: <your_restic_repo>
    volumeSnapshotClassName: <vsclass_name>

```

- 1 指定卷快照所在的命名空间。

- 2 指定安装 Operator 的命名空间。默认值为 **openshift-adp**。

3. 您可以执行以下步骤备份卷快照：

- a. 创建备份 CR：

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup_name>
  namespace: <protected_ns> 1
spec:
  includedNamespaces:
  - <app_ns>
  storageLocation: velero-sample-1
```

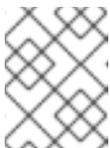
- 1 指定安装 Operator 的命名空间。默认命名空间是 **openshift-adp**。

- b. 等待 10 分钟，并输入以下命令来检查 **VolumeSnapshotBackup** CR 状态是否为 **Completed**：

```
$ oc get vsb -n <app_ns>
```

```
$ oc get vsb <vsb_name> -n <app_ns> -o jsonpath="{.status.phase}"
```

在对象存储中创建快照是在 DPA 中配置。



注意

如果 **VolumeSnapshotBackup** CR 的状态变为 **Failed**，请参阅 Velero 日志进行故障排除。

4. 您可以执行以下步骤来恢复卷快照：

- a. 删除由 Velero CSI 插件创建的 application 命名空间和 **volumeSnapshotContent**。
- b. 创建 **Restore** CR，并将 **restorePV** 设置为 **true**。

Restore CR 示例

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore_name>
  namespace: <protected_ns>
spec:
  backupName: <previous_backup_name>
  restorePVs: true
```

- c. 等待 10 分钟，并通过输入以下命令来检查 **VolumeSnapshotRestore** CR 状态是否为 **Completed**：

```
$ oc get vsr -n <app_ns>
```

```
$ oc get vsr <vsr_name> -n <app_ns> -o jsonpath="{.status.phase}"
```

- d. 检查您的应用程序数据和资源是否已恢复。



注意

如果 **VolumeSnapshotRestore** CR 的状态变成 'Failed', 请参阅 Velero 日志进行故障排除。

其他资源

- [在集群上为管理员安装 Operator](#)
- [在命名空间中为非管理员用户安装 Operator](#)

4.4.1.5. 创建备份 hook

您可以通过编辑备份自定义资源(CR)来创建 **Backup** hook 以在 pod 中运行的容器中运行命令。

在 pod 备份前运行 *Pre* hook。在备份后运行 *Post* hook。

流程

- 在 **Backup** CR 的 **spec.hooks** 块中添加 hook，如下例所示：

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  namespace: openshift-adp
spec:
  hooks:
    resources:
      - name: <hook_name>
        includedNamespaces:
          - <namespace> ①
        excludedNamespaces: ②
          - <namespace>
        includedResources: []
        - pods ③
        excludedResources: [] ④
        labelSelector: ⑤
          matchLabels:
            app: velero
            component: server
        pre: ⑥
          - exec:
              container: <container> ⑦
              command:
                - /bin/uname ⑧
                - -a
```

```

    onError: Fail 9
    timeout: 30s 10
    post: 11
  ...

```

- 1** 可选：您可以指定 hook 应用的命名空间。如果没有指定这个值，则 hook 适用于所有命名空间。
- 2** 可选：您可以指定 hook 不应用到的命名空间。
- 3** 目前，pod 是唯一可以应用 hook 的支持的资源。
- 4** 可选：您可以指定 hook 不应用到的资源。
- 5** 可选：此 hook 仅适用于与标签匹配的对象。如果没有指定这个值，则 hook 适用于所有命名空间。
- 6** 备份前要运行的 hook 数组。
- 7** 可选：如果没有指定容器，该命令将在 pod 的第一个容器中运行。
- 8** 这是正在添加的 init 容器的入口点。
- 9** 错误处理允许的值是 **Fail** 和 **Continue**。默认值为 **Fail**。
- 10** 可选：等待命令运行的时间。默认值为 **30s**。
- 11** 此块定义了备份后运行的一组 hook，其参数与 pre-backup hook 相同。

4.4.1.6. 调度备份

您可以通过创建 **Schedule** 自定义资源(CR)而不是 **Backup** CR 来调度备份。



警告

在您的备份调度中留有足够的时间，以便在创建另一个备份前完成了当前的备份。

例如，如果对一个命名空间进行备份通常需要 10 分钟才能完成，则调度的备份频率不应该超过每 15 分钟一次。

先决条件

- 您必须安装用于数据保护(OADP)Operator 的 OpenShift API。
- **DataProtectionApplication** CR 必须处于 **Ready** 状态。

流程

1. 检索 **backupStorageLocations** CR：

```
$ oc get backupStorageLocations
```

输出示例

```
NAME           PHASE     LAST VALIDATED  AGE  DEFAULT
velero-sample-1 Available  11s             31m
```

2. 创建一个 **Schedule** CR，如下例所示：

```
$ cat << EOF | oc apply -f -
apiVersion: velero.io/v1
kind: Schedule
metadata:
  name: <schedule>
  namespace: openshift-adp
spec:
  schedule: 0 7 * * * ①
  template:
    hooks: {}
    includedNamespaces:
    - <namespace> ②
    storageLocation: <velero-sample-1> ③
    defaultVolumesToRestic: true ④
    ttl: 720h0m0s
EOF
```

- ① 调度备份的 **cron** 表达式，例如 **0 7 * * *** 代表在每天 7:00 执行备份。
- ② 要备份的命名空间数组。
- ③ **backupStorageLocations** CR 的名称。
- ④ 可选：如果使用 Restic 备份卷，请添加 **defaultVolumesToRestic: true** 键-值对。

3. 在调度的备份运行后验证 **Schedule** CR 的状态是否为 **Completed**：

```
$ oc get schedule -n openshift-adp <schedule> -o jsonpath='{.status.phase}'
```

4.4.1.7. 删除备份

您可以通过删除 **Backup** 自定义资源 (CR) 来删除备份文件。



警告

删除 **Backup** CR 和关联的对象存储数据后，您无法恢复删除的数据。

先决条件

- 您创建了 **Backup** CR。
- 您知道 **Backup** CR 的名称以及包含它的命名空间。
- 下载 Velero CLI 工具。
- 您可以访问集群中的 Velero 二进制文件。

流程

- 选择以下操作之一来删除 **Backup** CR：
 - 要删除 **Backup** CR 并保留关联的对象存储数据，请运行以下命令：

```
$ oc delete backup <backup_CR_name> -n <velero_namespace>
```
 - 要删除 **Backup** CR 并删除关联的对象存储数据，请运行以下命令：

```
$ velero backup delete <backup_CR_name> -n <velero_namespace>
```

其中：

<backup_CR_name>

指定 **Backup** 自定义资源的名称。

<velero_namespace>

指定包含 **Backup** 自定义资源的命名空间。

其他资源

- [下载 Velero CLI 工具](#)

4.4.2. 恢复应用程序

您可以通过创建 [Restore 自定义资源\(CR\)](#) 来恢复应用程序备份。

您可以创建 [restore hooks](#)，以在应用程序容器启动前或在应用程序容器本身中运行命令。

4.4.2.1. 创建恢复 CR

您可以通过创建一个 **Restore** CR 来恢复 **Backup** 自定义资源(CR)。

先决条件

- 您必须安装用于数据保护(OADP)Operator 的 OpenShift API。
- **DataProtectionApplication** CR 必须处于 **Ready** 状态。
- 您必须具有 Velero **Backup** CR。
- 调整请求的大小，以便持久性卷 (PV) 容量与备份时请求的大小匹配。

流程

1. 创建一个 **Restore** CR，如下例所示：

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore>
  namespace: openshift-adp
spec:
  backupName: <backup> 1
  includedResources: [] 2
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - resticrepositories.velero.io
  restorePVs: true

```

1 备份 CR 的名称。

2 可选。指定要包含在恢复过程中的资源数组。资源可以是缩写方式（例如，'po' 代表 'pods'）或完全限定的方式。如果未指定，则会包含所有资源。

2. 输入以下命令验证 **Restore** CR 的状态是否为 **Completed** :

```
$ oc get restore -n openshift-adp <restore> -o jsonpath='{.status.phase}'
```

3. 输入以下命令验证备份资源是否已恢复 :

```
$ oc get all -n <namespace> 1
```

1 备份的命名空间。

4. 如果您使用 Restic 恢复 **DeploymentConfig** 对象，或使用 post-restore hook，请输入以下命令运行 **dc-restic-post-restore.sh** cleanup 脚本 :

```
$ bash dc-restic-post-restore.sh <restore-name>
```



注意

在恢复过程中，OADP Velero 插件会缩减 **DeploymentConfig** 对象，并将 pod 恢复为独立 pod，以防止集群在恢复时立即删除恢复的 **DeploymentConfig** pod，并允许 Restic 和 post-restore hook 在恢复的 pod 上完成其操作。清理脚本会删除这些断开连接的 pod，并将任何 **DeploymentConfig** 对象扩展至适当的副本数。

例 4.1. dc-restic-post-restore.sh cleanup 脚本

```

#!/bin/bash
set -e

# if sha256sum exists, use it to check the integrity of the file
if command -v sha256sum >/dev/null 2>&1; then

```

```

CHECKSUM_CMD="sha256sum"
else
CHECKSUM_CMD="shasum -a 256"
fi

label_name () {
  if [ "${#1}" -le "63" ]; then
echo $1
return
  fi
  sha=$(echo -n $1|$CHECKSUM_CMD)
  echo "${1:0:57}${sha:0:6}"
}

OADP_NAMESPACE=${OADP_NAMESPACE:=openshift-adp}

if [[ $# -ne 1 ]]; then
  echo "usage: ${BASH_SOURCE} restore-name"
  exit 1
fi

echo using OADP Namespace $OADP_NAMESPACE
echo restore: $1

label=$(label_name $1)
echo label: $label

echo Deleting disconnected restore pods
oc delete pods -l oadp.openshift.io/disconnected-from-dc=$label

for dc in $(oc get dc --all-namespaces -l oadp.openshift.io/replicas-modified=$label -o
jsonpath='{range .items[*]}{.metadata.namespace}{","}{.metadata.name}{","}
{.metadata.annotations.oadp\.openshift\.io/original-replicas}{","}
{.metadata.annotations.oadp\.openshift\.io/original-paused}{"\n"}')
do
  IFS=' ' read -ra dc_arr <<< "$dc"
  if [ ${#dc_arr[0]} -gt 0 ]; then
echo Found deployment ${dc_arr[0]}/${dc_arr[1]}, setting replicas: ${dc_arr[2]}, paused:
${dc_arr[3]}
cat <<EOF | oc patch dc -n ${dc_arr[0]} ${dc_arr[1]} --patch-file /dev/stdin
spec:
  replicas: ${dc_arr[2]}
  paused: ${dc_arr[3]}
EOF
  fi
done

```

4.4.2.2. 创建恢复 hook

您可以创建恢复 hook，以便在 pod 中运行的容器运行命令，同时通过编辑 **Restore** 自定义资源(CR)恢复应用程序。

您可以创建两种类型的恢复 hook：

- **init** hook 将 init 容器添加到 pod，以便在应用程序容器启动前执行设置任务。
如果您恢复 Restic 备份，则会在恢复 hook init 容器前添加 **restic-wait** init 容器。
- **exec** hook 在恢复的 pod 的容器中运行命令或脚本。

流程

- 在 **Restore** CR 的 **spec.hooks** 块中添加 hook，如下例所示：

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore>
  namespace: openshift-adp
spec:
  hooks:
    resources:
      - name: <hook_name>
        includedNamespaces:
          - <namespace> ❶
        excludedNamespaces:
          - <namespace>
        includedResources:
          - pods ❷
        excludedResources: []
        labelSelector: ❸
          matchLabels:
            app: velero
            component: server
        postHooks:
          - init:
              initContainers:
                - name: restore-hook-init
                  image: alpine:latest
                  volumeMounts:
                    - mountPath: /restores/pvc1-vm
                      name: pvc1-vm
                  command:
                    - /bin/ash
                    - -c
              timeout: ❹
          - exec:
              container: <container> ❺
              command:
                - /bin/bash ❻
                - -c
                - "psql < /backup/backup.sql"
              waitTimeout: 5m ❼
              execTimeout: 1m ❽
              onError: Continue ❾

```

❶ 可选：hook 应用的命名空间数组。如果没有指定这个值，则 hook 适用于所有命名空间。

❷ 目前，pod 是唯一可以应用 hook 的支持的资源。

- 3 可选：此 hook 仅适用于与标签选择器匹配的对象。
- 4 可选：超时指定了 Velero 等待 `initContainers` 完成的最大时间长度。
- 5 可选：如果没有指定容器，该命令将在 pod 的第一个容器中运行。
- 6 这是正在添加的 init 容器的入口点。
- 7 可选：等待容器就绪的时间。这应该足够长，以便容器可以启动，在相同容器中的任何以前的 hook 可以完成。如果没有设置，恢复过程会无限期等待。
- 8 可选：等待命令运行的时间。默认值为 **30s**。
- 9 错误处理的允许值为 **Fail** 和 **Continue**：
 - **Continue**: 只记录命令失败。
 - **Fail**: 任何 pod 中的任何容器中没有更多恢复 hook 运行。**Restore** CR 的状态将是 **PartiallyFailed**。

4.5. 故障排除

您可以使用 [OpenShift CLI 工具](#) 或 [Velero CLI 工具](#) 调试 Velero 自定义资源(CR)。Velero CLI 工具提供更详细的日志和信息。

您可以检查 [安装问题](#)、[备份和恢复 CR 问题](#)，以及 [Restic 问题](#)。

您可以使用 [must-gather 工具](#) 来收集日志、CR 信息和 Prometheus 指标数据。

您可以通过以下方法获取 Velero CLI 工具：

- 下载 Velero CLI 工具
- 访问集群中的 Velero 部署中的 Velero 二进制文件

4.5.1. 下载 Velero CLI 工具

您可以按照 [Velero 文档页面](#) 中的说明下载并安装 Velero CLI 工具。

该页面包括：

- 使用 Homebrew 的 macOS
- GitHub
- 使用 Chocolatey 的 Windows

先决条件

- 您可以访问启用了 DNS 和容器网络的 Kubernetes 集群 v1.16 或更高版本。
- 您已在本地安装了 **kubectl**。

流程

1. 打开浏览器，进入到 Verleo 网站上的"安装 CLI"。
2. 按照 macOS、GitHub 或 Windows 的适当流程。
3. 根据以下表，下载适用于 OADP 和 OpenShift Container Platform 版本的 Velero 版本：

表 4.2. OADP-Velero-OpenShift Container Platform 版本关系

OADP 版本	Velero 版本	OpenShift Container Platform 版本
1.0.0	1.7	4.6 及更新的版本
1.0.1	1.7	4.6 及更新的版本
1.0.2	1.7	4.6 及更新的版本
1.0.3	1.7	4.6 及更新的版本
1.1.0	{velero-version}	4.9 及更新的版本
1.1.1	{velero-version}	4.9 及更新的版本
1.1.2	{velero-version}	4.9 及更新的版本

4.5.2. 访问集群中的 Velero 部署中的 Velero 二进制文件

您可以使用 shell 命令访问集群中的 Velero 部署中的 Velero 二进制文件。

先决条件

- 您的 **DataProtectionApplication** 自定义资源的状态为 **Reconcile complete**。

流程

- 输入以下命令设定所需的别名：

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

4.5.3. 使用 OpenShift CLI 工具调试 Velero 资源

您可以使用 OpenShift CLI 工具检查 Velero 自定义资源(CR)和 **Velero** pod 日志来调试失败的备份或恢复。

Velero CR

使用 **oc describe** 命令检索与 **Backup** 或 **Restore** CR 关联的警告和错误概述：

```
$ oc describe <velero_cr> <cr_name>
```

Velero pod 日志

使用 **oc logs** 命令检索 **Velero** pod 日志：

```
$ oc logs pod/<velero>
```

Velero pod 调试日志

您可以在 **DataProtectionApplication** 资源中指定 Velero 日志级别，如下例所示。



注意

这个选项可从 OADP 1.0.3 开始。

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: velero-sample
spec:
  configuration:
    velero:
      logLevel: warning
```

可用的 **logLevel** 值如下：

- **trace**
- **debug**
- **info**
- **warning**
- 错误
- **fatal**
- **panic**

对于多数日志，建议使用 **debug**。

4.5.4. 使用 Velero CLI 工具调试 Velero 资源

您可以调试 **Backup** 和 **Restore** 自定义资源(CR)并使用 Velero CLI 工具检索日志。

Velero CLI 工具比 OpenShift CLI 工具提供更详细的信息。

语法

使用 **oc exec** 命令运行 Velero CLI 命令：

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> <command> <cr_name>
```

示例

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

帮助选项

使用 **velero --help** 列出所有 Velero CLI 命令：

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
--help
```

describe 命令

使用 **velero describe** 命令检索与 **Backup** 或 **Restore** CR 关联的警告和错误概述：

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
<backup_restore_cr> describe <cr_name>
```

示例

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

logs 命令

使用 **velero logs** 命令检索 **Backup** 或 **Restore** CR 的日志：

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
<backup_restore_cr> logs <cr_name>
```

示例

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
restore logs ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
```

4.5.5. Velero 和准入 Webhook 的问题

Velero 在恢复过程中解决准入 Webhook 问题的能力有限。如果您的工作负载带有准入 webhook，您可能需要使用额外的 Velero 插件或更改如何恢复工作负载。

通常，带有准入 Webhook 的工作负载需要您首先创建特定类型的资源。如果您的工作负载具有子资源，因为准入 webhook 通常阻止子资源，则会出现这种情况。

例如，创建或恢复顶层对象，如 **service.serving.knative.dev** 通常会创建子资源。如果您首先这样做，则不需要使用 Velero 创建和恢复这些资源。这可避免由 Velero 可使用的准入 Webhook 阻断子资源的问题。

4.5.5.1. 为使用准入 webhook 的 Velero 备份恢复临时解决方案

本节介绍了使用准入 webhook 的一些类型的 Velero 备份恢复资源所需的额外步骤。

4.5.5.1.1. 恢复 Knative 资源

您可能会遇到使用 Velero 备份使用准入 webhook 的 Knative 资源的问题。

在备份和恢复使用准入 webhook 的 Knative 资源时，您可以通过首先恢复顶层 **Service** 资源来避免这个问题。

流程

- 恢复顶层 **service.serving.knative.dev Service** 资源：

```
$ velero restore <restore_name> \
  --from-backup=<backup_name> --include-resources \
  service.serving.knative.dev
```

4.5.5.1.2. 恢复 IBM AppConnect 资源

如果您使用 Velero 恢复具有准入 webhook 的 IBM AppConnect 资源时遇到问题，您可以在此过程中运行检查。

流程

1. 检查集群中是否有 **kind: MutatingWebhookConfiguration** 的变异准入插件：

```
$ oc get mutatingwebhookconfigurations
```

2. 检查每个 **kind: MutatingWebhookConfiguration** 的 YAML 文件，以确保其没有规则块创建存在问题的对象。如需更多信息，请参阅[官方 Kubernetes 文档](#)。
3. 检查在备份时使用的 **type: Configuration.appconnect.ibm.com/v1beta1** 中的 **spec.version** 被已安装的 Operator 支持。

其他资源

- [准入插件](#)
- [Webhook 准入插件](#)
- [Webhook 准入插件类型](#)

4.5.6. 安装问题

在安装数据保护应用程序时，您可能会遇到使用无效目录或不正确的凭证导致的问题。

4.5.6.1. 备份存储包含无效目录

Velero pod 日志显示错误消息，**备份存储包含无效的顶级目录**。

原因

对象存储包含不是 Velero 目录的顶级目录。

解决方案

如果对象存储不适用于 Velero，则必须通过设置 **DataProtectionApplication** 清单中的 **spec.backupLocations.velero.objectStorage.prefix** 参数为存储桶指定一个前缀。

4.5.6.2. AWS 凭证不正确

oadp-aws-registry pod 日志会显示错误消息 **InvalidAccessKeyId: The AWS Access Key Id you provided does not exist in our records.**

Velero pod 日志显示错误消息 **NoCredentialProviders: no valid provider in chain.**

原因

用于创建 **Secret** 对象的 **credentials-velero** 文件会错误地格式化。

解决方案

确保 **credentials-velero** 文件已正确格式化，如下例所示：

credentials-velero 文件示例

```
[default] ❶
aws_access_key_id=AKIAIOSFODNN7EXAMPLE ❷
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

- ❶ AWS 默认配置集。
- ❷ 不用使用括号 (", ") 把值括起来。

4.5.7. 备份和恢复 CR 问题

您可能会遇到 **Backup** 和 **Restore** 自定义资源(CR)的常见问题。

4.5.7.1. 备份 CR 无法检索卷

Backup CR 显示错误消息 **InvalidVolume.NotFound: The volume 'vol-xxxx' does not exist.**

原因

持久性卷(PV)和快照位置位于不同的区域。

解决方案

1. 编辑 **DataProtectionApplication** 清单中的 **spec.snapshotLocations.velero.config.region** 键的值，使快照位置位于与 PV 相同的区域。
2. 创建新的 **Backup** CR。

4.5.7.2. 备份 CR 状态在进行中

Backup CR 的状态保留在 **InProgress** 阶段，且未完成。

原因

如果备份中断，则无法恢复。

解决方案

1. 检索 **Backup** CR 的详细信息：

```
$ oc -n {namespace} exec deployment/velero -c velero -- ./velero \
  backup describe <backup>
```

2. 删除 **Backup** CR：

```
$ oc delete backup <backup> -n openshift-adp
```

您不需要清理备份位置，因为正在进行的 **Backup** CR 没有上传文件到对象存储。

3. 创建新的 **Backup** CR。

4.5.7.3. 备份 CR 状态处于 PartiallyFailed

在没有 Restic 使用时一个 **Backup** CR 的状态保留在 **PartiallyFailed** 阶段，且没有完成。从属 PVC 的快照没有创建。

原因

如果备份是基于 CSI 快照类创建的，但缺少标签，CSI 快照插件将无法创建快照。因此，**Velero** pod 会记录类似如下的错误：

```
+
time="2023-02-17T16:33:13Z" level=error msg="Error backing up item" backup=openshift-adp/user1-backup-check5 error="error executing custom action (groupResource=persistentvolumeclaims, namespace=busy1, name=pvc1-user1): rpc error: code = Unknown desc = failed to get volumesnapshotclass for storageclass ocs-storagecluster-ceph-rbd: failed to get volumesnapshotclass for provisioner openshift-storage.rbd.csi.ceph.com, ensure that the desired volumesnapshot class has the velero.io/csi-volumesnapshot-class label" logSource="/remote-source/velero/app/pkg/backup/backup.go:417" name=busybox-79799557b5-vprq
```

解决方案

1. 删除 **Backup** CR：

```
$ oc delete backup <backup> -n openshift-adp
```

2. 如果需要，清理 **BackupStorageLocation** 上存储的数据以释放空间。
3. 将标签 **velero.io/csi-volumesnapshot-class=true** 应用到 **VolumeSnapshotClass** 对象：

```
$ oc label volumesnapshotclass/<snapclass_name> velero.io/csi-volumesnapshot-class=true
```

4. 创建新的 **Backup** CR。

4.5.8. Restic 问题

在使用 Restic 备份应用程序时，您可能会遇到这些问题。

4.5.8.1. 启用了 root_squash 的 NFS 数据卷的 Restic 权限错误

Restic pod 日志显示错误消息，**controller=pod-volume-backup error="fork/exec/usr/bin/restic: permission denied"**。

原因

如果您的 NFS 数据卷启用了 **root_squash**，**Restic** 映射到 **nfsnobody**，且没有创建备份的权限。

解决方案

您可以通过为 **Restic** 创建补充组并将组 ID 添加到 **DataProtectionApplication** 清单中来解决这个问题：

1. 在 NFS 数据卷中为 **Restic** 创建补充组。
2. 在 NFS 目录上设置 **setgid** 位，以便继承组所有权。
3. 将 **spec.configuration.restic.supplementalGroups** 参数和组 ID 添加到 **DataProtectionApplication** 清单中，如下例所示：

```
spec:
  configuration:
    restic:
      enable: true
      supplementalGroups:
        - <group_id> 1
```

- 1 指定补充组 ID。

4. 等待 **Restic** pod 重启，以便应用更改。

4.5.8.2. 在存储桶被强制后重新创建 Restic Backup CR

如果您为命名空间创建 Restic **Backup** CR，请清空对象存储的存储桶，然后为同一命名空间重新创建 **Backup** CR，重新创建的 **Backup** CR 会失败。

velero pod 日志显示以下错误消息：**stderr=Fatal: unable to open config file: Stat: The specified key does not exist.\nls there a repository at the following location?.**

原因

如果 Restic 目录从对象存储中删除，Velero 不会从 **ResticRepository** 清单重新创建或更新 Restic 存储库。如需更多信息，请参阅 [Velero 问题 4421](#)。

解决方案

- 运行以下命令，从命名空间中删除相关的 Restic 存储库：

```
$ oc delete resticrepository openshift-adp <name_of_the_restic_repository>
```

在以下错误日志中，**mysql-persistent** 是有问题的 Restic 存储库。存储库的名称会出现在其说明中。

```
time="2021-12-29T18:29:14Z" level=info msg="1 errors encountered backup up item" backup=velero/backup65 logSource="pkg/backup/backup.go:431" name=mysql-7d99fc949-qbkds time="2021-12-29T18:29:14Z" level=error msg="Error backing up item" backup=velero/backup65 error="pod volume backup failed: error running restic backup, stderr=Fatal: unable to open config file: Stat: The specified key does not exist.\nls there a repository at the following location?\nsrc3:http://minio-minio.apps.mayap-oadp-veleo-1234.qe.devcluster.openshift.com/mayapvelerooadp2/velero1/restic/mysql-persistent\n: exit status 1" error.file="/remote-source/src/github.com/vmware-tanzu/velero/pkg/restic/backupper.go:184"
```

```
error.function="github.com/vmware-tanzu/velero/
pkg/restic.(*backupper).BackupPodVolumes"
logSource="pkg/backup/backup.go:435" name=mysql-7d99fc949-qbkds
```

4.5.9. 使用 must-gather 工具

您可以使用 **must-gather** 工具收集有关 OADP 自定义资源的日志、指标和信息。

must-gather 数据必须附加到所有客户案例。

先决条件

- 您必须使用具有 **cluster-admin** 角色的用户登录到 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI (**oc**)。

流程

1. 进入要存储 **must-gather** 数据的目录。
2. 为以下数据收集选项之一运行 **oc adm must-gather** 命令：

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel8:v1.1
```

数据保存为 **must-gather/must-gather.tar.gz**。您可以将此文件上传到[红帽客户门户网站](#)中的支持问题单中。

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel8:v1.1 \
-- /usr/bin/gather_metrics_dump
```

此操作可能需要很长时间。数据保存为 **must-gather/metrics/prom_data.tar.gz**。

使用 Prometheus 控制台查看指标数据

您可以使用 Prometheus 控制台查看指标数据。

流程

1. 解压缩 **prom_data.tar.gz** 文件：

```
$ tar -xvzf must-gather/metrics/prom_data.tar.gz
```

2. 创建本地 Prometheus 实例：

```
$ make prometheus-run
```

命令输出 Prometheus URL。

输出

```
Started Prometheus on http://localhost:9090
```

3. 启动 Web 浏览器，再导航到 URL 以使用 Prometheus Web 控制台查看数据。

4. 查看数据后，删除 Prometheus 实例和数据：

```
$ make prometheus-cleanup
```

4.6. 与 OADP 一起使用的 API

本文档提供有关您可以在 OADP 一起使用的以下 API 的信息：

- Velero API
- OADP API

4.6.1. Velero API

Velero API 文档由 Velero 维护，而不是由红帽维护。它可在 [Velero API 类型](#) 中找到。

4.6.2. OADP API

下表提供了 OADP API 的结构：

表 4.3. DataProtectionApplicationSpec

属性	类型	描述
backupLocations	[] BackupLocation	定义用于 BackupStorageLocations 的配置列表。
snapshotLocations	[] SnapshotLocation	定义 VolumeSnapshotLocations 使用的配置列表。
unsupportedOverrides	map [UnsupportedImageKey] string	可用于覆盖为开发而部署的依赖镜像。选项为 veleroImageFqin , awsPluginImageFqin , openshiftPluginImageFqin , azurePluginImageFqin , gcpPluginImageFqin , csiPluginImageFqin , dataMoverImageFqin , resticRestoreImageFqin , kubevirtPluginImageFqin , and operator-type 。
podAnnotations	map [string] string	用于将注解添加到 Operator 部署的 pod。
podDnsPolicy	DNSPolicy	定义 Pod 的 DNS 的配置。
podDnsConfig	PodDNSConfig	定义除了由 DNSPolicy 生成的以外的 pod 的 DNS 参数。

属性	类型	描述
backupImages	*bool	用于指定是否要部署 registry 以启用镜像的备份和恢复。
配置	*ApplicationConfig	用于定义数据保护应用服务器配置。
功能	*特性	定义 DPA 的配置以启用技术预览功能。

[OADP API 的完整架构定义](#)。

表 4.4. BackupLocation

属性	类型	描述
velero	*velero.BackupStorageLocationSpec	存储卷快照的位置，如 备份存储位置 所述。
bucket	*CloudStorageLocation	[技术预览] 在某些云存储供应商处自动创建存储桶，用作备份存储位置。



重要

bucket 参数只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅[技术预览功能支持范围](#)。

类型 [BackupLocation](#) 的完整 schema 定义。

表 4.5. SnapshotLocation

属性	类型	描述
velero	*VolumeSnapshotLocationSpec	用于存储卷快照的位置，如 卷快照位置 。

类型 [SnapshotLocation](#) 的完整 schema 定义。

表 4.6. ApplicationConfig

属性	类型	描述
velero	* VeleroConfig	定义 Velero 服务器配置。
restic	* ResticConfig	定义 Restic 服务器配置。

类型 [ApplicationConfig](#) 的完整 schema 定义。

表 4.7. VeleroConfig

属性	类型	描述
featureFlags	[] string	定义为 Velero 实例启用的功能列表。
defaultPlugins	[] string	可以安装以下类型的默认 Velero 插件： aws 、 azure 、 csi 、 gcp 、 kubevirt 和 openshift 。
customPlugins	[] CustomPlugin	用于安装自定义 Velero 插件。 默认的以及自定义的插件信息包括在 OADP plug-ins 中
restoreResourcesVersionPriority	字符串	代表一个配置映射，它在定义与 EnableAPIGroupVersions 功能标记结合使用时会被创建。定义此字段会在 Velero 服务器功能标记中添加 EnableAPIGroupVersions 。
noDefaultBackupLocation	bool	要在没有默认备份存储位置的情况下安装 Velero，您必须设置 noDefaultBackupLocation 标志来确认安装。
podConfig	* PodConfig	定义 Velero pod 的配置。
logLevel	字符串	Velero 服务器日志级别（在最精细的日志中使用 debug ，对 Velero 默认保留未设置）。有效选项包括 trace 、 debug 、 info 、 warning 、 error 、 fatal 和 panic 。

类型为 [VeleroConfig](#) 的完整 schema 定义。

表 4.8. CustomPlugin

属性	类型	描述
name	字符串	自定义插件的名称。
image	字符串	自定义插件的镜像。

类型 **CustomPlugin** 的完整 schema 定义。

表 4.9. ResticConfig

属性	类型	描述
enable	*bool	如果设置为 true ，则使用 Restic 启用备份和恢复。如果设置为 false ，则需要快照。
supplementalGroups	[]int64	定义要应用到 Restic pod 的 Linux 组。
timeout	字符串	定义 Restic 超时的用户提供的持续时间字符串。默认值为 1hr (1 小时)。一个代表时间段的字符串，可以是一组十进制数字序列，每个数字都可以带有一个可选的分数及单位后缀，如 300ms 、 -1.5h 或 2h45m 。有效时间单位是 ns 、 us (或 µs)、 ms 、 s 、 m 和 h 。
podConfig	*PodConfig	定义 Restic pod 的配置。

类型为 **ResticConfig** 的完整 schema 定义。

表 4.10. PodConfig

属性	类型	描述
nodeSelector	map [string] string	定义要提供给 Velero podSpec 或 Restic podSpec 的 nodeSelector 。
容限 (tolerations)	[]Toleration	定义要应用到 Velero 部署或 Restic daemonset 的容限列表。
resourceAllocations	ResourceRequirements	为一个 Velero pod 或 Restic pod 设置特定的资源限值和请求，如设置 Velero CPU 和内存分配 所述。

属性	类型	描述
labels	map [string] string	要添加到 pod 的标签。

类型 **PodConfig** 的完整 schema 定义。

表 4.11. 功能

属性	类型	描述
dataMover	* DataMover	定义 Data Mover 的配置。

类型 **Features** 的完整 schema 定义。

表 4.12. DataMover

属性	类型	描述
enable	bool	如果设置为 true ，请部署卷快照控制器和修改的 CSI Data Mover 插件。如果设置为 false ，则不会部署它们。
credentialName	字符串	Data Mover 用户提供的 Restic Secret 名称。
timeout	字符串	要完成 VolumeSnapshotBackup 和 VolumeSnapshotRestore 的用户提供的持续时间字符串。默认值为 10m （10 分钟）。一个代表时间段的字符串，可以是一组十进制数字序列，每个数字都可以带有一个可选的分数及单位后缀，如 300ms 、 -1.5h 或 2h45m 。有效时间单位是 ns 、 us （或 µs ）、 ms 、 s 、 m 和 h 。

OADP API 在 [OADP Operator](#) 中更为详细。

4.7. 高级 OADP 特性和功能

本文档提供有关 OpenShift API for Data Protection (OADP) 的高级功能。

4.7.1. 在同一集群中使用不同的 Kubernetes API 版本

4.7.1.1. 列出集群中的 Kubernetes API 组版本

源集群可能会提供多个 API 版本，其中的一个版本是首选的 API 版本。例如，带有名为 **Example** 的 API 的源集群可能包括在 **example.com/v1** 和 **example.com/v1beta2** API 组中。

如果您使用 Velero 备份和恢复这样的源集群，Velero 仅备份了使用 Kubernetes API 首选版本的该资源的版本。

要返回上例，如果 **example.com/v1** 是首选的 API，则 Velero 只备份使用 **example.com/v1** 的资源的版本。另外，目标集群需要 **example.com/v1** 在它的一组可用 API 资源中注册，以便 Velero 恢复目标集群上的资源。

因此，您需要在目标集群上生成 Kubernetes API 组版本列表，以确保在一组可用的 API 资源中注册了首选的 API 版本。

流程

- 输入以下命令：

```
$ oc api-resources
```

4.7.1.2. 关于启用 API 组版本

默认情况下，Velero 只备份使用 Kubernetes API 的首选版本的资源。但是，Velero 还包括一个[启用 API 组版本](#)功能，它解决了这个限制。当在源集群中启用时，这个功能会使 Velero 备份集群中支持的**所有** Kubernetes API 组版本，而不只是首选集群。当版本存储在备份 .tar 文件中被保存后，可以在目标集群上恢复它们。

例如，带有名为 **Example** 的 API 的源集群可能包括在 **example.com/v1** 和 **example.com/v1beta2** API 组中，**example.com/v1** 是首选 API。

如果没有启用 Enable API Group Versions 功能，Velero 仅备份 **Example** 的首选 API 组版本，即 **example.com/v1**。启用该功能后，Velero 还会备份 **example.com/v1beta2**。

当目标集群上启用了“启用 API 组版本”功能时，Velero 根据 API 组版本优先级顺序选择恢复的版本。



注意

启用 API 组版本仍处于测试阶段。

Velero 使用以下算法为 API 版本分配优先级，并将 **1** 作为最高优先级：

1. *destination* 集群的首选版本
2. *source_cluster* 的首选版本
3. 带有最高 Kubernetes 版本优先级的通用非首选支持版本

其他资源

- [启用 API 组版本功能](#)

4.7.1.3. 使用启用 API 组版本

您可以使用 Velero 的启用 API 组版本功能来备份集群中支持的**所有** Kubernetes API 组版本，而不只是首选版本。



注意

启用 API 组版本仍处于测试阶段。

流程

- 配置 **EnableAPIGroupVersions** 功能标记：

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      featureFlags:
        - EnableAPIGroupVersions
```

其他资源

- [启用 API 组版本功能](#)

第 5 章 CONTROL PLANE 备份和恢复

5.1. 备份 ETCD

etcd 是 OpenShift Container Platform 的以“键-值”形式进行的存储，它会保留所有资源对象的状态。

定期备份集群的 etcd 数据，并在 OpenShift Container Platform 环境以外的安全位置保存备份数据。不要在第一个证书轮转完成前（安装后的 24 小时内）进行 etcd 备份，否则备份将包含过期的证书。另外，建议您在非高峰期使用 etcd 备份，因为 etcd 快照有较高的 I/O 成本。

确保升级集群后执行 etcd 备份。这很重要，因为当恢复集群时，必须使用从同一 z-stream 发行版本中获取的 etcd 备份。例如，OpenShift Container Platform 4.y.z 集群必须使用从 4.y.z 中获得的 etcd 备份。



重要

通过在 control plane 主机上执行一次备份脚本来备份集群的 etcd 数据。不要为每个 control plane 主机进行备份。

在进行了 etcd 备份后，就可以[恢复到一个以前的集群状态](#)。

5.1.1. 备份 etcd 数据

按照以下步骤，通过创建 etcd 快照并备份静态 pod 的资源来备份 etcd 数据。这个备份可以被保存，并在以后需要时使用它来恢复 etcd 数据。



重要

只保存单一 control plane 主机的备份。不要从集群中的每个 control plane 主机进行备份。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 您已检查是否启用了集群范围代理。

提示

您可以通过查看 **oc get proxy cluster -o yaml** 的输出检查代理是否已启用。如果 **httpProxy**、**httpsProxy** 和 **noProxy** 字段设置了值，则会启用代理。

流程

1. 为 control plane 节点启动一个 debug 会话：

```
$ oc debug node/<node_name>
```

2. 将您的根目录改为 **/host**：

```
sh-4.2# chroot /host
```

3. 如果启用了集群范围的代理，请确定已导出了 **NO_PROXY**、**HTTP_PROXY** 和 **HTTPS_PROXY** 环境变量。

4. 运行 **cluster-backup.sh** 脚本，输入保存备份的位置。

提示

cluster-backup.sh 脚本作为 etcd Cluster Operator 的一个组件被维护，它是 **etcdctl snapshot save** 命令的包装程序（wrapper）。

```
sh-4.4# /usr/local/bin/cluster-backup.sh /home/core/assets/backup
```

脚本输出示例

```
found latest kube-apiserver: /etc/kubernetes/static-pod-resources/kube-apiserver-pod-6
found latest kube-controller-manager: /etc/kubernetes/static-pod-resources/kube-controller-
manager-pod-7
found latest kube-scheduler: /etc/kubernetes/static-pod-resources/kube-scheduler-pod-6
found latest etcd: /etc/kubernetes/static-pod-resources/etcd-pod-3
ede95fe6b88b87ba86a03c15e669fb4aa5bf0991c180d3c6895ce72eaade54a1
etcdctl version: 3.4.14
API version: 3.4
{"level":"info","ts":1624647639.0188997,"caller":"snapshot/v3_snapshot.go:119","msg":"created
temporary db file","path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db.part"}
{"level":"info","ts":"2021-06-
25T19:00:39.030Z","caller":"clientv3/maintenance.go:200","msg":"opened snapshot stream;
downloading"}
{"level":"info","ts":1624647639.0301006,"caller":"snapshot/v3_snapshot.go:127","msg":"fetching
snapshot","endpoint":"https://10.0.0.5:2379"}
{"level":"info","ts":"2021-06-
25T19:00:40.215Z","caller":"clientv3/maintenance.go:208","msg":"completed snapshot read;
closing"}
{"level":"info","ts":1624647640.6032252,"caller":"snapshot/v3_snapshot.go:142","msg":"fetched
snapshot","endpoint":"https://10.0.0.5:2379","size":"114 MB","took":1.584090459}
{"level":"info","ts":1624647640.6047094,"caller":"snapshot/v3_snapshot.go:152","msg":"saved",
"path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db"}
Snapshot saved at /home/core/assets/backup/snapshot_2021-06-25_190035.db
{"hash":3866667823,"revision":31407,"totalKey":12828,"totalSize":114446336}
snapshot db and kube resources are successfully saved to /home/core/assets/backup
```

在这个示例中，在 control plane 主机上的 **/home/core/assets/backup/** 目录中创建了两个文件：

- **snapshot_<datetimestamp>.db**：这个文件是 etcd 快照。**cluster-backup.sh** 脚本确认其有效。
- **static_kuberresources_<datetimestamp>.tar.gz**：此文件包含静态 pod 的资源。如果启用了 etcd 加密，它也包含 etcd 快照的加密密钥。



注意

如果启用了 etcd 加密，建议出于安全考虑，将第二个文件与 etcd 快照分开保存。但是，需要这个文件才能从 etcd 快照中进行恢复。

请记住，etcd 仅对值进行加密，而不对键进行加密。这意味着资源类型、命名空间和对象名称是不加密的。

5.2. 替换不健康的 ETCD 成员

本文档描述了替换一个不健康 etcd 成员的过程。

此过程取决于 etcd 成员不健康的原因，如机器没有运行，或节点未就绪，或 etcd pod 处于 crashlooping 状态。



注意

如果您丢失了大多数 control plane 主机，请按照灾难恢复流程[恢复到以前的一个集群状态](#)，而不是这个过程。

如果 control plane 证书在被替换的成员中无效，则必须遵循[从已过期 control plane 证书中恢复](#)的步骤，而不是此过程。

如果 control plane 节点丢失并且创建了一个新节点，etcd 集群 Operator 将处理生成新 TLS 证书并将节点添加为 etcd 成员。

5.2.1. 先决条件

- 在替换不健康的 etcd 成员，需要进行 [etcd 备份](#)。

5.2.2. 找出一个不健康的 etcd 成员

您可以识别集群是否有不健康的 etcd 成员。

先决条件

- 使用具有 **cluster-admin** 角色的用户访问集群。

流程

- 使用以下命令检查 **EtcMembersAvailable** 状态条件的状态：

```
$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?(@.type=="EtcMembersAvailable")]}{.message}{"\n"}
```

- 查看输出：

```
2 of 3 members are available, ip-10-0-131-183.ec2.internal is unhealthy
```

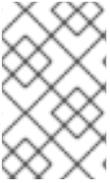
这个示例输出显示 **ip-10-0-131-183.ec2.internal** etcd 成员不健康。

5.2.3. 确定不健康的 etcd 成员的状态

替换不健康 etcd 成员的步骤取决于 etcd 的以下状态：

- 机器没有运行或者该节点未就绪
- etcd pod 处于 crashlooping 状态

此流程决定了 etcd 成员处于哪个状态。这可让您了解替换不健康的 etcd 成员要遵循的步骤。



注意

如果您知道机器没有运行或节点未就绪，但它们应该很快返回健康状态，那么您就不需要执行替换 etcd 成员的流程。当机器或节点返回一个健康状态时，etcd cluster Operator 将自动同步。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 您已找到不健康的 etcd 成员。

流程

1. 检查 **机器是否没有运行**:

```
$ oc get machines -A -ojsonpath='{range .items[*]}{@.status.nodeRef.name}{"\t"}
{@.status.providerStatus.instanceState}{"\n"}' | grep -v running
```

输出示例

```
ip-10-0-131-183.ec2.internal stopped 1
```

- 1** 此输出列出了节点以及节点机器的状态。如果状态不是 **running**，则代表机器没有运行。

如果机器没有运行，按照 **替换机器没有运行或节点没有就绪的非健康 etcd 成员** 过程进行操作。

2. 确定 **节点是否未就绪**。

如果以下任何一种情况是正确的，则代表节点没有就绪。

- 如果机器正在运行，检查节点是否不可访问：

```
$ oc get nodes -o jsonpath='{range .items[*]}{"\n"}{.metadata.name}{"\t"}{range
.spec.taints[*]}{.key}{ " "}' | grep unreachable
```

输出示例

```
ip-10-0-131-183.ec2.internal node-role.kubernetes.io/master
node.kubernetes.io/unreachable node.kubernetes.io/unreachable 1
```

- 1** 如果节点带有 **unreachable** 污点，则节点没有就绪。

- 如果该节点仍然可访问，则检查该节点是否列为 **NotReady**:

```
$ oc get nodes -l node-role.kubernetes.io/master | grep "NotReady"
```

输出示例

```
ip-10-0-131-183.ec2.internal NotReady master 122m v1.22.1 1
```

- 1** 如果节点列表为 **NotReady**，则 该节点没有就绪。

如果节点没有就绪，按照替换机器没有运行或节点没有就绪的 etcd 成员的步骤进行操作。

3. 确定 etcd Pod 是否处于 crashlooping 状态。

如果机器正在运行并且节点已就绪，请检查 etcd pod 是否处于 crashlooping 状态。

- a. 验证所有 control plane 节点都列为 **Ready** :

```
$ oc get nodes -l node-role.kubernetes.io/master
```

输出示例

```
NAME                                STATUS ROLES  AGE  VERSION
ip-10-0-131-183.ec2.internal Ready  master 6h13m v1.22.1
ip-10-0-164-97.ec2.internal  Ready  master 6h13m v1.22.1
ip-10-0-154-204.ec2.internal Ready  master 6h13m v1.22.1
```

- b. 检查 etcd pod 的状态是否为 **Error** 或 **CrashLoopBackOff**:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

输出示例

```
etcd-ip-10-0-131-183.ec2.internal      2/3  Error    7      6h9m 1
etcd-ip-10-0-164-97.ec2.internal      3/3  Running  0      6h6m
etcd-ip-10-0-154-204.ec2.internal      3/3  Running  0      6h6m
```

- 1 由于此 pod 的状态是 **Error**，因此 etcd pod 为 crashlooping 状态。

如果 etcd pod 为 crashlooping 状态，请按照替换 etcd pod 处于 crashlooping 状态的不健康的 etcd 成员的步骤进行操作。

5.2.4. 替换不健康的 etcd 成员

根据不健康的 etcd 成员的状态，使用以下一个流程：

- 替换机器没有运行或节点未就绪的不健康 etcd 成员
- 替换其 etcd Pod 处于 crashlooping 状态的不健康 etcd 成员
- 替换不健康的裸机 etcd 成员

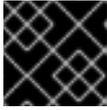
5.2.4.1. 替换机器没有运行或节点未就绪的不健康 etcd 成员

此流程详细介绍了替换因机器没有运行或节点未就绪造成不健康的 etcd 成员的步骤。

先决条件

- 您已找出不健康的 etcd 成员。
- 您已确认机器没有运行，或者该节点未就绪。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

- 已进行 etcd 备份。



重要

执行此流程前务必要进行 etcd 备份，以便在遇到任何问题时可以恢复集群。

流程

1. 删除不健康的成员。

- 选择一个不在受影响节点上的 pod:

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

输出示例

```
etcd-ip-10-0-131-183.ec2.internal    3/3   Running   0    123m
etcd-ip-10-0-164-97.ec2.internal    3/3   Running   0    123m
etcd-ip-10-0-154-204.ec2.internal    3/3   Running   0    124m
```

- 连接到正在运行的 etcd 容器，传递没有在受影响节点上的 pod 的名称：

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- 查看成员列表：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
+-----+
|   ID   | STATUS |   NAME   |   PEER ADDRS   |   CLIENT
ADDRS   |
+-----+-----+-----+-----+-----+
+-----+
| 6fc1e7c9db35841d | started | ip-10-0-131-183.ec2.internal | https://10.0.131.183:2380 |
https://10.0.131.183:2379 |
| 757b6793e2408b6c | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
+-----+
```

记录不健康的 etcd 成员的 ID 和名称，因为稍后需要这些值。**\$ etcdctl endpoint health** 命令将列出已删除的成员，直到完成替换过程并添加了新成员。

- 通过向 **etcdctl member remove** 命令提供 ID 来删除不健康的 etcd 成员：

```
sh-4.2# etcdctl member remove 6fc1e7c9db35841d
```

输出示例

```
Member 6fc1e7c9db35841d removed from cluster ead669ce1fbfb346
```

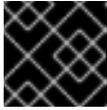
- e. 再次查看成员列表，并确认成员已被删除：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
-----+
|   ID   | STATUS |   NAME   |   PEER ADDRS   |   CLIENT
ADDRS   |
+-----+-----+-----+-----+-----+
-----+
| 757b6793e2408b6c | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
-----+
```

现在您可以退出节点 shell。

**重要**

删除成员后，在剩余的 etcd 实例重启时，集群可能无法访问。

2. 输入以下命令关闭仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

此命令可确保您可以成功重新创建机密并推出静态 pod。

3. 删除已删除的不健康 etcd 成员的旧 secret。

- a. 列出已删除的不健康 etcd 成员的 secret。

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

- 1** 传递您之前在这个过程中记录的不健康 etcd 成员的名称。

有一个对等的、服务和指标的 secret，如以下输出所示：

输出示例

```
etcd-peer-ip-10-0-131-183.ec2.internal      kubernetes.io/tls      2    47m
etcd-serving-ip-10-0-131-183.ec2.internal  kubernetes.io/tls      2    47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal  kubernetes.io/tls      2
```

47m

- b. 删除已删除的不健康 etcd 成员的 secret。
 - i. 删除 peer (对等) secret:

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

- ii. 删除 serving secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

- iii. 删除 metrics secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

- 4. 删除并重新创建 control plane 机器。重新创建此机器后，会强制一个新修订版本并自动扩展 etcd。

如果您正在运行安装程序置备的基础架构，或者您使用 Machine API 创建机器，请按照以下步骤执行。否则，您必须使用最初创建 master 时使用的相同方法创建新的 master。

- a. 获取不健康成员的机器。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

```
NAME                                PHASE  TYPE      REGION  ZONE  AGE
NODE                                PROVIDERID  STATE
clustername-8qw5l-master-0          Running m4.xlarge us-east-1 us-east-1a 3h37m
ip-10-0-131-183.ec2.internal aws:///us-east-1a/i-0ec2782f8287dfb7e stopped
❶
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-1b 3h37m
ip-10-0-154-204.ec2.internal aws:///us-east-1b/i-096c349b700a19631 running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-1c 3h37m
ip-10-0-164-97.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large us-east-1 us-east-1a 3h28m
ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-east-1b 3h28m
ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-east-1c 3h28m
ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-06861c00007751b0a running
```

- ❶ 这是不健康节点的 control plane 机器 **ip-10-0-131-183.ec2.internal**。

- b. 将机器配置保存到文件系统中的文件中：

```
$ oc get machine clustername-8qw5l-master-0 \ ❶
```

```
-n openshift-machine-api \
-o yaml \
> new-master-machine.yaml
```

- 1 为不健康的节点指定 control plane 机器的名称。

- c. 编辑上一步中创建的 **new-master-machine.yaml** 文件，以分配新名称并删除不必要的字段。

- i. 删除整个 **status** 部分：

```
status:
  addresses:
    - address: 10.0.131.183
      type: InternalIP
    - address: ip-10-0-131-183.ec2.internal
      type: InternalDNS
    - address: ip-10-0-131-183.ec2.internal
      type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Node
    name: ip-10-0-131-183.ec2.internal
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: awsproviderconfig.openshift.io/v1beta1
    conditions:
      - lastProbeTime: "2020-04-20T16:53:50Z"
        lastTransitionTime: "2020-04-20T16:53:50Z"
        message: machine successfully created
        reason: MachineCreationSucceeded
        status: "True"
        type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: AWSMachineProviderStatus
```

- ii. 将 **metadata.name** 字段更改为新名称。
建议您保留与旧机器相同的基础名称，并将结束号码改为下一个可用数字。在本例中，**clustername-8qw5l-master-0** 改为 **clustername-8qw5l-master-3**。

例如：

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: clustername-8qw5l-master-3
  ...
```

- iii. 删除 **spec.providerID** 字段：

```
providerID: aws:///us-east-1a/i-0fdb85790d76d0c3f
```

- d. 运行以下命令来删除 **BareMetalHost** 对象，将 **<host_name>** 替换为不健康节点的裸机主机的名称：

```
$ oc delete bmh -n openshift-machine-api <host_name>
```

- e. 运行以下命令删除不健康成员的机器，将 **<machine_name>** 替换为不健康节点的 control plane 机器的名称，如 **clustername-8qw5l-master-0**：

```
$ oc delete machine -n openshift-machine-api <machine_name>
```

- f. 验证机器是否已删除：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

```
NAME                               PHASE  TYPE      REGION  ZONE  AGE
NODE                               PROVIDERID  STATE
clustername-8qw5l-master-1        Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-154-204.ec2.internal aws:///us-east-1b/i-096c349b700a19631 running
clustername-8qw5l-master-2        Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-164-97.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large us-east-1 us-east-
1a 3h28m ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-010ef6279b4662ced
running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-east-1b
3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-east-
1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-06861c00007751b0a
running
```

- g. 使用 **new-master-machine.yaml** 文件创建新机器：

```
$ oc apply -f new-master-machine.yaml
```

- h. 验证新机器是否已创建：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

```
NAME                               PHASE  TYPE      REGION  ZONE  AGE
NODE                               PROVIDERID  STATE
clustername-8qw5l-master-1        Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-154-204.ec2.internal aws:///us-east-1b/i-096c349b700a19631 running
clustername-8qw5l-master-2        Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-164-97.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba running
clustername-8qw5l-master-3        Provisioning m4.xlarge us-east-1 us-east-1a
85s ip-10-0-133-53.ec2.internal aws:///us-east-1a/i-015b0888fe17bc2c8 running
❶ clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large us-east-1 us-
east-1a 3h28m ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-010ef6279b4662ced
running
```

```

clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-east-
1b 3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-0cb45ac45a166173b
running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-06861c00007751b0a
running

```

- 1 新机器 **clustername-8qw5l-master-3** 将被创建，并当阶段从 **Provisioning** 变为 **Running** 后就可以使用。

创建新机器可能需要几分钟时间。当机器或节点返回一个健康状态时，etcd cluster Operator 将自动同步。

5. 输入以下命令重新打开仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

6. 您可以输入以下命令验证 **unsupportedConfigOverrides** 部分是否已从对象中删除：

```
$ oc get etcd/cluster -oyaml
```

7. 如果使用单节点 OpenShift，请重启该节点。否则，您可能会在 etcd 集群 Operator 中遇到以下错误：

输出示例

```

EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-
0": the object has been modified; please apply your changes to the latest version and try
again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been
modified; please apply your changes to the latest version and try again, Operation cannot be
fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply
your changes to the latest version and try again]

```

验证

1. 验证所有 etcd pod 是否都正常运行。
在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

输出示例

```

etcd-ip-10-0-133-53.ec2.internal      3/3   Running   0       7m49s
etcd-ip-10-0-164-97.ec2.internal      3/3   Running   0       123m
etcd-ip-10-0-154-204.ec2.internal     3/3   Running   0       124m

```

如果上一命令的输出只列出两个 pod，您可以手动强制重新部署 etcd。在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-"'$( date --rfc-
3339=ns )''}}' --type=merge 1
```

1 **forceRedeploymentReason** 值必须是唯一的，这就是为什么附加时间戳的原因。

2. 验证只有三个 etcd 成员。

- a. 连接到正在运行的 etcd 容器，传递没有在受影响节点上的 pod 的名称：
在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- b. 查看成员列表：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
+-----+
| ID      | STATUS | NAME          | PEER ADDRS      | CLIENT
ADDRS    |
+-----+-----+-----+-----+-----+
+-----+
| 5eb0d6b8ca24730c | started | ip-10-0-133-53.ec2.internal | https://10.0.133.53:2380 |
https://10.0.133.53:2379 |
| 757b6793e2408b6c | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
+-----+
```

如果上一命令的输出列出了超过三个 etcd 成员，您必须删除不需要的成员。



警告

确保删除正确的 etcd 成员；如果删除了正常的 etcd 成员则有可能导致仲裁丢失。

5.2.4.2. 替换其 etcd Pod 处于 crashlooping 状态的不健康 etcd 成员

此流程详细介绍了替换因 etcd pod 处于 crashlooping 状态造成不健康的 etcd 成员的步骤。

先决条件

- 您已找出不健康的 etcd 成员。
- 已确认 etcd pod 处于 crashlooping 状态。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

- 已进行 etcd 备份。



重要

执行此流程前务必要进行 etcd 备份，以便在遇到任何问题时可以恢复集群。

流程

1. 停止处于 crashlooping 状态的 etcd pod。

- a. 对处于 crashlooping 状态的节点进行调试。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc debug node/ip-10-0-131-183.ec2.internal 1
```

- 1 使用不健康节点的名称来替换它。

- b. 将您的根目录改为 **/host**：

```
sh-4.2# chroot /host
```

- c. 将现有 etcd pod 文件从 Kubelet 清单目录中移出：

```
sh-4.2# mkdir /var/lib/etcd-backup
```

```
sh-4.2# mv /etc/kubernetes/manifests/etcd-pod.yaml /var/lib/etcd-backup/
```

- d. 将 etcd 数据目录移到不同的位置：

```
sh-4.2# mv /var/lib/etcd/ /tmp
```

现在您可以退出节点 shell。

2. 删除不健康的成员。

- a. 选择一个不在受影响节点上的 pod。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

输出示例

```
etcd-ip-10-0-131-183.ec2.internal    2/3   Error    7    6h9m
etcd-ip-10-0-164-97.ec2.internal    3/3   Running  0    6h6m
etcd-ip-10-0-154-204.ec2.internal    3/3   Running  0    6h6m
```

- b. 连接到正在运行的 etcd 容器，传递没有在受影响节点上的 pod 的名称。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- c. 查看成员列表：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
-----+
|   ID   | STATUS |   NAME   |   PEER ADDRS   |   CLIENT
ADDRS   |
+-----+-----+-----+-----+-----+
-----+
| 62bcf33650a7170a | started | ip-10-0-131-183.ec2.internal | https://10.0.131.183:2380 |
https://10.0.131.183:2379 |
| b78e2856655bc2eb | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
-----+
```

记录不健康的 etcd 成员的 ID 和名称，因为稍后需要这些值。

- d. 通过向 **etcdctl member remove** 命令提供 ID 来删除不健康的 etcd 成员：

```
sh-4.2# etcdctl member remove 62bcf33650a7170a
```

输出示例

```
Member 62bcf33650a7170a removed from cluster ead669ce1fbfb346
```

- e. 再次查看成员列表，并确认成员已被删除：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
-----+
|   ID   | STATUS |   NAME   |   PEER ADDRS   |   CLIENT
ADDRS   |
+-----+-----+-----+-----+-----+
-----+
| b78e2856655bc2eb | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
-----+
```

现在您可以退出节点 shell。

3. 输入以下命令关闭仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

此命令可确保您可以成功重新创建机密并推出静态 pod。

4. 删除已删除的不健康 etcd 成员的旧 secret。

a. 列出已删除的不健康 etcd 成员的 secret。

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

1 传递您之前在这个过程中记录的不健康 etcd 成员的名称。

有一个对等的、服务和指标的 secret，如以下输出所示：

输出示例

```
etcd-peer-ip-10-0-131-183.ec2.internal      kubernetes.io/tls      2    47m
etcd-serving-ip-10-0-131-183.ec2.internal  kubernetes.io/tls      2    47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal kubernetes.io/tls      2
47m
```

b. 删除已删除的不健康 etcd 成员的 secret。

i. 删除 peer（对等）secret:

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

ii. 删除 serving secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

iii. 删除 metrics secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

5. 强制 etcd 重新部署。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "single-master-recovery-$( date --rfc-3339=ns )"' --type=merge 1
```

1 **forceRedeploymentReason** 值必须是唯一的，这就是为什么附加时间戳的原因。

当 etcd 集群 Operator 执行重新部署时，它会确保所有 control plane 节点都有可正常工作的 etcd pod。

6. 输入以下命令重新打开仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

7. 您可以输入以下命令验证 **unsupportedConfigOverrides** 部分是否已从对象中删除：

```
$ oc get etcd/cluster -oyaml
```

8. 如果使用单节点 OpenShift，请重启该节点。否则，您可能在 etcd 集群 Operator 中遇到以下错误：

输出示例

```
EtcDCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]
```

验证

- 确认新成员可用且健康。
 - a. 连接到正在运行的 etcd 容器。
在一个终端中使用 cluster-admin 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- b. 验证所有成员是否健康：

```
sh-4.2# etcdctl endpoint health
```

输出示例

```
https://10.0.131.183:2379 is healthy: successfully committed proposal: took = 16.671434ms
https://10.0.154.204:2379 is healthy: successfully committed proposal: took = 16.698331ms
https://10.0.164.97:2379 is healthy: successfully committed proposal: took = 16.621645ms
```

5.2.4.3. 替换机器没有运行或节点未就绪的不健康裸机 etcd 成员

此流程详细介绍了替换因机器没有运行或节点未就绪造成不健康的裸机 etcd 成员的步骤。

如果您正在运行安装程序置备的基础架构，或者您使用 Machine API 创建机器，请按照以下步骤执行。否则，您必须使用最初创建控制平面节点时使用的相同方法创建新的控制平面。

先决条件

- 您已找出不健康的裸机 etcd 成员。
- 您已确认机器没有运行，或者该节点未就绪。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

- 已进行 etcd 备份。



重要

执行此流程前务必要进行 etcd 备份，以便在遇到任何问题时可以恢复集群。

流程

1. 验证并删除不健康的成员。

- 选择一个不在受影响节点上的 pod:

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd -o wide
```

输出示例

```
etcd-openshift-control-plane-0 5/5 Running 11 3h56m 192.168.10.9 openshift-
control-plane-0 <none> <none>
etcd-openshift-control-plane-1 5/5 Running 0 3h54m 192.168.10.10 openshift-
control-plane-1 <none> <none>
etcd-openshift-control-plane-2 5/5 Running 0 3h58m 192.168.10.11 openshift-
control-plane-2 <none> <none>
```

- 连接到正在运行的 etcd 容器，传递没有在受影响节点上的 pod 的名称：

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

- 查看成员列表：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
+-----+
| ID          | STATUS | NAME                | PEER ADDRS          | CLIENT
ADDRS        | IS LEARNER |                      |                      |
+-----+-----+-----+-----+-----+
+-----+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380/ |
https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380/ |
https://192.168.10.10:2379/ | false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380/ |
https://192.168.10.9:2379/ | false |
+-----+-----+-----+-----+-----+
+-----+
```

记录不健康的 etcd 成员的 ID 和名称，因为稍后需要这些值。**etcdctl endpoint health** 命令将列出已删除的成员，直到完成替换过程并添加了新成员。

- d. 通过向 `etcdctl member remove` 命令提供 ID 来删除不健康的 etcd 成员：



警告

确保删除正确的 etcd 成员；如果删除了正常的 etcd 成员则有可能导致仲裁丢失。

```
sh-4.2# etcdctl member remove 7a8197040a5126c8
```

输出示例

```
Member 7a8197040a5126c8 removed from cluster b23536c33f2cdd1b
```

- e. 再次查看成员列表，并确认成员已被删除：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
+-----+
| ID          | STATUS | NAME                | PEER ADDRS          | CLIENT
ADDRS        | IS LEARNER |                      |                      |
+-----+-----+-----+-----+-----+
+-----+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380/ |
https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380/ |
https://192.168.10.10:2379/ | false |
+-----+-----+-----+-----+-----+
+-----+
```

现在您可以退出节点 shell。



重要

删除成员后，在剩余的 etcd 实例重启时，集群可能无法访问。

2. 输入以下命令关闭仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

此命令可确保您可以成功重新创建机密并推出静态 pod。

3. 运行以下命令，删除已删除的不健康 etcd 成员的旧 secret。

- a. 列出已删除的不健康 etcd 成员的 secret。

```
$ oc get secrets -n openshift-etcd | grep openshift-control-plane-2
```

传递您之前在这个过程中记录的不健康 etcd 成员的名称。

有一个对等的、服务和指标的 secret，如以下输出所示：

```
etcd-peer-openshift-control-plane-2      kubernetes.io/tls 2 134m
etcd-serving-metrics-openshift-control-plane-2 kubernetes.io/tls 2 134m
etcd-serving-openshift-control-plane-2    kubernetes.io/tls 2 134m
```

b. 删除已删除的不健康 etcd 成员的 secret。

i. 删除 peer（对等）secret:

```
$ oc delete secret etcd-peer-openshift-control-plane-2 -n openshift-etcd
secret "etcd-peer-openshift-control-plane-2" deleted
```

ii. 删除 serving secret:

```
$ oc delete secret etcd-serving-metrics-openshift-control-plane-2 -n openshift-etcd
secret "etcd-serving-metrics-openshift-control-plane-2" deleted
```

iii. 删除 metrics secret:

```
$ oc delete secret etcd-serving-openshift-control-plane-2 -n openshift-etcd
secret "etcd-serving-openshift-control-plane-2" deleted
```

4. 删除 control plane 机器。

如果您正在运行安装程序置备的基础架构，或者您使用 Machine API 创建机器，请按照以下步骤执行。否则，您必须使用最初创建控制平面节点时使用的相同方法创建新的控制平面。

a. 获取不健康成员的机器。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

```
NAME                                PHASE  TYPE  REGION  ZONE  AGE  NODE STATE
PROVIDERID
examplecluster-control-plane-0      Running                3h11m openshift-control-
plane-0 baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-
3ff2-41c5-b099-0aa41222964e externally provisioned 1
examplecluster-control-plane-1      Running                3h11m openshift-control-
plane-1 baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-
329c-475e-8d81-03b20280a3e1 externally provisioned
examplecluster-control-plane-2      Running                3h11m openshift-control-
plane-2 baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-
61d8-410f-be5b-6a395b056135 externally provisioned
examplecluster-compute-0            Running                165m  openshift-compute-0
```

```
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f    provisioned
examplecluster-compute-1    Running    165m    openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fd6e6eb-2066-4241-91dc-e7ea72ab13b9    provisioned
```

- 1 这是不健康节点的 control plane 机器，**examplecluster-control-plane-2**。

- b. 将机器配置保存到文件系统中的文件中：

```
$ oc get machine examplecluster-control-plane-2 \ 1
-n openshift-machine-api \
-o yaml \
> new-master-machine.yaml
```

- 1 为不健康的节点指定 control plane 机器的名称。

- c. 编辑上一步中创建的 **new-master-machine.yaml** 文件，以分配新名称并删除不必要的字段。

- i. 删除整个 **status** 部分：

```
status:
  addresses:
    - address: ""
      type: InternalIP
    - address: fe80::4adf:37ff:feb0:8aa1%ens1f1.373
      type: InternalDNS
    - address: fe80::4adf:37ff:feb0:8aa1%ens1f1.371
      type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Machine
    name: fe80::4adf:37ff:feb0:8aa1%ens1f1.372
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: machine.openshift.io/v1beta1
    conditions:
      - lastProbeTime: "2020-04-20T16:53:50Z"
        lastTransitionTime: "2020-04-20T16:53:50Z"
        message: machine successfully created
        reason: MachineCreationSucceeded
        status: "True"
        type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: Machine
```

5. 将 **metadata.name** 字段更改为新名称。

建议您保留与旧机器相同的基础名称，并将结束号码改为下一个可用数字。在本例中，**examplecluster-control-plane-2** 改为 **examplecluster-control-plane-3**。

例如：

```

apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: examplecluster-control-plane-3
  ...

```

- a. 删除 **spec.providerID** 字段：

```

  providerID: baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-
  61d8-410f-be5b-6a395b056135

```

- b. 删除 **metadata.annotations** 和 **metadata.generation** 字段：

```

  annotations:
    machine.openshift.io/instance-state: externally provisioned
    ...
  generation: 2

```

- c. 删除 **spec.conditions**、**spec.lastUpdated**、**spec.nodeRef** 和 **spec.phase** 字段：

```

  lastTransitionTime: "2022-08-03T08:40:36Z"
  message: 'Drain operation currently blocked by: [{"Name:EtcdQuorumOperator
  Owner:clusteroperator/etcd}]'
  reason: HookPresent
  severity: Warning
  status: "False"

  type: Drainable
  lastTransitionTime: "2022-08-03T08:39:55Z"
  status: "True"
  type: InstanceExists

  lastTransitionTime: "2022-08-03T08:36:37Z"
  status: "True"
  type: Terminable
  lastUpdated: "2022-08-03T08:40:36Z"
  nodeRef:
    kind: Node
    name: openshift-control-plane-2
    uid: 788df282-6507-4ea2-9a43-24f237ccbc3c
  phase: Running

```

6. 运行以下命令，确保 Bare Metal Operator 可用：

```
$ oc get clusteroperator baremetal
```

输出示例

```

NAME          VERSION AVAILABLE PROGRESSING DEGRADED SINCE MESSAGE
baremetal    4.10.x  True     False     False   3d15h

```

7. 运行以下命令来删除旧的 **BareMetalHost** 对象：

-

```
$ oc delete bmh openshift-control-plane-2 -n openshift-machine-api
```

输出示例

```
baremetalhost.metal3.io "openshift-control-plane-2" deleted
```

8. 运行以下命令来删除不健康成员的机器：

```
$ oc delete machine -n openshift-machine-api examplecluster-control-plane-2
```

删除 **BareMetalHost** 和 **Machine** 对象后，**Machine** Controller 会自动删除 **Node** 对象。

如果删除机器因任何原因或者命令被移动而延迟而延迟而延迟，您可以通过删除机器对象终结器字段来强制删除。



重要

不要通过按 **Ctrl+c** 中断机器删除。您必须允许命令继续完成。打开一个新的终端窗口来编辑并删除 `finalizer` 字段。

- a. 运行以下命令来编辑机器配置：

```
$ oc edit machine -n openshift-machine-api examplecluster-control-plane-2
```

- b. 删除 **Machine** 自定义资源中的以下字段，然后保存更新的文件：

```
finalizers:
- machine.machine.openshift.io
```

输出示例

```
machine.machine.openshift.io/examplecluster-control-plane-2 edited
```

9. 运行以下命令验证机器是否已删除：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

```
NAME                               PHASE   TYPE   REGION  ZONE  AGE   NODE
PROVIDERID                          STATE
examplecluster-control-plane-0     Running                3h11m openshift-control-plane-0
baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-3ff2-41c5-b099-0aa41222964e  externally provisioned
examplecluster-control-plane-1     Running                3h11m openshift-control-plane-1
baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-329c-475e-8d81-03b20280a3e1  externally provisioned
examplecluster-compute-0           Running                165m  openshift-compute-0
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f  provisioned
```

```
examplecluster-compute-1    Running    165m  openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-91dc-
e7ea72ab13b9    provisioned
```

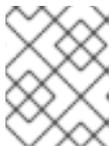
10. 运行以下命令验证节点是否已删除：

```
$ oc get nodes

NAME                STATUS ROLES  AGE  VERSION
openshift-control-plane-0 Ready master 3h24m v1.24.0+9546431
openshift-control-plane-1 Ready master 3h24m v1.24.0+9546431
openshift-compute-0   Ready worker 176m v1.24.0+9546431
openshift-compute-1   Ready worker 176m v1.24.0+9546431
```

11. 创建新的 **BareMetalHost** 对象和 secret，以存储 BMC 凭证：

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: openshift-control-plane-2-bmc-secret
  namespace: openshift-machine-api
data:
  password: <password>
  username: <username>
type: Opaque
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-control-plane-2
  namespace: openshift-machine-api
spec:
  automatedCleaningMode: disabled
  bmc:
    address: redfish://10.46.61.18:443/redfish/v1/Systems/1
    credentialsName: openshift-control-plane-2-bmc-secret
    disableCertificateVerification: true
    bootMACAddress: 48:df:37:b0:8a:a0
    bootMode: UEFI
    externallyProvisioned: false
    online: true
    rootDeviceHints:
      deviceName: /dev/sda
    userData:
      name: master-user-data-managed
      namespace: openshift-machine-api
EOF
```



注意

用户名和密码可从其他裸机主机的 secret 中找到。**bmc:address** 中使用的协议可以从其他 bmh 对象获取。



重要

如果您从现有 control plane 主机重复使用 **BareMetalHost** 对象定义，请不要将 external **Provisioned** 字段保留为 **true**。

如果 OpenShift Container Platform 安装程序置备，现有 control plane **BareMetalHost** 对象可能会将 **externallyProvisioned** 标记设为 **true**。

检查完成后，**BareMetalHost** 对象会被创建并可用置备。

12. 使用可用的 **BareMetalHost** 对象验证创建过程：

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

NAME	STATE	CONSUMER	ONLINE	ERROR	AGE
openshift-control-plane-0	externally provisioned	examplecluster-control-plane-0	true		4h48m
openshift-control-plane-1	externally provisioned	examplecluster-control-plane-1	true		4h48m
openshift-control-plane-2	available	examplecluster-control-plane-3	true		47m
openshift-compute-0	provisioned	examplecluster-compute-0	true		4h48m
openshift-compute-1	provisioned	examplecluster-compute-1	true		4h48m

a. 使用 **new-master-machine.yaml** 文件创建新 control plane 机器：

```
$ oc apply -f new-master-machine.yaml
```

b. 验证新机器是否已创建：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例

NAME	PHASE	TYPE	REGION	ZONE	AGE	NODE
examplecluster-control-plane-0	Running				3h11m	openshift-control-plane-0
examplecluster-control-plane-0	externally provisioned					baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-3ff2-41c5-b099-0aa41222964e
examplecluster-control-plane-1	Running				3h11m	openshift-control-plane-1
examplecluster-control-plane-1	externally provisioned					baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-329c-475e-8d81-03b20280a3e1
examplecluster-control-plane-2	Running				3h11m	openshift-control-plane-2
examplecluster-control-plane-2	externally provisioned					baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-61d8-410f-be5b-6a395b056135
examplecluster-compute-0	Running				165m	openshift-compute-0
examplecluster-compute-0	provisioned					baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f
examplecluster-compute-1	Running				165m	openshift-compute-1
examplecluster-compute-1	provisioned					baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-91dc-e7ea72ab13b9

1 新机器 **clustername-8qw5l-master-3** 会被创建，并在阶段从 **Provisioning** 变为 **Running** 后就绪。

创建新机器需要几分钟时间。当机器或节点返回一个健康状态时，etcd cluster Operator 将自动同步。

- c. 运行以下命令验证裸机主机是否被置备，且没有报告的错误：

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

输出示例

```
$ oc get bmh -n openshift-machine-api
NAME                                STATE                CONSUMER                                ONLINE ERROR AGE
openshift-control-plane-0          externally provisioned examplecluster-control-plane-0 true
4h48m
openshift-control-plane-1          externally provisioned examplecluster-control-plane-1 true
4h48m
openshift-control-plane-2          provisioned           examplecluster-control-plane-3 true
47m
openshift-compute-0                provisioned           examplecluster-compute-0 true
4h48m
openshift-compute-1                provisioned           examplecluster-compute-1 true
4h48m
```

- d. 运行以下命令验证新节点是否已添加并处于就绪状态：

```
$ oc get nodes
```

输出示例

```
$ oc get nodes
NAME                                STATUS ROLES AGE VERSION
openshift-control-plane-0          Ready master 4h26m v1.24.0+9546431
openshift-control-plane-1          Ready master 4h26m v1.24.0+9546431
openshift-control-plane-2          Ready master 12m v1.24.0+9546431
openshift-compute-0                Ready worker 3h58m v1.24.0+9546431
openshift-compute-1                Ready worker 3h58m v1.24.0+9546431
```

13. 输入以下命令重新打开仲裁保护：

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

14. 您可以输入以下命令验证 **unsupportedConfigOverrides** 部分是否已从对象中删除：

```
$ oc get etcd/cluster -oyaml
```

15. 如果使用单节点 OpenShift，请重启该节点。否则，您可能在 etcd 集群 Operator 中遇到以下错误：

输出示例

```
EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been
```

modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]

验证

1. 验证所有 etcd pod 是否都正常运行。
在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd -o wide
```

输出示例

```
etcd-openshift-control-plane-0    5/5    Running    0    105m
etcd-openshift-control-plane-1    5/5    Running    0    107m
etcd-openshift-control-plane-2    5/5    Running    0    103m
```

如果上一命令的输出只列出两个 pod，您可以手动强制重新部署 etcd。在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc patch etcd cluster -p='{ "spec": { "forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"' }' --type=merge 1
```

- 1** **forceRedeploymentReason** 值必须是唯一的，这就是为什么附加时间戳的原因。

要验证是否有完全有三个 etcd 成员，连接到正在运行的 etcd 容器，传递没有在受影响节点上的 pod 的名称。在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

2. 查看成员列表：

```
sh-4.2# etcdctl member list -w table
```

输出示例

```
+-----+-----+-----+-----+-----+
+-----+
| ID | STATUS | NAME | PEER ADDRS | CLIENT ADDRS |
| IS LEARNER |
+-----+-----+-----+-----+-----+
+-----+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380 |
https://192.168.10.11:2379 | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380 |
https://192.168.10.10:2379 | false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380 |
https://192.168.10.9:2379 | false |
+-----+-----+-----+-----+-----+
+-----+
```



注意

如果上一命令的输出列出了超过三个 etcd 成员，您必须删除不需要的成员。

3. 运行以下命令，验证所有 etcd 成员是否健康：

```
# etcdctl endpoint health --cluster
```

输出示例

```
https://192.168.10.10:2379 is healthy: successfully committed proposal: took = 8.973065ms
https://192.168.10.9:2379 is healthy: successfully committed proposal: took = 11.559829ms
https://192.168.10.11:2379 is healthy: successfully committed proposal: took = 11.665203ms
```

4. 运行以下命令，验证所有节点是否处于最新的修订版本：

```
$ oc get etcd -o=jsonpath='{range.items[0].status.conditions[?(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

```
AllNodesAtLatestRevision
```

5.3. 灾难恢复

5.3.1. 关于灾难恢复

灾难恢复文档为管理员提供了如何从 OpenShift Container Platform 集群可能出现的几个灾难情形中恢复的信息。作为管理员，您可能需要遵循以下一个或多个步骤将集群恢复为工作状态。



重要

灾难恢复要求您至少有一个健康的 control plane 主机。

恢复到一个以前的集群状态

如果您希望将集群恢复到一个以前的状态时（例如，管理员错误地删除了一些关键信息），则可以使用这个解决方案。这包括您丢失了大多数 control plane 主机并导致 etcd 仲裁丢失，且集群离线的情况。只要您执行了 etcd 备份，就可以按照这个步骤将集群恢复到之前的状态。

如果适用，您可能还需要从过期的 control plane 证书中恢复。



警告

在一个正在运行的集群中恢复到以前的集群状态是破坏性的，而不稳定的操作。这仅应作为最后的手段使用。

在执行恢复前，请参阅[关于恢复集群状态](#)以了解有关对集群的影响的更多信息。



注意

如果大多数 master 仍可用，且仍有 etcd 仲裁，请按照以下步骤[替换一个不健康的 etcd 成员](#)。

从 control plane 证书已过期的情况下恢复

如果 control plane 证书已经过期，则可以使用这个解决方案。例如：在第一次证书轮转前（在安装后 24 小时内）关闭了集群，您的证书将不会被轮转，且会过期。可以按照以下步骤从已过期的 control plane 证书中恢复。

5.3.2. 恢复到一个以前的集群状态

为了将集群还原到以前的状态，您必须已通过创建快照[备份了 etcd 数据](#)。您将需要使用此快照来还原集群状态。

5.3.2.1. 关于恢复集群状态

您可以使用 etcd 备份将集群恢复到以前的状态。在以下情况中可以使用这个方法进行恢复：

- 集群丢失了大多数 control plane 主机（仲裁丢失）。
- 管理员删除了一些关键内容，必须恢复才能恢复集群。



警告

在一个正在运行的集群中恢复到以前的集群状态是破坏性的，而不稳定的操作。这仅应作为最后的手段使用。

如果您可以使用 Kubernetes API 服务器检索数据，则代表 etcd 可用，且您不应该使用 etcd 备份来恢复。

恢复 etcd 实际相当于把集群返回到以前的一个状态，所有客户端都会遇到一个有冲突的、并行历史记录。这会影响 kubelet、Kubernetes 控制器、SDN 控制器和持久性卷控制器等监视组件的行为。

当 etcd 中的内容与磁盘上的实际内容不匹配时，可能会导致 Operator churn，从而导致 Kubernetes API 服务器、Kubernetes 控制器管理器、Kubernetes 调度程序和 etcd 的 Operator 在磁盘上的文件与 etcd 中的内容冲突时卡住。这可能需要手动操作来解决问题。

在极端情况下，集群可能会丢失持久性卷跟踪，删除已不存在的关键工作负载，重新镜像机器，以及重写带有过期证书的 CA 捆绑包。

5.3.2.2. 恢复到一个以前的集群状态

您可以使用保存的 etcd 备份来恢复以前的集群状态，或恢复丢失了大多数 control plane 主机的集群。



重要

恢复集群时，必须使用同一 z-stream 发行版本中获取的 etcd 备份。例如，OpenShift Container Platform 4.7.2 集群必须使用从 4.7.2 开始的 etcd 备份。

先决条件

- 使用具有 **cluster-admin** 角色的用户访问集群。
- 用作恢复主机的健康 control plane 主机。
- SSH 对 control plane 主机的访问。
- 包含从同一备份中获取的 etcd 快照和静态 pod 资源的备份目录。该目录中的文件名必须采用以下格式: **snapshot_<datetimestamp>.db** 和 **static_kubernetes_<datetimestamp>.tar.gz**。



重要

对于非恢复 control plane 节点，不需要建立 SSH 连接或停止静态 pod。您可以逐个删除并重新创建其他非恢复 control plane 机器。

流程

1. 选择一个要用作恢复主机的 control plane 主机。这是您要在其中运行恢复操作的主机。
2. 建立到每个 control plane 节点（包括恢复主机）的 SSH 连接。
恢复过程启动后，Kubernetes API 服务器将无法访问，因此您无法访问 control plane 节点。因此，建议在一个单独的终端中建立到每个 control plane 主机的 SSH 连接。



重要

如果没有完成这个步骤，将无法访问 control plane 主机来完成恢复过程，您将无法从这个状态恢复集群。

3. 将 etcd 备份目录复制复制到恢复 control plane 主机上。
此流程假设您将 **backup** 目录（其中包含 etcd 快照和静态 pod 资源）复制到恢复 control plane 主机的 **/home/core/** 目录中。
4. 在任何其他 control plane 节点上停止静态 pod。



注意

不需要手动停止恢复主机上的 pod。恢复脚本将停止恢复主机上的 pod。

- a. 访问不是恢复主机的 control plane 主机。
- b. 将现有 etcd pod 文件从 Kubelet 清单目录中移出：

```
$ sudo mv /etc/kubernetes/manifests/etcd-pod.yaml /tmp
```

- c. 验证 etcd pod 是否已停止。

```
$ sudo crictl ps | grep etcd | grep -v operator
```

命令输出应该为空。如果它不是空的，请等待几分钟后再重新检查。

- d. 将现有 Kubernetes API 服务器 pod 文件移出 kubelet 清单目录中：

```
$ sudo mv /etc/kubernetes/manifests/kube-apiserver-pod.yaml /tmp
```

- e. 验证 Kubernetes API 服务器 pod 是否已停止。

```
$ sudo crictl ps | grep kube-apiserver | grep -v operator
```

命令输出应该为空。如果它不是空的，请等待几分钟后重新检查。

- f. 将 etcd 数据目录移到不同的位置：

```
$ sudo mv /var/lib/etcd/ /tmp
```

- g. 在其他不是恢复主机的 control plane 主机上重复此步骤。

5. 访问恢复 control plane 主机。

6. 如果启用了集群范围的代理，请确定已导出了 **NO_PROXY**、**HTTP_PROXY**和 **HTTPS_PROXY** 环境变量。

提示

您可以通过查看 **oc get proxy cluster -o yaml** 的输出检查代理是否已启用。如果 **httpProxy**、**httpsProxy**和 **noProxy** 字段设置了值，则会启用代理。

7. 在恢复 control plane 主机上运行恢复脚本，提供到 etcd 备份目录的路径：

```
$ sudo -E /usr/local/bin/cluster-restore.sh /home/core/backup
```

脚本输出示例

```
...stopping kube-scheduler-pod.yaml
...stopping kube-controller-manager-pod.yaml
...stopping etcd-pod.yaml
...stopping kube-apiserver-pod.yaml
Waiting for container etcd to stop
.complete
Waiting for container etcdctl to stop
.....complete
Waiting for container etcd-metrics to stop
complete
Waiting for container kube-controller-manager to stop
complete
Waiting for container kube-apiserver to stop
.....complete
Waiting for container kube-scheduler to stop
complete
Moving etcd data-dir /var/lib/etcd/member to /var/lib/etcd-backup
starting restore-etcd static pod
starting kube-apiserver-pod.yaml
static-pod-resources/kube-apiserver-pod-7/kube-apiserver-pod.yaml
starting kube-controller-manager-pod.yaml
```

```
static-pod-resources/kube-controller-manager-pod-7/kube-controller-manager-pod.yaml
starting kube-scheduler-pod.yaml
static-pod-resources/kube-scheduler-pod-8/kube-scheduler-pod.yaml
```



注意

如果在上次 etcd 备份后更新了节点，则恢复过程可能会导致节点进入 **NotReady** 状态。

8. 检查节点以确保它们处于 **Ready** 状态。

a. 运行以下命令：

```
$ oc get nodes -w
```

输出示例

```
NAME                STATUS ROLES    AGE  VERSION
host-172-25-75-28   Ready  master    3d20h v1.23.3+e419edf
host-172-25-75-38   Ready  infra,worker 3d20h v1.23.3+e419edf
host-172-25-75-40   Ready  master    3d20h v1.23.3+e419edf
host-172-25-75-65   Ready  master    3d20h v1.23.3+e419edf
host-172-25-75-74   Ready  infra,worker 3d20h v1.23.3+e419edf
host-172-25-75-79   Ready  worker    3d20h v1.23.3+e419edf
host-172-25-75-86   Ready  worker    3d20h v1.23.3+e419edf
host-172-25-75-98   Ready  infra,worker 3d20h v1.23.3+e419edf
```

所有节点都可能需要几分钟时间报告其状态。

b. 如果有任何节点处于 **NotReady** 状态，登录到节点，并从每个节点上的 `/var/lib/kubelet/pki` 目录中删除所有 PEM 文件。您可以 SSH 到节点，或使用 web 控制台中的终端窗口。

```
$ ssh -i <ssh-key-path> core@<master-hostname>
```

pki 目录示例

```
sh-4.4# pwd
/var/lib/kubelet/pki
sh-4.4# ls
kubelet-client-2022-04-28-11-24-09.pem kubelet-server-2022-04-28-11-24-15.pem
kubelet-client-current.pem           kubelet-server-current.pem
```

9. 在所有 control plane 主机上重启 kubelet 服务。

a. 在恢复主机中运行以下命令：

```
$ sudo systemctl restart kubelet.service
```

b. 在所有其他 control plane 主机上重复此步骤。

10. 批准待处理的 CSR：

a. 获取当前 CSR 列表：

```
$ oc get csr
```

输出示例

```
NAME      AGE  SIGNERNAME                                REQUESTOR
CONDITION
csr-2s94x 8m3s kubernetes.io/kubelet-serving             system:node:<node_name>
Pending 1
csr-4bd6t 8m3s kubernetes.io/kubelet-serving             system:node:<node_name>
Pending 2
csr-4hl85 13m  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
3
csr-zhhhp 3m8s kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
4
...
```

1 **2** 一个待处理的 kubelet 服务 CSR（用于用户置备的安装）。

3 **4** 一个待处理的 **node-bootstrapper** CSR。

b. 查看一个 CSR 的详细信息以验证其是否有效：

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

c. 批准每个有效的 **node-bootstrapper** CSR：

```
$ oc adm certificate approve <csr_name>
```

d. 对于用户置备的安装，请批准每个有效的 kubelet 服务 CSR：

```
$ oc adm certificate approve <csr_name>
```

11. 确认单个成员 control plane 已被成功启动。

a. 从恢复主机上，验证 etcd 容器是否正在运行。

```
$ sudo crictl ps | grep etcd | grep -v operator
```

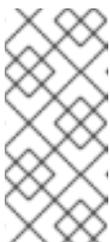
输出示例

```
3ad41b7908e32
36f86e2eeaafe662df0d21041eb22b8198e0e58abeeae8c743c3e6e977e8009
About a minute ago Running          etcd          0
7c05f8af362f0
```

b. 从恢复主机上，验证 etcd pod 是否正在运行。

■

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```



注意

如果您试图在运行这个命令前运行 **oc login** 并接收以下错误，请等待一些时间以便身份验证控制器启动并再次尝试。

```
Unable to connect to the server: EOF
```

输出示例

```
NAME                                READY STATUS  RESTARTS AGE
etcd-ip-10-0-143-125.ec2.internal  1/1  Running   1    2m47s
```

如果状态是 **Pending**，或者输出中列出了多个正在运行的 etcd pod，请等待几分钟，然后再次检查。

- c. 对不是恢复主机的每个已丢失的 control plane 主机重复此步骤。



注意

只有在使用 **OVNKubernetes** Container Network Interface (CNI) 插件时才执行以下步骤。

12. 在所有主机上重启 Open Virtual Network (OVN) Kubernetes pod。

- a. 删除北向数据库 (nbdb) 和南向数据库 (sbdb)。使用 Secure Shell (SSH) 访问恢复主机和剩余的 control plane 节点，再运行以下命令：

```
$ sudo rm -f /var/lib/ovn/etc/*.db
```

- b. 运行以下命令删除所有 OVN-Kubernetes control plane pod：

```
$ oc delete pods -l app=ovnkube-master -n openshift-ovn-kubernetes
```

- c. 运行以下命令，确保所有 OVN-Kubernetes control plane pod 已再次部署，并处于 **Running** 状态：

```
$ oc get pods -l app=ovnkube-master -n openshift-ovn-kubernetes
```

输出示例

```
NAME                                READY STATUS  RESTARTS AGE
ovnkube-master-nb24h 4/4  Running   0    48s
ovnkube-master-rm8kw 4/4  Running   0    47s
ovnkube-master-zbqnh 4/4  Running   0    56s
```

- d. 运行以下命令删除所有 **ovnkube-node** pod：

```
$ oc get pods -n openshift-ovn-kubernetes -o name | grep ovnkube-node | while read p ;
do oc delete $p -n openshift-ovn-kubernetes ; done
```

- e. 运行以下命令，确保所有 **ovnkube-node** pod 已再次部署，并处于 **Running** 状态：

```
$ oc get pods -n openshift-ovn-kubernetes | grep ovnkube-node
```

13. 逐个删除并重新创建其他非恢复 control plane 机器。重新创建机器后，会强制一个新修订版本，etcd 会自动扩展。

- 如果使用用户置备的裸机安装，您可以使用最初创建它时使用的相同方法重新创建 control plane 机器。如需更多信息，请参阅“在裸机上安装用户置备的集群”。



警告

不要为恢复主机删除并重新创建机器。

- 如果您正在运行安装程序置备的基础架构，或者您使用 Machine API 创建机器，请按照以下步骤执行：



警告

不要为恢复主机删除并重新创建机器。

对于安装程序置备的基础架构上的裸机安装，不会重新创建 control plane 机器。如需更多信息，请参阅“替换裸机控制平面节点”。

- a. 为丢失的 control plane 主机之一获取机器。
 在一个终端中使用 cluster-admin 用户连接到集群，运行以下命令：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例：

```
NAME                               PHASE  TYPE      REGION  ZONE  AGE
NODE                               PROVIDERID  STATE
clustername-8qw5l-master-0        Running m4.xlarge us-east-1 us-east-1a
3h37m ip-10-0-131-183.ec2.internal  aws:///us-east-1a/i-0ec2782f8287dfb7e
stopped 1
clustername-8qw5l-master-1        Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-143-125.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2        Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-154-194.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large  us-east-1 us-
east-1a 3h28m ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
```

```

010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-
06861c00007751b0a running

```

- 1 这是用于丢失的 control plane 主机 **ip-10-0-131-183.ec2.internal** 的 control plane 机器。

b. 将机器配置保存到文件系统中的文件中：

```

$ oc get machine clustername-8qw5l-master-0 \ 1
-n openshift-machine-api \
-o yaml \
> new-master-machine.yaml

```

- 1 为丢失的 control plane 主机指定 control plane 机器的名称。

c. 编辑上一步中创建的 **new-master-machine.yaml** 文件，以分配新名称并删除不必要的字段。

i. 删除整个 **status** 部分：

```

status:
  addresses:
    - address: 10.0.131.183
      type: InternalIP
    - address: ip-10-0-131-183.ec2.internal
      type: InternalDNS
    - address: ip-10-0-131-183.ec2.internal
      type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Node
    name: ip-10-0-131-183.ec2.internal
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: awsproviderconfig.openshift.io/v1beta1
    conditions:
      - lastProbeTime: "2020-04-20T16:53:50Z"
        lastTransitionTime: "2020-04-20T16:53:50Z"
        message: machine successfully created
        reason: MachineCreationSucceeded
        status: "True"
        type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: AWSMachineProviderStatus

```

ii. 将 **metadata.name** 字段更改为新名称。

建议您保留与旧机器相同的基础名称，并将结束号码改为下一个可用数字。在本例中，**clustername-8qw5l-master-0** 被改为 **clustername-8qw5l-master-3**：

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: clustername-8qw5l-master-3
  ...
```

iii. 删除 **spec.providerID** 字段：

```
providerID: aws:///us-east-1a/i-0fdb85790d76d0c3f
```

iv. 删除 **metadata.annotations** 和 **metadata.generation** 字段：

```
annotations:
  machine.openshift.io/instance-state: running
  ...
generation: 2
```

v. 删除 **metadata.resourceVersion** 和 **metadata.uid** 字段：

```
resourceVersion: "13291"
uid: a282eb70-40a2-4e89-8009-d05dd420d31a
```

d. 删除丢失的 control plane 主机的机器：

```
$ oc delete machine -n openshift-machine-api clustername-8qw5l-master-0 1
```

1 为丢失的 control plane 主机指定 control plane 机器的名称。

e. 验证机器是否已删除：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例：

```
NAME                                PHASE  TYPE      REGION  ZONE  AGE
NODE                                PROVIDERID  STATE
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-143-125.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-154-194.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large  us-east-1 us-
east-1a 3h28m ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large  us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal  aws:///us-east-1b/i-
0cb45ac45a166173b running
```

```

clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-
06861c00007751b0a running

```

- f. 使用 **new-master-machine.yaml** 文件创建新机器：

```
$ oc apply -f new-master-machine.yaml
```

- g. 验证新机器是否已创建：

```
$ oc get machines -n openshift-machine-api -o wide
```

输出示例：

```

NAME                                PHASE    TYPE    REGION  ZONE
AGE  NODE                                PROVIDERID  STATE
clustername-8qw5l-master-1          Running   m4.xlarge us-east-1 us-east-
1b 3h37m ip-10-0-143-125.ec2.internal aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running   m4.xlarge us-east-1 us-east-
1c 3h37m ip-10-0-154-194.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-master-3          Provisioning m4.xlarge us-east-1 us-east-
1a 85s ip-10-0-173-171.ec2.internal aws:///us-east-1a/i-015b0888fe17bc2c8
running ①
clustername-8qw5l-worker-us-east-1a-wbtgd Running   m4.large us-east-1
us-east-1a 3h28m ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running   m4.large us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running   m4.large us-east-1
us-east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-
06861c00007751b0a running

```

- ① 新机器 **clustername-8qw5l-master-3** 会被创建，并在阶段从 **Provisioning** 变为 **Running** 后就绪。

创建新机器可能需要几分钟时间。当机器或节点返回一个健康状态时，etcd cluster Operator 将自动同步。

- h. 对不是恢复主机的每个已丢失的 control plane 主机重复此步骤。

14. 在一个单独的终端窗口中，使用以下命令以具有 **cluster-admin** 角色的用户身份登录到集群：

```
$ oc login -u <cluster_admin> ①
```

- ① 对于 **<cluster_admin>**，使用 **cluster-admin** 角色指定一个用户名。

15. 强制 etcd 重新部署。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc patch etcd cluster -p='{ "spec": { "forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"' --type=merge 1
```

- 1 **forceRedeploymentReason** 值必须是唯一的，这就是为什么附加时间戳的原因。

当 etcd cluster Operator 执行重新部署时，现有节点开始使用与初始 bootstrap 扩展类似的新 pod。

16. 验证所有节点是否已更新至最新的修订版本。
在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[? (@.type=="NodeInstallerProgressing")]}{.reason}{ "\n"}{.message}{ "\n"}'
```

查看 etcd 的 **NodeInstallerProgressing** 状态条件，以验证所有节点是否处于最新的修订。在更新成功后，输出会显示 **AllNodesAtLatestRevision**：

```
AllNodesAtLatestRevision  
3 nodes are at revision 7 1
```

- 1 在本例中，最新的修订版本号是 7。

如果输出包含多个修订号，如 2 个节点为修订版本 6；1 个节点为修订版本 7，这意味着更新仍在进行中。等待几分钟后重试。

17. 在重新部署 etcd 后，为 control plane 强制进行新的 rollout。由于 kubelet 使用内部负载均衡器连接到 API 服务器，因此 Kubernetes API 将在其他节点上重新安装自己。
在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令。

- a. 为 Kubernetes API 服务器强制进行新的推出部署：

```
$ oc patch kubeapiserver cluster -p='{ "spec": { "forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"' --type=merge
```

验证所有节点是否已更新至最新的修订版本。

```
$ oc get kubeapiserver -o=jsonpath='{range .items[0].status.conditions[? (@.type=="NodeInstallerProgressing")]}{.reason}{ "\n"}{.message}{ "\n"}'
```

查看 **NodeInstallerProgressing** 状态条件，以验证所有节点是否处于最新版本。在更新成功后，输出会显示 **AllNodesAtLatestRevision**：

```
AllNodesAtLatestRevision  
3 nodes are at revision 7 1
```

- 1 在本例中，最新的修订版本号是 7。

如果输出包含多个修订号，如 2 个节点为修订版本 6；1 个节点为修订版本 7，这意味着更新仍在进行中。等待几分钟后重试。

- b. 为 Kubernetes 控制器管理器强制进行新的推出部署：

```
$ oc patch kubecontrollermanager cluster -p='{ "spec": {"forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"'}}' --type=merge
```

验证所有节点是否已更新至最新的修订版本。

```
$ oc get kubecontrollermanager -o=jsonpath='{range .items[0].status.conditions[?(@.type=="NodeInstallerProgressing")]}{.reason}{ "\n"}{.message}{ "\n"}'
```

查看 **NodeInstallerProgressing** 状态条件，以验证所有节点是否处于最新版本。在更新成功后，输出会显示 **AllNodesAtLatestRevision**：

```
AllNodesAtLatestRevision
3 nodes are at revision 7 1
```

1 在本例中，最新的修订版本号是 7。

如果输出包含多个修订号，如 **2** 个节点为修订版本 **6**；**1** 个节点为修订版本 **7**，这意味着更新仍在进行中。等待几分钟后重试。

c. 为 Kubernetes 调度程序强制进行新的推出部署：

```
$ oc patch kubescheduler cluster -p='{ "spec": {"forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"'}}' --type=merge
```

验证所有节点是否已更新至最新的修订版本。

```
$ oc get kubescheduler -o=jsonpath='{range .items[0].status.conditions[?(@.type=="NodeInstallerProgressing")]}{.reason}{ "\n"}{.message}{ "\n"}'
```

查看 **NodeInstallerProgressing** 状态条件，以验证所有节点是否处于最新版本。在更新成功后，输出会显示 **AllNodesAtLatestRevision**：

```
AllNodesAtLatestRevision
3 nodes are at revision 7 1
```

1 在本例中，最新的修订版本号是 7。

如果输出包含多个修订号，如 **2** 个节点为修订版本 **6**；**1** 个节点为修订版本 **7**，这意味着更新仍在进行中。等待几分钟后重试。

18. 验证所有 control plane 主机是否已启动并加入集群。

在一个终端中使用 **cluster-admin** 用户连接到集群，运行以下命令：

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

输出示例

```
etcd-ip-10-0-143-125.ec2.internal    2/2   Running   0    9h
etcd-ip-10-0-154-194.ec2.internal    2/2   Running   0    9h
etcd-ip-10-0-173-171.ec2.internal    2/2   Running   0    9h
```

为确保所有工作负载在恢复过程后返回到正常操作，请重启存储 Kubernetes API 信息的每个 pod。这包括 OpenShift Container Platform 组件，如路由器、Operator 和第三方组件。

请注意，在完成这个过程后，可能需要几分钟才能恢复所有服务。例如，在重启 OAuth 服务器 pod 前，使用 **oc login** 进行身份验证可能无法立即正常工作。

5.3.2.3. 其他资源

- [在裸机上安装用户置备的集群](#)
- [创建堡垒主机以通过 SSH 访问 OpenShift Container Platform 实例和 control plane 节点。](#)
- [替换裸机 control plane 节点](#)

5.3.2.4. 恢复持久性存储状态的问题和解决方法

如果您的 OpenShift Container Platform 集群使用任何形式的持久性存储，集群的状态通常存储在 etcd 外部。它可能是在 pod 中运行的 Elasticsearch 集群，或者在 **StatefulSet** 对象中运行的数据库。从 etcd 备份中恢复时，还会恢复 OpenShift Container Platform 中工作负载的状态。但是，如果 etcd 快照是旧的，其状态可能无效或过期。



重要

持久性卷 (PV) 的内容绝不会属于 etcd 快照的一部分。从 etcd 快照恢复 OpenShift Container Platform 集群时，非关键工作负载可能会访问关键数据，反之亦然。

以下是生成过时状态的一些示例情况：

- MySQL 数据库在由 PV 对象支持的 pod 中运行。从 etcd 快照恢复 OpenShift Container Platform 不会使卷恢复到存储供应商上，且不会生成正在运行的 MySQL pod，尽管 pod 会重复尝试启动。您必须通过在存储供应商中恢复卷，然后编辑 PV 以指向新卷来手动恢复这个 pod。
- Pod P1 使用卷 A，它附加到节点 X。如果另一个 pod 在节点 Y 上使用相同的卷，则执行 etcd 恢复时，pod P1 可能无法正确启动，因为卷仍然被附加到节点 Y。OpenShift Container Platform 并不知道附加，且不会自动分离它。发生这种情况时，卷必须从节点 Y 手动分离，以便卷可以在节点 X 上附加，然后 pod P1 才可以启动。
- 在执行 etcd 快照后，云供应商或存储供应商凭证会被更新。这会导致任何依赖于这些凭证的 CSI 驱动程序或 Operator 无法正常工作。您可能需要手动更新这些驱动程序或 Operator 所需的凭证。
- 在生成 etcd 快照后，会从 OpenShift Container Platform 节点中删除或重命名设备。Local Storage Operator 会为从 `/dev/disk/by-id` 或 `/dev` 目录中管理的每个 PV 创建符号链接。这种情况可能会导致本地 PV 引用不再存在的设备。
要解决这个问题，管理员必须：
 1. 手动删除带有无效设备的 PV。
 2. 从对应节点中删除符号链接。
 3. 删除 **LocalVolume** 或 **LocalVolumeSet** 对象（请参阅 *Storage → Configuring persistent storage → Persistent storage → Persistent storage → Deleting the Local Storage Operator Resources*）。

5.3.3. 从 control plane 证书已过期的情况下恢复

5.3.3.1. 从 control plane 证书已过期的情况下恢复

集群可以从过期的 control plane 证书中自动恢复。

但是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。对于用户置备的安装，您可能需要批准待处理的 kubelet 服务 CSR。

使用以下步骤批准待处理的 CSR：

流程

1. 获取当前 CSR 列表：

```
$ oc get csr
```

输出示例

```
NAME      AGE  SIGNERNAME                                REQUESTOR
CONDITION
csr-2s94x  8m3s  kubernetes.io/kubelet-serving             system:node:<node_name>
Pending 1
csr-4bd6t  8m3s  kubernetes.io/kubelet-serving             system:node:<node_name>
Pending 2
csr-4hl85  13m   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper  Pending 3
csr-zhphp  3m8s  kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper  Pending 4
...
```

1 2 一个待处理的 kubelet 服务 CSR（用于用户置备的安装）。

3 4 一个待处理的 **node-bootstrapper** CSR。

2. 查看一个 CSR 的详细信息以验证其是否有效：

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

3. 批准每个有效的 **node-bootstrapper** CSR：

```
$ oc adm certificate approve <csr_name>
```

4. 对于用户置备的安装，请批准每个有效的 kubelet 服务 CSR：

```
$ oc adm certificate approve <csr_name>
```