Red Hat Advanced Cluster Management for Kubernetes 2.7

Observability（可观察性）

请参阅更多信息，了解如何启用和自定义可观察性服务来优化受管集群。

Last Updated: 2024-06-12

Red Hat Advanced Cluster Management for Kubernetes 2.7 Observability（可观察性）

请参阅更多信息，了解如何启用和自定义可观察性服务来优化受管集群。

法律通告

摘要

请参阅更多信息，了解如何启用和自定义可观察性服务来优化受管集群。

目录

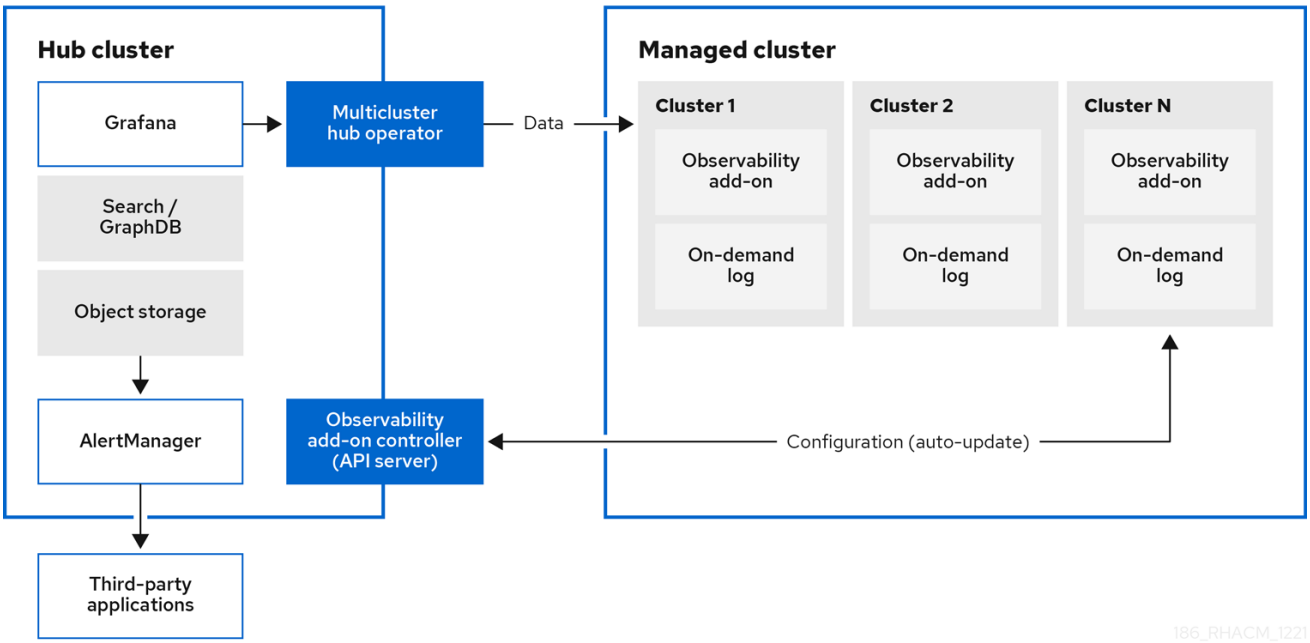
第 1 章 观察环境简介

启用可观察（observability）服务后，您可以使用 Red Hat Advanced Cluster Management for Kubernetes 深入了解受管集群并进行优化。这些信息可以帮助节约成本并防止不必要的事件发生。

- 观察环境
- 启用可观察性服务
- 在控制台简介中搜索
- 定制可观察性
- 设计您的 Grafana 仪表板
- 在 Red Hat Insights 中使用可观察性
- 管理 insight PolicyReports

1.1. 观察环境

您可以使用 Red Hat Advanced Cluster Management for Kubernetes 深入了解受管集群并进行优化。在 hub 集群中启用 observability 服务 operator(**multicluster-observability-operator**)以监控受管集群的健康状态。在以下部分了解多集群观察服务的架构。



注：*on-demand 日志*可让工程师实时获取给定 pod 的日志。hub 集群的日志不会被聚合。这些日志可以通过搜索服务以及控制台的其他部分进行访问。

- 观察（Observability）服务
- 支持
- 指标类型
- Observability pod 容量请求

- Observability 服务中使用的持久性存储
- 相关资源

1.1.1. 观察（Observability）服务

默认情况下，产品安装中包含了可观察性（observability）功能，但不启用它。由于对持久性存储的要求，observability 服务默认不会启用。有关可观察性的信息，请参阅 *支持* 部分。

启用该服务后，**observability-endpoint-operator** 会自动部署到每个导入或创建的集群中。此控制器从 Red Hat OpenShift Container Platform Prometheus 收集数据，然后将其发送到 Red Hat Advanced Cluster Management hub 集群。如果 hub 集群导入为 **local-cluster**，则在它上也会启用可观察性，并从 hub 集群收集指标。

observability 服务部署了一个 Prometheus AlertManager 实例，它允许通过第三方应用程序转发警报。它还包括一个 Grafana 实例，通过仪表板（静态）或数据探索启用数据视觉化。Red Hat Advanced Cluster Management 支持 Grafana 的版本 8.1.3。您还可以设计自己的 Grafana 仪表板。如需更多信息，请参阅指定 Grafana 仪表板。您可以通过创建自定义记录规则或警报规则来自定义可观察性服务。

1.1.2. 支持

- Red Hat Advanced Cluster Management 已测试并被 Red Hat OpenShift Data Foundation（以前称为 Red Hat OpenShift Container Storage）完全支持。
- Red Hat Advanced Cluster Management 支持在用户提供的兼容 S3 API 的第三方对象存储中多集群可观察 Operator 的功能。Observability 服务使用 Thanos 支持的、稳定的对象存储。
- Red Hat Advanced Cluster Management 使用商业、合理的努力来帮助识别根本原因。如果创建了相关的支持问题，且确定问题的根本原因是客户提供的兼容 S3 对象存储，则需要通过客户支持频道解决问题。
- Red Hat Advanced Cluster Management 不会承诺修复客户提供的、问题的根本原因是 S3 兼容对象存储供应商的支持问题单。

1.1.3. 指标类型

默认情况下，OpenShift Container Platform 使用 Telemetry 服务向红帽发送指标数据。**acm_managed_cluster_info** 由 Red Hat Advanced Cluster Management 提供，包含在 Telemetry 中，但 *不会* 显示在 Red Hat Advanced Cluster Management *Observe 环境概述* 仪表板中。

查看框架支持的指标类型表：

表 1.1. 参数表

指标名称	指标类型	标签/标签	Status
acm_managed_cluster_info	量表	**hub_cluster_id**, **managed_cluster_id**, **vendor**, **cloud**, **version**, **available**, **created_via**, **core_worker**, **socket_worker**	稳定

指标名称	指标类型	标签/标签	Status
policy_governance_info	量表	**type**, **policy**, **policy_namespace**, **cluster_namespace**	稳定。如需了解更多详细信息，请参阅监管指标。
policyreport_info	量表	**managed_cluster_id**, **category**, **policy**, **result**, **severity**	稳定。如需了解更多详细信息，请参阅管理 Insights PolicyReports。
config_policies_evaluation_duration_seconds_bucket	Histogram	无。	稳定。如需了解更多详细信息，请参阅监管指标。
config_policies_evaluation_duration_seconds_count	Histogram	无。	稳定。如需了解更多详细信息，请参阅监管指标。
config_policies_evaluation_duration_seconds_sum	Histogram	无。	稳定。如需了解更多详细信息，请参阅监管指标。

1.1.4. Observability pod 容量请求

Observability 组件需要 2701mCPU 和 11972Mi 内存来安装可观察性服务。下表是启用了 **observability-addons** 的五个受管集群的 pod 容量请求列表：

表 1.2. Observability pod 容量请求

Deployment 或 StatefulSet	容器名称	CPU（mCPU）	内存（Mi）	Replicas	Pod 总计 CPU	Pod 内存总量
observability-alertmanager	alertmanager	4	200	3	12	600
	config-reloader	4	25	3	12	75
	alertmanager-proxy	1	20	3	3	60
observability-grafana	grafana	4	100	2	8	200
	grafana-dashboard-loader	4	50	2	8	100

Deployment 或 StatefulSet	容器名称	CPU（mCPU）	内存（Mi）	Replicas	Pod 总计 CPU	Pod 内存总量
observability-observatorium-api	observatorium-api	20	128	2	40	256
observability-observatorium-operator	observatorium-operator	100	100	1	10	50
observability-rbac-query-proxy	rbac-query-proxy	20	100	2	40	200
	oauth-proxy	1	20	2	2	40
observability-thanos-compact	thanos-compact	100	512	1	100	512
observability-thanos-query	thanos-query	300	1024	2	600	2048
observability-thanos-query-frontend	thanos-query-frontend	100	256	2	200	512
observability-thanos-query-frontend-memcached	memcached	45	128	3	135	384
	exporter	5	50	3	15	150
observability-thanos-receive-controller	thanos-receive-controller	4	32	1	4	32
observability-thanos-receive-default	thanos-receive	300	512	3	900	1536

Deployment 或 StatefulSet	容器名称	CPU（mCPU）	内存（Mi）	Replicas	Pod 总计 CPU	Pod 内存总量
observability-thanos-rule	thanos-rule	50	512	3	150	1536
	configmap-reloader	4	25	3	12	75
observability-thanos-store-memcached	memcached	45	128	3	135	384
	exporter	5	50	3	15	150
observability-thanos-store-shard	thanos-store	100	1024	3	300	3072

1.1.5. Observability 服务中使用的持久性存储

安装 Red Hat Advanced Cluster Management 时，必须创建以下持久性卷(PV)，以便 PVC 可以自动附加到 PVC。请注意，当没有指定默认存储类或想要使用非默认存储类来托管 PV 时，您必须在 **MultiClusterObservability** CR 中定义存储类。建议您使用 Block Storage，类似于 Prometheus 使用的存储。另外，**alertmanager**、**thanos-compactor**、**thanos-ruler**、**thanos-receive-default** 和 **thanos-store-shard** 的每个副本必须具有自己的 PV。查看下表：

表 1.3. 持久性卷表列表

持久性卷名称	目的
alertmanager	Alertmanager 将 **nflog** 数据和静默的警报信息存储在它的存储中。**nflog** 是一个只能附加的日志，它包括当前有效的和已解决的日志，以及通知的接收者和用来识别内容的哈希摘要数据。
thanos-compact	紧凑器需要本地磁盘空间来存储用于处理的中间数据，以及 bucket 状态缓存。所需空间取决于基础块的大小。紧凑器必须有足够的空间下载所有源块，然后在磁盘上构建紧凑块。磁盘上的数据可以安全地在重新启动之间删除，并且应该是首次尝试使崩溃循环解压器。不过，建议为紧凑器永久磁盘提供压缩器持久磁盘，以便在重启期间高效地使用存储桶状态缓存。
thanos-rule	thanos 标尺通过以固定间隔发出查询来评估 Prometheus 记录和警报规则。规则结果以 Prometheus 2.0 存储格式写回磁盘。在这个有状态集合中保留的数据量在 API 版本 **observability.open-cluster-management.io/v1beta1** 中被修复。它在 **observability.open-cluster-management.io/v1beta2**: ***RetentionInLocal*** 中以一个 API 参数的形式公开

thanos-receive-default	Thanos 接收器接受传入数据（Prometheus 远程写入请求），并将这些数据写入 Prometheus TSDB 的一个本地实例。TSDB 块定期（每 2 小时）上传到对象存储，以进行长期存储和压缩。在这个有状态集合中保留的小时数或天数。有状态集合作为一个本地的缓存，它在 API 版本 **observability.open-cluster-management.io/v1beta** 中被修复。它在 **observability.open-cluster-management.io/v1beta2**: ***RetentionInLocal*** 中以一个 API 参数的形式公开
thanos-store-shard	它主要充当一个 API 网关，因此不需要大量的本地磁盘空间。它在启动时加入 Thanos 集群，并公告它可以访问的数据。它在本地磁盘上保留少量的、与所有远程块相关的信息，并与存储桶保持同步。这些数据通常在重启时会被安全地删除，这会增加启动时间。

注意 ：时间序列历史数据存储在对象存储中。Thanos 使用对象存储作为指标和与其相关的元数据的主存储。有关对象存储和降级的详情，请参阅启用可观察性服务。

1.1.6. 相关资源

- 有关启用可观察性的更多信息，请参阅启用可观察性服务。
- 请参阅 自定义可观察性 以了解如何配置可观察性服务、查看指标和其他数据。
- 从 OpenShift Container Platform 文档中了解使用遥测来收集并发送哪些指标类型。如需更多信息，请参阅 Telemetry 收集的信息。
- 请参阅 Prometheus 记录规则。
- 另请参阅 Prometheus 警报规则

1.2. 启用可观察性服务

监控使用 observability 服务（**multicluster-observability-operator**）的受管集群的监控状态。

需要的访问权限： 集群管理员、**open-cluster-management:cluster-manager-admin** 角色或 S3 管理员。

- 先决条件
- 启用可观察性
- 创建 MultiClusterObservability 自定义资源
- 从 Red Hat OpenShift Container Platform 控制台启用可观察性
- 使用外部指标查询
- 禁用可观察性

1.2.1. 先决条件

- 您必须安装 Red Hat Advanced Cluster Management for Kubernetes。如需更多信息，请参阅在线安装。
- 如果没有指定默认存储类，则必须在 **MultiClusterObservability** 自定义资源中定义存储类。
- 需要直接网络访问 hub 集群。不支持对负载均衡器和代理的网络访问。如需更多信息，请参阅网络。
- 您必须配置对象存储来创建存储解决方案。Red Hat Advanced Cluster Management 支持带有稳定对象存储的以下云供应商：
  - Amazon Web Services S3 (AWS S3)
  - Red Hat Ceph (S3 compatible API)
  - Google Cloud Storage
  - Azure 存储
  - Red Hat OpenShift Data Foundation (以前称为 Red Hat OpenShift Container Storage)
  - Red Hat OpenShift on IBM (ROKS)
    重要：当您配置对象存储时，请确保满足敏感数据持久时所需的加密要求。Observability 服务使用 Thanos 支持的、稳定的对象存储。

1.2.2. 启用可观察性

通过创建一个 **MultiClusterObservability** 自定义资源实例来启用可观察性服务。在启用可观察性前，请参阅 Observability pod 容量请求以了解更多信息。

注 ：当由 Red Hat Advanced Cluster Management 管理的 OpenShift Container Platform 受管集群上启用或禁用了可观察性时，observability 端点 Operator 会添加额外的 **alertmanager** 配置来自动重启本地 Prometheus，以此更新 **cluster-monitoring-config** ConfigMap。

完成以下步骤以启用可观察服务：

1. 登录到您的 Red Hat Advanced Cluster Management hub 集群。
2. 使用以下命令，为可观察服务创建一个命名空间：

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

3. 生成 pull-secret。如果在 **open-cluster-management** 命名空间中安装了 Red Hat Advanced Cluster Management，请运行以下命令：

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

   如果命名空间中没有定义 **multiclusterhub-operator-pull-secret**，将 **openshift-config** 命名空间中的 **pull-secret** 复制到 **open-cluster-management-observability** 命名空间中。运行以下命令:

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

   然后，在 **open-cluster-management-observability** 命名空间中创建 pull-secret，运行以下命令：

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

重要：如果使用 OpenShift Container Platform 文档修改集群的全局 pull secret，请务必更新可观察命名空间中的全局 pull secret。如需了解更多详细信息，请参阅更新全局 pull secret。

4. 为您的云供应商的对象存储创建 secret。您的 secret 必须包含存储解决方案的凭证。例如，运行以下命令：

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

查看以下受支持对象存储的 secret 示例：

- 对于 Amazon S3 或 S3 兼容，您的 secret 可能类似以下文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_S3_BUCKET
      endpoint: YOUR_S3_ENDPOINT 1
      insecure: true
      access_key: YOUR_ACCESS_KEY
      secret_key: YOUR_SECRET_KEY
```

1 输入没有协议部分的 URL。输入类似以下 URL 的 Amazon S3 端点的 URL：**example.redhat.com:443**。

如需了解更多详细信息，请参阅 Amazon Simple Storage Service 用户指南 。

详情请参阅 Amazon Simple Storage Service 用户指南 。

- 对于 Google，您的 secret 可能类似以下文件：

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

如需了解更多详细信息，请参阅 Google Cloud Storage。

- 对于 Azure，您的 secret 可能类似以下文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: AZURE
    config:
      storage_account: YOUR_STORAGE_ACCT
      storage_account_key: YOUR_STORAGE_KEY
      container: YOUR_CONTAINER
      endpoint: blob.core.windows.net (1)
      max_retries: 0
```

(1) 如果使用 **msi_resource** 路径，则端点身份验证通过使用 system-assigned 受管身份完成。您的值必须类似以下端点：**https://<storage-account-name>.blob.core.windows.net**。

如果您使用 **user_assigned_id** 路径，则端点身份验证通过使用用户分配的受管身份完成。当您使用 **user_assigned_id** 时，**msi_resource** 端点的默认值为 **https://<storage_account>.<endpoint>**。如需了解更多详细信息，请参阅 Azure Storage 文档。

注 ：如果您将 Azure 用作 Red Hat OpenShift Container Platform 集群的对象存储，则不支持与集群关联的存储帐户。您必须创建新存储帐户。

- 对于 Red Hat OpenShift Data Foundation，您的 secret 可能类似以下文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_RH_DATA_FOUNDATION_BUCKET
      endpoint: YOUR_RH_DATA_FOUNDATION_ENDPOINT (1)
      insecure: false
      access_key: YOUR_RH_DATA_FOUNDATION_ACCESS_KEY
      secret_key: YOUR_RH_DATA_FOUNDATION_SECRET_KEY
```

(1) 输入没有协议部分的 URL。输入您的 Red Hat OpenShift Data Foundation 端点的 URL，它可能类似以下 URL：**example.redhat.com:443**。

如需了解更多详细信息，请参阅 Red Hat OpenShift Data Foundation。

- 对于 IBM 上的 Red Hat OpenShift (ROKS)，您的 secret 可能类似以下文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
    type: s3
    config:
      bucket: YOUR_ROKS_S3_BUCKET
      endpoint: YOUR_ROKS_S3_ENDPOINT 1
      insecure: true
      access_key: YOUR_ROKS_ACCESS_KEY
      secret_key: YOUR_ROKS_SECRET_KEY
```

1 输入没有协议部分的 URL。输入您的 Red Hat OpenShift Data Foundation 端点的 URL，它可能类似以下 URL：**example.redhat.com:443**。

如需了解更多详细信息，请参阅 IBM 云文档 Cloud Object Storage。务必使用服务凭据来连接对象存储。如需了解更多详细信息，请参阅 IBM Cloud 文档，云对象存储和服务凭证。

- 对于 Amazon S3 或 S3 兼容存储，您还可以使用由 AWS 安全令牌服务(AWS STS)生成的简短的、有有限权限的凭证。如需了解更多详细信息，请参阅 AWS 安全令牌服务 文档。
使用 AWS 安全服务生成访问密钥需要以下额外步骤：
  - 创建一个 IAM 策略，限制对 S3 存储桶的访问。
  - 使用信任策略创建 IAM 角色，为 OpenShift Container Platform 服务帐户生成 JWT 令牌
  - 为需要访问 S3 存储桶的可观察服务帐户指定注解。您可以参阅如何在 AWS (ROSA)集群上使用 Red Hat OpenShift Service on AWS (ROSA)集群的可观察性，以便在 *Set 环境* 步骤中使用 AWS STS 令牌。如需了解更多详细信息，请参阅 Red Hat OpenShift Service on AWS (ROSA)，以及 ROSA with STS explained 了解有关使用 STS 令牌的要求和设置的信息。

完成以下步骤，使用 AWS 安全服务生成访问密钥：

1. 设置 AWS 环境。运行以下命令：

```
export POLICY_VERSION=$(date +"%m-%d-%y")
export TRUST_POLICY_VERSION=$(date +"%m-%d-%y")
export CLUSTER_NAME=<my-cluster>
export S3_BUCKET=$CLUSTER_NAME-acm-observability
export REGION=us-east-2
export NAMESPACE=open-cluster-management-observability
export SA=tbd
export SCRATCH_DIR=/tmp/scratch
export OIDC_PROVIDER=$(oc get authentication.config.openshift.io cluster -o json | jq -r
.spec.serviceAccountIssuer| sed -e "s/^https:\/\//")
export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

```
export AWS_PAGER=""
rm -rf $SCRATCH_DIR
mkdir -p $SCRATCH_DIR
```

2. 使用以下命令创建 S3 存储桶：

```
aws s3 mb s3://$S3_BUCKET
```

3. 创建一个 **s3-policy** JSON 文件来访问 S3 存储桶。运行以下命令:

```
{
   "Version": "$POLICY_VERSION",
   "Statement": [
      {
         "Sid": "Statement",
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket",
            "s3:GetObject",
            "s3:DeleteObject",
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:CreateBucket",
            "s3:DeleteBucket"
         ],
         "Resource": [
            "arn:aws:s3:::$S3_BUCKET/*",
            "arn:aws:s3:::$S3_BUCKET"
         ]
      }
   ]
}
```

4. 使用以下命令应用策略：

```
S3_POLICY=$(aws iam create-policy --policy-name $CLUSTER_NAME-acm-obs \
--policy-document file://$SCRATCH_DIR/s3-policy.json \
--query 'Policy.Arn' --output text)
echo $S3_POLICY
```

5. 创建 **TrustPolicy** JSON 文件。运行以下命令:

```
{
 "Version": "$TRUST_POLICY_VERSION",
 "Statement": [
   {
    "Effect": "Allow",
    "Principal": {
      "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-provider/${OIDC_PROVIDER}"
    },
    "Action": "sts:AssumeRoleWithWebIdentity",
    "Condition": {
      "StringEquals": {
        "${OIDC_PROVIDER}:sub": [
```

```
            "system:serviceaccount:${NAMESPACE}:observability-thanos-query",
            "system:serviceaccount:${NAMESPACE}:observability-thanos-store-shard",
            "system:serviceaccount:${NAMESPACE}:observability-thanos-compact"
            "system:serviceaccount:${NAMESPACE}:observability-thanos-rule",
            "system:serviceaccount:${NAMESPACE}:observability-thanos-receive",
          ]
        }
      }
    }
  ]
}
```

6. 使用以下命令，为 AWS Prometheus 和 CloudWatch 创建角色：

```
S3_ROLE=$(aws iam create-role \
  --role-name "$CLUSTER_NAME-acm-obs-s3" \
  --assume-role-policy-document file://$SCRATCH_DIR/TrustPolicy.json \
  --query "Role.Arn" --output text)
echo $S3_ROLE
```

7. 将策略附加到角色。运行以下命令:

```
aws iam attach-role-policy \
  --role-name "$CLUSTER_NAME-acm-obs-s3" \
  --policy-arn $S3_POLICY
```

您的 secret 可能类似以下文件：**config** 部分指定 **signature_version2: false**，且不指定 **access_key** 和 **secret_key** ：

```
apiVersion: v1
kind: Secret
metadata:
  name: thanos-object-storage
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  thanos.yaml: |
 type: s3
 config:
   bucket: $S3_BUCKET
   endpoint: s3.$REGION.amazonaws.com
   signature_version2: false
```

8. 在 **MultiClusterObservability** 自定义资源时指定服务帐户注解，如 *创建 MultiClusterObservability CR* 部分所述。

9. 您可以使用以下命令为云供应商检索 S3 access key 和 secret 密钥：您必须在 secret 中对 **base64** 字符串进行解码、编辑和编码：

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

echo $ACCESS_KEY
```

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

echo $SECRET_KEY
```

10. 通过检查以下部署和有状态集的 pod 来验证是否启用了可观察性。您可能会收到以下信息：

```
observability-thanos-query (deployment)
observability-thanos-compact (statefulset)
observability-thanos-receive-default  (statefulset)
observability-thanos-rule  (statefulset)
observability-thanos-store-shard-x  (statefulsets)
```

1.2.2.1. 创建 MultiClusterObservability 自定义资源

使用 **MultiClusterObservability** 自定义资源为各种组件指定持久性卷存储大小。您必须在初始创建 **MultiClusterObservability** 自定义资源时设置存储大小。当您部署后更新存储大小值时，只有在存储类支持动态卷扩展时，更改才会生效。如需更多信息，请参阅 Red Hat OpenShift Container Platform 文档中的 扩展持久性卷。

完成以下步骤，在 hub 集群中创建 **MultiClusterObservability** 自定义资源：

1. 创建名为 ***multiclusterobservability_cr.yaml*** 的 **MultiClusterObservability** 自定义资源 YAML 文件。
   查看以下默认 YAML 文件以查看可观察性：

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

   您可能需要修改 **advanced** 部分中的 **retentionConfig** 参数的值。如需更多信息，请参阅 Thanos Downsampling 分辨率和保留时间。根据受管集群的数量，您可能需要为有状态的集合更新存储量。如果您的 S3 存储桶被配置为使用 STS 令牌，请给服务帐户通过 S3 角色使用 STS。
   查看以下配置：

```
spec:
  advanced:
    compact:
      serviceAccountAnnotations:
        eks.amazonaws.com/role-arn: $S3_ROLE
    store:
      serviceAccountAnnotations:
        eks.amazonaws.com/role-arn: $S3_ROLE
    rule:
      serviceAccountAnnotations:
        eks.amazonaws.com/role-arn: $S3_ROLE
```

```
    receive:
      serviceAccountAnnotations:
        eks.amazonaws.com/role-arn: $S3_ROLE
    query:
      serviceAccountAnnotations:
        eks.amazonaws.com/role-arn: $S3_ROLE
```

如需更多信息，请参阅 Observability API。

2. 要在基础架构机器集上部署，您必须通过更新 **MultiClusterObservability** YAML 中的 **nodeSelector** 来为设置设置一个标签。您的 YAML 可能类似以下内容：

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

如需更多信息，请参阅创建基础架构机器集。

3. 运行以下命令，将可观察 YAML 应用到集群：

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

用于 Thanos、Grafana 和 AlertManager 的所有 pod 在 **open-cluster-management-observability** 命名空间中创建。所有连接到 Red Hat Advanced Cluster Management hub 集群的受管集群都会被启用，以将指标数据发送回 Red Hat Advanced Cluster Management Observability 服务。

4. 通过启动 Grafana 仪表板来验证 observability 服务是否已启用，并且数据是否填充。在控制台 *Overview* 页面或 *Clusters* 页面点击位于控制台标头旁的 **Grafana 链接**。
注：如果要排除特定的受管集群收集可观察性数据,请在集群中添加以下集群标签：**observability: disabled**。

observability 服务被启用。启用 observability 服务后，会启动以下功能：

- 所有来自受管集群的警报管理器都转发到 Red Hat Advanced Cluster Management hub 集群。
- 所有连接到 Red Hat Advanced Cluster Management hub 集群的受管集群都会被启用，以将警报发送回 Red Hat Advanced Cluster Management observability 服务。您可以配置 Red Hat Advanced Cluster Management Alertmanager 来处理重复数据删除、分组和将警报路由到正确的接收器集成，如电子邮件、PagerDuty 或 OpsGenie。您还可以处理静默和禁止警报。
注 ： 只有 Red Hat OpenShift Container Platform 版本 4.8 或更高版本的受管集群支持将警报转发到 Red Hat Advanced Cluster Management hub 集群功能。在安装启用了可观察性服务的 Red Hat Advanced Cluster Management 后，来自 OpenShift Container Platform v4.8 及更新的版本的警报会自动转发到 hub 集群。请参阅转发警报以了解更多信息。
    - 使用以下 URL 访问 OpenShift Container Platform 3.11 Grafana 仪表板：**https://$ACM_URL/grafana/dashboards**。选择名为 *OCP 3.11* 的文件夹来查看 OpenShift Container Platform 3.11 仪表板。

1.2.3. 从 Red Hat OpenShift Container Platform 控制台启用可观察性

另外，您还可以从 Red Hat OpenShift Container Platform 控制台启用可观察性，创建一个名为 **open-cluster-management-observability** 的项目。务必在 **open-cluster-management-observability** 项目中创建名为 **multiclusterhub-operator-pull-secret** 的镜像 pull-secret。

在 **open-cluster-management-observability** 项目中创建名为 **thanos-object-storage** 的对象存储 secret, 。输入对象存储 secret 详细信息，然后单击 **Create**。请参阅 *Enabling observability* 部分的第 4 步来查看 secret 的示例。

创建 **MultiClusterObservability** 自定义资源实例。当您收到以下信息时，代表 OpenShift Container Platform 中已成功启用 obseravbility 服务：**Observability components are deployed and running**。

1.2.3.1. 使用外部指标查询

Observability 提供了一个外部 API，用于通过 OpenShift 路由（**rbac-query-proxy**）查询指标。查看以下使用 **userbac-query-proxy** 路由的任务：

- 您可以使用以下命令获取路由的详情：

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

- 若要访问 **therbac-query-proxy** 路由，您必须具有 OpenShift OAuth 访问令牌。该令牌应当与用户或服务帐户关联，该帐户有权获取命名空间。如需更多信息，请参阅管理用户拥有的 OAuth 访问令牌。

- 获取默认 CA 证书，并将密钥 **tls.crt** 的内容存储在本地文件中。运行以下命令:

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

- 运行以下命令以查询指标：

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

  注：QUERY_EXPRESSION 是标准的 Prometheus 查询表达式。例如，通过将前面提到的命令中的 URL 替换为以下 URL：**https://{PROXY_ROUTE_URL}/api/v1/query?query=cluster_infrastructure_provider** 来查询指标 **cluster_infrastructure_provider**。如需了解更多详细信息，请参阅查询 Prometheus。

- 您还可以替换 **therbac-query-proxy** 路由的证书：请参阅 OpenSSL 命令来生成 CA 证书以 创建证书。当您自定义 **csr.cnf** 时，将 **DNS.1** 更新为 **therbac-query-proxy** 路由的主机名。

  - 运行以下命令，使用生成的证书创建 **proxy-byo-ca** 和 **proxy-byo-cert secret**：

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

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

1.2.3.2. 单节点 OpenShift 集群的动态指标

动态指标收集根据特定条件支持自动指标收集。默认情况下，SNO 集群不会收集 pod 和容器资源指标。当 SNO 集群达到特定级别的资源消耗后，会动态收集定义的粒度指标。当集群资源消耗在一段时间内持续低于阈值时，细致的指标集合会停止。

这些指标会根据由集合规则指定的受管集群上的条件动态收集。由于这些指标是动态收集的，所以以下 Red Hat Advanced Cluster Management Grafana 仪表板不会显示任何数据。激活集合规则并收集相应指标时，以下面板会在启动集合规则的时间里显示数据：

- Kubernetes/Compute Resources/Namespace (Pods)
- Kubernetes/Compute Resources/Namespace (Workloads)
- Kubernetes/Compute Resources/Nodes (Pods)
- Kubernetes/Compute Resources/Pod
- Kubernetes/Compute Resources/Workload

集合规则包括以下条件：

- 动态收集的一组指标。
- 使用 PromQL 表达式定义的条件。
- 集合的间隔，该集合必须设为 **true**。
- 一个匹配表达式，用于选择需要评估收集规则的集群。

默认情况下，集合规则每 30 秒或以特定时间间隔在受管集群上持续评估。集合间隔和时间间隔中的最低值具有高优先权。当收集规则条件在 **for** 属性指定的持续时间内保留后，收集规则将启动，由规则指定的指标会在受管集群上自动收集。指标集合会在受管集群上不再存在集合规则条件后自动停止，至少有 15 分钟。

集合规则分组为名为 **gather_rules** 的参数部分，该部分可启用或禁用作为组。Red Hat Advanced Cluster Management 安装包含集合规则组 **SNOResourceUsage**，它有两个默认集合规则：**HighCPUUsage** 和 **HighMemoryUsage**。**HighCPUUsage** 集合规则从节点 CPU 使用率超过 70% 时开始。如果 SNO 集群的总内存利用率超过可用节点内存的 70%，则 **HighMemoryUsage** 集合规则开始。目前，前面提到的阈值是固定的，且无法更改。当集合规则开始超过 **for** 属性指定的间隔时，系统会自动开始收集 **dynamic_metrics** 中指定的指标。

在以下 YAML 文件中查看 **collect_rules** 部分的动态指标列表：

```
collect_rules:
  - group: SNOResourceUsage
    annotations:
      description: >
        By default, a SNO cluster does not collect pod and container resource metrics. Once a SNO
cluster
        reaches a level of resource consumption, these granular metrics are collected dynamically.
        When the cluster resource consumption is consistently less than the threshold for a period of
time,
        collection of the granular metrics stops.
    selector:
      matchExpressions:
        - key: clusterType
          operator: In
          values: ["SNO"]
    rules:
    - collect: SNOHighCPUUsage
      annotations:
        description: >
          Collects the dynamic metrics specified if the cluster cpu usage is constantly more than 70% for
2 minutes
      expr: (1 - avg(rate(node_cpu_seconds_total{mode=\"idle\"}[5m]))) * 100 > 70
      for: 2m
```

```
    dynamic_metrics:
      names:
        - container_cpu_cfs_periods_total
        - container_cpu_cfs_throttled_periods_total
        - kube_pod_container_resource_limits
        - kube_pod_container_resource_requests
        - namespace_workload_pod:kube_pod_owner:relabel
        - node_namespace_pod_container:container_cpu_usage_seconds_total:sum_irate
        - node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate
  - collect: SNOHighMemoryUsage
    annotations:
      description: >
        Collects the dynamic metrics specified if the cluster memory usage is constantly more than 70%
for 2 minutes
    expr: (1 - sum(:node_memory_MemAvailable_bytes:sum) /
sum(kube_node_status_allocatable{resource=\"memory\"})) * 100 > 70
    for: 2m
    dynamic_metrics:
      names:
        - kube_pod_container_resource_limits
        - kube_pod_container_resource_requests
        - namespace_workload_pod:kube_pod_owner:relabel
      matches:
        - __name__="container_memory_cache",container!=""
        - __name__="container_memory_rss",container!=""
        - __name__="container_memory_swap",container!=""
        - __name__="container_memory_working_set_bytes",container!=""
```

可以在 **custom-allowlist** 中禁用 **collect_rules.group**，如下例所示。当 **collect_rules.group** 禁用时，指标集合会恢复到之前的行为。这些指标定期收集，指定间隔：

```
collect_rules:
  - group: -SNOResourceUsage
```

只有在启动规则时，数据才会在 Grafana 中显示。

1.2.4. 禁用可观察性

要禁用可观察性服务，请卸载 **observability** 资源。在 OpenShift Container Platform 控制台导航中，选择 **Operators** > **Installed Operators** > **Advanced Cluster Manager for Kubernetes**。删除 **MultiClusterObservability** 自定义资源。

要了解更多有关如何定制可观察性的信息，请参阅定制可观察性。

1.3. 在控制台简介中搜索

对于 Red Hat Advanced Cluster Management for Kubernetes，搜索功能可让您了解所有集群中的 Kubernetes 资源。搜索也对 Kubernetes 资源以及与其他资源的关系进行索引。

- 搜索组件
- 搜索自定义和配置

1.3.1. 搜索组件

搜索构架由以下组件组成：

- **search-collector**：查看 Kubernetes 资源，收集所有受管集群中资源元数据、计算资源关系，并将收集的数据发送到 **search-indexer**。受管集群中的 **search-collector** 作为名为 **klusterlet-addon-search** 的 pod 运行。
- **search-indexer**: 从收集器接收资源元数据并写入 PostgreSQL 数据库。**search-indexer** 还监视 hub 集群中的资源来跟踪活跃受管集群。
- **search-api**：通过 GraphQL 提供 **search-indexer** 中所有集群数据的访问权限，并强制实施基于角色的访问控制 (RBAC)。
- **search-postgres** ：将从 PostgreSQL 数据库实例中的所有受管集群收集数据。

在 hub 集群中默认配置搜索。当您置备或手动导入受管集群时，**klusterlet-addon-search** 会被启用。如果要禁用对受管集群的搜索，请参阅修改集群的 klusterlet 附加设置以了解更多信息。

1.3.2. 搜索自定义和配置

您可以修改 **search-v2-operator** 自定义资源中的默认值。要查看自定义资源的详情，请运行以下命令：

```
oc get search search-v2-operator -o yaml
```

搜索 Operator 会监视 **search-v2-operator** 自定义资源，协调更改并更新活跃的 pod。查看以下配置描述：

- PostgreSQL 数据库存储：
  安装 Red Hat Advanced Cluster Management 时，PostgreSQL 数据库被配置为将 PostgreSQL 数据保存在一个空目录 (**emptyDir**) 卷中。如果空目录大小有限，您可以在持久性卷声明 (PVC) 中保存 PostgreSQL 数据，以提高搜索性能。您可以从 Red Hat Advanced Cluster Management hub 集群中选择一个存储类来备份搜索数据。例如，如果您选择 **gp2** 存储类，您的配置可能类似以下示例：

  ```
  apiVersion: search.open-cluster-management.io/v1alpha1
  kind: Search
  metadata:
    name: search-v2-operator
    namespace: open-cluster-management
    labels:
      cluster.open-cluster-management.io/backup: ""
  spec:
    dbStorage:
      size: 10Gi
      storageClassName: gp2
  ```

  此配置会创建一个名为 **gp2-search** 的 PVC，并挂载到 **search-postgres** pod。默认情况下，存储大小为 **10Gi**。您可以修改存储大小。例如，**20Gi** 可能足以满足大约 200 个受管集群。

- PostgreSQL 数据库配置：
  PostgreSQL 支持数据库性能优化，以优化您的数据库性能。调优配置可以使用 ConfigMap 指定。此 ConfigMap 包含支持的调优参数的 name-value 对。请参阅以下示例命令使用调优参数创建 ConfigMap：

```
oc create configmap tuning-config --from-literal
POSTGRESQL_SHARED_BUFFERS=128MB --from-literal
POSTGRESQL_EFFECTIVE_CACHE_SIZE=128MB --from-literal WORK_MEM=64MB
```

使用前面的 ConfigMap 优化 PostgreSQL 数据库配置。例如，添加 **tuning-config** 作为参数值：

```
apiVersion: search.open-cluster-management.io/v1alpha1
kind: Search
metadata:
  name: search-v2-operator
  namespace: open-cluster-management
  labels:
    cluster.open-cluster-management.io/backup: ""
spec:
  dbConfig: tuning-config
  dbStorage:
  size: 10Gi
  storageClassName: gp2-search
```

- 通过调整 pod 内存或 CPU 要求、副本数和更新任何四个搜索 Pod （indexer, database, queryapi, 或 collector pod）的日志级别来优化成本。更新 **search-v2-operator** 自定义资源的 **deployment** 部分。**search-v2-operator** 管理四个部署，它们可以单独更新。您的 **search-v2-operator** 自定义资源可能类似以下文件：

```
apiVersion: search.open-cluster-management.io/v1alpha1
kind: Search
metadata:
  name: search-v2-operator
  namespace: open-cluster-management
spec:
  dbConfig: tuning-config
  deployments:
    collector:
      resources: (1)
        limits:
          cpu: 500m
          memory: 128Mi
        requests:
          cpu: 250m
          memory: 64Mi
    indexer:
      replicaCount: 3
    database: {}
    queryapi:
      arguments: (2)
      - -v=3
```

(1) 您可以将资源应用到 indexer, database, queryapi, 或 collector pod。

(2) 您可以通过添加 **-v=3** 参数来控制前四个 pod 的日志级别详细程度。

请参阅以下示例，其中内存资源应用到 indexer pod：

```
    indexer:
```

```
    resources:
      limits:
        memory: 5Gi
      requests:
        memory: 1Gi
```

- 搜索 pod 的节点放置：
  您可以使用 **nodeSelector** 参数或 **tolerations** 参数更新搜索 pod 的**放置**。查看以下示例配置：

```
spec:
 dbStorage:
  size: 10Gi
 deployments:
  collector: {}
  database: {}
  indexer: {}
  queryapi: {}
 nodeSelector:
  node-role.kubernetes.io/infra: ""
 tolerations:
 - effect: NoSchedule
  key: node-role.kubernetes.io/infra
  operator: Exists
```

- 有关如何管理搜索的说明，请参阅管理搜索。
- 有关 Red Hat Advanced Cluster Management for Kubernetes 控制台的更多信息，请参阅 Web 控制台。

1.4. 管理搜索

使用 search 从集群中查询资源数据。

需要的访问权限：集群管理员

继续阅读以下主题：

- 创建搜索可配置的集合
- 自定义搜索控制台
- 在控制台中查询
  - 查询 ArgoCD 应用程序
- 更新受管集群上的 klusterlet-addon-search 部署

1.4.1. 创建搜索可配置的集合

通过列出 allow 和 deny list 部分中的资源，创建 **search-collector-config** ConfigMap 以定义从集群中收集哪些 Kubernetes 资源。列出 ConfigMap 中的 **data.AllowedResources** 和 **data.DeniedResources** 部分中的资源。运行以下命令来创建资源：

```
oc apply -f yourconfigMapFile.yaml
```

ConfigMap 可能类似以下 YAML 文件：

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: search-collector-config
 namespace: <namespace where search-collector add-on is deployed>
data:
 AllowedResources: |-
  - apiGroups:
      - "*"
    resources:
      - services
      - pods
  - apiGroups:
      - admission.k8s.io
      - authentication.k8s.io
    resources:
      - "*"
 DeniedResources: |-
  - apiGroups:
      - "*"
    resources:
      - secrets
  - apiGroups:
      - admission.k8s.io
    resources:
      - policies
      - iampolicies
      - certificatepolicies
```

以上 ConfigMap 示例允许从所有 **apiGroups** 收集服务和 **pod**，同时允许从 **admission.k8s.io** 和 **authentication.k8s.io apiGroups** 收集所有资源。同时，ConfigMap 示例还阻止所有 **apiGroups** 的 **secret** 集合，同时防止来自 **apiGroup admission.k8s.io** 的 **policies**, **iampolicies**, and **certificatepolicies** 的集合。

注：如果您不提供 ConfigMap，则默认收集所有资源。如果您只提供 **AllowedResources**，则 **AllowedResources** 中未列出的所有资源都会被自动排除。同时 **AllowedResources** 和 **DeniedResources** 中列出的资源也会被排除。

1.4.2. 自定义搜索控制台

您可以从 OpenShift Container Platform 控制台自定义搜索结果限制。更新 **multicluster-engine** 命名空间中的 **console-mce-config**。这些设置适用于所有用户，并可能会影响性能。查看以下性能参数描述：

- **SAVED_SEARCH_LIMIT** - 每个用户保存的最大搜索量。默认情况下，每个用户可以保存的搜索数量被限制为 10 个。默认值为 **10**。要更新限制，请在 **console-config** ConfigMap 中添加以下键值：**SAVED_SEARCH_LIMIT: x**.
- **SEARCH_RESULT_LIMIT** - 控制台中显示的最大搜索结果量。默认值为 **1000**。要删除此限制，请将设为 **-1**。
- **SEARCH_AUTOCOMPLETE_LIMIT** - 为搜索栏 typeahead 检索的最大建议数。默认值为 **10,000**。要删除此限制，请将设为 **-1**。从 OpenShift Container Platform 控制台运行以下 **patch** 命令，将搜索结果改为 100 个项目：

```
oc patch configmap console-mce-config -n multicluster-engine --type merge -p '{"data":
{"SEARCH_RESULT_LIMIT":"100"}}'
```

1.4.3. 在控制台中查询

您可以在 *搜索框中* 输入任何文字，结果会包括带有这个值的任何属性（如名称或命名空间）。用户无法搜索包含空空格的值。

如需更具体的搜索结果，请在搜索中包含属性。您可以组合属性的相关值以获取更精确的搜索范围。例如，搜索 **cluster:dev red** 以接收与 **dev** 集群中字符串"red" 匹配的结果。

完成以下步骤，使用搜索进行查询：

1. 在导航菜单中点击 **Search**。
2. 在 *搜索框中* 输入要搜索的内容，搜索功能会查找包含该值的资源。
   - 当搜索资源时，会收到与原始搜索结果关联的其他资源。这可帮助您了解这些资源如何与系统中的其他资源进行交互。
   - 搜索返回并列出带有搜索资源的集群。对于 *hub* 集群中的资源，集群名称会显示为 *local-cluster*。
   - 您的搜索结果按 **kind** 分组，每个资源 **kind** 在一个表格中分组。
   - 您的搜索选项依赖于集群对象。
   - 您可以使用特定标签重新定义结果。在查询标签时，搜索是区分大小写的。请参见以下示例，您可以选择过滤：**名称**、**命名空间**、**状态** 和其他资源字段。Auto-complete 提供了重新定义搜索的建议。请参见以下示例：
   - 搜索单个字段，如 **kind:pod** 以查找所有 pod 资源。
   - 搜索多个字段，如 **kind:pod namespace:default** 以在默认命名空间中查找 pod。
     备注：
     - 您还可以使用字符（如 **>, >=, <, <=, !=**）为搜索添加条件。
     - 当使用多个属性值进行搜索时，会返回满足任何一个搜索值的结果。请参见以下示例：
     - 例如，当搜索 **kind:pod name:a** 时，任何名为 **a** 的 pod 都会被返回。
     - 当搜索 **kind:pod name:a,b** 时，任何名为 **a** 或 **b** 的 pod 都会被返回。
     - 搜索 **kind:pod status:!Running** 以查找所有状态不是 **Running** 的 pod 资源。
     - 搜索 **kind:pod restarts:>1** 以查找重启至少两次的所有 pod。
3. 如果要保存搜索，请点击 **Save search** 图标。

1.4.3.1. 查询 ArgoCD 应用程序

当搜索 ArgoCD 应用程序时，您会被定向到 *Applications* 页面。完成以下步骤，从 *Search* 页面访问 ArgoCD 应用程序：

1. 登录到您的 Red Hat Advanced Cluster Management hub 集群。

2. 在控制台标头中选择*搜索*图标。
3. 使用以下值过滤查询：**kind:application** 和 **apigroup:argoproj.io**。
4. 选择要查看的应用程序。*Application* 页面中显示应用的信息的概览。

1.4.4. 更新受管集群上的 klusterlet-addon-search 部署

要从受管集群收集 Kubernetes 对象，**klusterlet-addon-search** pod 在启用了搜索的所有受管集群中运行。此部署在 **open-cluster-management-agent-addon** 命名空间中运行。具有大量资源的受管集群可能需要更多内存才能使 **klusterlet-addon-search** 部署正常工作。

受管集群中的 **klusterlet-addon-search** pod 的资源要求可在 Red Hat Advanced Cluster Management hub 集群的 **ManagedClusterAddon** 自定义资源中指定。每个带有受管集群名称的受管集群都有一个命名空间。从与受管集群名称匹配的命名空间中编辑 **ManagedClusterAddon** 自定义资源。运行以下命令以更新 **xyz** 受管集群中的资源要求：

```
oc edit managedclusteraddon search-collector -n xyz
```

将资源要求作为注解附加。查看以下示例：

```
apiVersion: addon.open-cluster-management.io/v1alpha1
kind: ManagedClusterAddOn
metadata:
  annotations: addon.open-cluster-management.io/search_memory_limit: 2048Mi
  addon.open-cluster-management.io/search_memory_request: 512Mi
```

该注解会覆盖受管集群上的资源要求，并使用新资源要求自动重启 pod。

返回到 Observing 环境简介。

1.5. 定制可观察性

以下部分介绍了对可观察性服务所收集的数据进行自定义、管理和查看的信息。

使用 **must-gather** 命令收集有关为可观察性资源创建的新信息的日志。如需更多信息，请参阅故障排除文档中的 *Must-gather* 部分。

- 创建自定义规则
- 配置 AlertManager
- 转发警报
  - 为受管集群禁用转发警报
- 静默警报
- 阻止警报
- 添加自定义指标
  - 添加用户工作负载指标
  - 删除默认指标

- 将指标导出到外部端点
  - 为外部端点创建 Kubernetes secret
  - 更新 *MultiClusterObservability* 自定义资源
  - 查看指标导出的状态
- 添加*高级*配置
- 从控制台更新 *MultiClusterObservability* 自定义资源副本
- 自定义路由认证
- 自定义用于访问对象存储的证书
- 查看和查找数据
  - 查看 etcd 表
  - 查看 Kubernetes API 服务器仪表板的集群团队服务级别概述
  - 查看 Kubernetes API 服务器仪表板的集群服务级别概述。
- 禁用可观察性

1.5.1. 创建自定义规则

通过在可观察性资源中添加 Prometheus 记录规则和 警报规则，为可观察性安装创建自定义规则。如需更多信息，请参阅 Prometheus 配置。

- 记录规则可让您根据需要预先计算或计算昂贵的表达式。结果保存为一组新的时间序列。
- 通过警报规则，您可以根据如何将警报发送到外部服务来指定警报条件。
  使用 Prometheus 定义自定义规则来创建警报条件，并将通知发送到外部消息服务。

  注： 当您更新自定义规则时，**observability-thanos-rule** pod 会自动重启。

  在 **open-cluster-management-observability** 命名空间中创建一个名为 **thanos-ruler-custom-rules** 的 ConfigMap。键必须被命名为 **custom_rules.yaml**，如下例所示。您可以在配置中创建多个规则。

  - 默认情况下，开箱即用的警报规则在 **open-cluster-management-observability** 命名空间中的 **thanos-ruler-default-rules** ConfigMap 中定义。
    例如，您可以创建一个自定义警报规则，在 CPU 使用量超过了您定义的值时通知您。您的 YAML 可能类似以下内容：

```
data:
  custom_rules.yaml: |
    groups:
      - name: cluster-health
        rules:
        - alert: ClusterCPUHealth-jb
          annotations:
            summary: Notify when CPU utilization on a cluster is greater than the defined
utilization limit
            description: "The cluster has a high CPU usage: {{ $value }} core for {{
```

```
$labels.cluster }} {{ $labels.clusterID }}."
        expr: |
          max(cluster:cpu_usage_cores:sum) by (clusterID, cluster, prometheus) > 0
        for: 5s
        labels:
          cluster: "{{ $labels.cluster }}"
          prometheus: "{{ $labels.prometheus }}"
          severity: critical
```

- 您还可以在 **thanos-ruler-custom-rules** ConfigMap 中创建自定义记录规则。例如，您可以创建一个记录规则，让您可以获取 pod 的容器内存缓存的总和。您的 YAML 可能类似以下内容：

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

注：如果这是第一个新的自定义规则，它会立即创建。对于 ConfigMap 的更改，会自动重新加载配置。由于 **observability-thanos-ruler** sidecar 中的 **config-reload**，所以会重新载入配置。

要验证警报规则是否正常工作，启动 Grafana 仪表板，进入 **Explore** 页面并查询 **ALERTS**。只有启动警报时，Grafana 才会在 Grafana 中提供警报。

1.5.2. 配置 AlertManager

集成外部消息工具，如 email、Slack 和 PagerDuty 以接收来自 AlertManager 的通知。您必须覆盖 **open-cluster-management-observability** 命名空间中的 **alertmanager-config** secret 来添加集成，并为 AlertManager 配置路由。完成以下步骤以更新自定义接收器规则：

1. 从 **alertmanager-config** secret 中提取数据。运行以下命令:

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

2. 运行以下命令，编辑并保存 **alertmanager.yaml** 文件配置：

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

更新的 secret 可能与以下类似：

```
global
  smtp_smarthost: 'localhost:25'
  smtp_from: 'alertmanager@example.org'
  smtp_auth_username: 'alertmanager'
  smtp_auth_password: 'password'
templates:
- '/etc/alertmanager/template/*.tmpl'
```

```
route:
  group_by: ['alertname', 'cluster', 'service']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 3h
  receiver: team-X-mails
  routes:
  - match_re:
      service: ^(foo1|foo2|baz)$
    receiver: team-X-mails
```

您的更改会在修改后立即生效。如需 AlertManager 的示例，请参阅 prometheus/alertmanager。

1.5.3. 转发警报

启用可观察性后，来自 OpenShift Container Platform 受管集群的警报会自动发送到 hub 集群。您可以使用 **alertmanager-config** YAML 文件，为警报配置外部通知系统。

查看 **alertmanager-config** YAML 文件示例：

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

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

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

如果要配置代理进行警报转发，请在 **alertmanager-config** YAML 文件中添加以下 **global** 条目：

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

1.5.3.1. 为受管集群禁用转发警报

禁用受管集群的警报转发。在 **MultiClusterObservability** 自定义资源中添加以下注解：

```
metadata:
    annotations:
      mco-disable-alerting: "true"
```

设置注解时，受管集群上的警报转发配置会被恢复。对 **openshift-monitoring** 命名空间中的 **ocp-monitoring-config** ConfigMap 所做的任何更改都会被恢复。设置注解可确保 **ocp-monitoring-config** ConfigMap 不再由 observability operator 端点管理或更新。更新配置后，受管集群中的 Prometheus 实例会重启。

重要： 如果您有一个带有指标数据的 Prometheus 实例，且 Prometheus 实例重启了 Prometheus 实例，则受管集群上的指标将会丢失。但是，hub 集群中的指标不会受到影响。

恢复更改后，会在 **open-cluster-management-addon-observability** 命名空间中创建一个名为 **cluster-monitoring-reverted** 的 ConfigMap。任何新的、手动添加的警报转发配置都不会从 ConfigMap 恢复。

验证 hub 集群警报管理器不再将受管集群警报传播到第三方消息传递工具。请参阅上一节中 *配置 AlertManager*。

1.5.4. 静默警报

添加您不想接收的警报。您可以根据警报名称、匹配标签或持续时间来静默警报。添加要静默的警报后，会创建一个 ID。您的静默警报的 ID 可能类似以下字符串 **d839aca9-ed46-40be-84c4-dca8773671da**。

继续读取静默警报的方法：

- 要静默 Red Hat Advanced Cluster Management 警报，您必须有权访问 **open-cluster-management-observability** 命名空间中的 **alertmanager-main** pod。例如，在 pod 终端中输入以下命令来静默 **SampleAlert** ：

  ```
  amtool silence add --alertmanager.url="http://localhost:9093" --author="user" --comment="Silencing sample alert" alertname="SampleAlert"
  ```

- 通过使用多个匹配标签静默警报。以下命令使用 **match-label-1** 和 **match-label-2** ：

  ```
  amtool silence add --alertmanager.url="http://localhost:9093" --author="user" --comment="Silencing sample alert" <match-label-1>=<match-value-1> <match-label-2>=<match-value-2>
  ```

- 如果要在特定时间段内静默警报，请使用 **--duration** 标志。运行以下命令，以静默一个小时的 **SampleAlert** ：

  ```
  amtool silence add --alertmanager.url="http://localhost:9093" --author="user" --comment="Silencing sample alert" --duration="1h" alertname="SampleAlert"
  ```

  您还可以为静默的警报指定开始或结束时间。输入以下命令在特定开始时静默 **SampleAlert** ：

  ```
  amtool silence add --alertmanager.url="http://localhost:9093" --author="user" --comment="Silencing sample alert" --start="2023-04-14T15:04:05-07:00" alertname="SampleAlert"
  ```

- 要查看创建的所有静默警报，请运行以下命令：

  ```
  amtool silence --alertmanager.url="http://localhost:9093"
  ```

- 如果您不再需要静默警报，请运行以下命令终止警报：

  ```
  amtool silence expire --alertmanager.url="http://localhost:9093" "d839aca9-ed46-40be-84c4-dca8773671da"
  ```

- 要结束所有警报的静默，请运行以下命令：

  ```
  amtool silence expire --alertmanager.url="http://localhost:9093" $(amtool silence query --alertmanager.url="http://localhost:9093" -q)
  ```

-

1.5.5. 阻止警报

在全局范围内限制 Red Hat Advanced Cluster Management 警报，这不太严重。通过在 **open-cluster-management-observability** 命名空间中定义 **alertmanager-config** 中的禁止规则来限制警报。

当一组与另一组现有匹配者匹配的一组参数时，禁止规则会静默警报。为了让规则生效，目标和源警报必须具有 **equal** 列表中标签名称相同的标签值。您的 **inhibit_rules** 可能类似以下：

```
global:
  resolve_timeout: 1h
inhibit_rules:(1)
  - equal:
      - namespace
    source_match:(2)
      severity: critical
    target_match_re:
      severity: warning|info
```

1. 定义了 **inhibit_rules** 参数部分，以查找同一命名空间中的警报。当一个命名空间中出现了一个 **critical** 警报时，如果在那个命名空间中还有其他包括严重性级别为 **warning** 或 **info** 的警报时，只有 **critical** 警报会路由到 Alertmanager 接收器。匹配时可能会显示以下警报：

```
ALERTS{alertname="foo", namespace="ns-1", severity="critical"}
ALERTS{alertname="foo", namespace="ns-1", severity="warning"}
```

2. 如果 **source_match** 和 **target_match_re** 参数的值不匹配，则警报将路由到接收器：

```
ALERTS{alertname="foo", namespace="ns-1", severity="critical"}
ALERTS{alertname="foo", namespace="ns-2", severity="warning"}
```

- 要查看 Red Hat Advanced Cluster Management 中的禁止的警报，请输入以下命令：

```
amtool alert --alertmanager.url="http://localhost:9093" --inhibited
```

1.5.6. 添加自定义指标

将指标添加到 **metrics_list.yaml** 文件中，用来从受管集群中收集数据。

在添加自定义指标前，请确定使用以下命令启用了 **mco observability** ： **oc get mco observability -o yaml**。在 **status.conditions.message** 中检查以下消息：**Observability components are deployed and running**

创建名为 **observability-metrics-custom-allowlist.yaml** 的文件，并将自定义指标名称添加到 **metrics_list.yaml** 参数。ConfigMap 的 YAML 可能类似以下内容：

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: observability-metrics-custom-allowlist
data:
  metrics_list.yaml: |
```

```
    names:
      - node_memory_MemTotal_bytes
    rules:
    - record: apiserver_request_duration_seconds:histogram_quantile_90
      expr:
histogram_quantile(0.90,sum(rate(apiserver_request_duration_seconds_bucket{job=\"apiserver\",
      verb!=\"WATCH\"}[5m])) by (verb,le))
```

对于用户工作负载指标，请参阅*添加用户工作负载指标*部分。

- 在 **names** 部分中，添加要从受管集群收集的自定义指标的名称。
- 在 **rules** 部分中，仅为 **expr** 和 **record** 参数对输入一个值来定义查询表达式。指标作为来自受管集群的 **record** 参数中定义的名称来收集。返回的指标值是运行查询表达式后的结果。
- **name** 和 **rules** 部分是可选的。您可以使用其中一个或两个部分。

使用以下命令，在 **open-cluster-management-observability** 命名空间中创建 **observability-metrics-custom-allowlist** ConfigMap：**oc apply -n open-cluster-management-observability -f observability-metrics-custom-allowlist.yaml**。

通过 Grafana 仪表板，查询来自 **Explore** 页的指标数据来验证是否收集了您的自定义指标的数据。您也可以在您自己的仪表板中使用自定义指标。有关查看仪表板的更多信息，请参阅指定 Grafana 仪表板。

1.5.6.1. 添加用户工作负载指标

您可以从 OpenShift Container Platform 中的工作负载收集 OpenShift Container Platform 用户定义的指标。您必须启用监控，请参阅为用户定义的项目启用监控。

如果您有一个为用户定义的工作负载启用监控的受管集群，用户工作负载位于 **test** 命名空间中，并生成指标。Prometheus 从 OpenShift Container Platform 用户工作负载收集这些指标。

通过在 **test** 命名空间中创建一个名为 **observability-metrics-custom-allowlist** 的 ConfigMap，从用户工作负载收集指标。查看以下示例：

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: observability-metrics-custom-allowlist
  namespace: test
data:
  uwl_metrics_list.yaml: |
    names:
      - sample_metrics
```

- **uwl_metrics_list.yaml** 是 ConfigMap 数据的密钥。
- ConfigMap 数据的值采用 YAML 格式。**name** 部分包含您要从 **test** 命名空间收集的指标名称列表。创建 ConfigMap 后，由可观察收集器收集目标命名空间中的指定指标并推送到 hub 集群。

1.5.6.2. 删除默认指标

如果您不需要收集管理集群中的特定指标数据，从 **observability-metrics-custom-allowlist.yaml** 文件中删除相应的指标。当您删除指标时，不会在受管集群上收集指标数据。如前文所述，首先验证 **mco observability** 是否已启用。

您可以在 metrics 名称的开头使用连字符 **-** 将默认指标名称添加到 **metrics_list.yaml** 参数中。例如： **-cluster_infrastructure_provider**。

使用以下命令，在 **open-cluster-management-observability** 命名空间中创建 **observability-metrics-custom-allowlist** ConfigMap：**oc apply -n open-cluster-management-observability -f observability-metrics-custom-allowlist.yaml**。

验证特定指标是否没有从受管集群中收集。当您从 Grafana 仪表板查询指标时，指标不会显示。

1.5.7. 将指标导出到外部端点

您可以自定义可观察性，将指标导出到外部端点，该端点实时支持 Prometheus Remote-Write 规格。如需更多信息，请参阅 Prometheus Remote-Write 规格。

1.5.7.1. 为外部端点创建 Kubernetes secret

您必须使用 **open-cluster-management-observability** 命名空间中外部端点的访问信息创建一个 Kubernetes secret。查看以下示例 secret：

```
apiVersion: v1
kind: Secret
metadata:
  name: victoriametrics
  namespace: open-cluster-management-observability
type: Opaque
stringData:
  ep.yaml: |
    url: http://victoriametrics:8428/api/v1/write
    http_client_config:
      basic_auth:
        username: test
        password: test
```

ep.yaml 是内容的密钥，在下一步中 **MultiClusterObservability** 自定义资源中使用。目前，可观察性支持在不进行安全检查的情况下将指标导出到端点，以及基本身份验证或 **tls** 启用。查看下表以了解支持的参数的完整列表：

Name	描述	模式
url *必需*	外部端点的 URL。	字符串
http_client_config *optional*	HTTP 客户端的高级配置。	HttpClientConfig

HttpClientConfig

Name	描述	模式
basic_auth *可选*	用于基本身份验证的 HTTP 客户端配置。	BasicAuth
tls_config *optional*	TLS 的 HTTP 客户端配置。	TLSConfig

BasicAuth

Name	描述	模式
username *可选*	基本授权的用户名。	字符串
password *可选*	基本授权的密码。	字符串

TLSConfig

Name	描述	模式
secret_name *必需*	包含证书的 secret 的名称。	字符串
ca_file_key *optional*	secret 中的 CA 证书的密钥（只在 **insecure_skip_verify** 设为 true 时才可选）。	字符串
cert_file_key *required*	secret 中客户端证书的密钥。	字符串
key_file_key *required*	机密中的客户端密钥的键。	字符串
insecure_skip_verify *optional*	用于跳过目标证书的验证参数。	bool

1.5.7.2. 更新 *MultiClusterObservability* 自定义资源

创建 Kubernetes secret 后，您必须更新 **MultiClusterObservability** 自定义资源，以便在 **spec.storageConfig** 参数中添加 **writeStorage**。查看以下示例：

```
spec:
  storageConfig:
    writeStorage:
    - key: ep.yaml
      name: victoriametrics
```

writeStorage 的值是一个列表。当您要将指标导出到一个外部端点时，您可以在列表中添加一个项目。如果您在列表中添加多个项，则指标将导出到多个外部端点。每个项目包含两个属性：*name* 和 *key*。*name* 是包含端点访问信息的 Kubernetes secret 的名称， *key* 是 secret 中内容的密钥。查看以下描述表

1.5.7.3. 查看指标导出的状态

启用指标导出后，您可以通过检查 **acm_remote_write_requests_total** 指标来查看指标导出的状态。在 hub 集群的 OpenShift 控制台中点 *Observe* 部分中的 *Metrics* 进入 **Metrics** 页面。

然后查询 **acm_remote_write_requests_total** 指标。该指标的值是在一个 observatorium API 实例上具有特定响应的请求总数。**name** 标签是外部端点的名称。**code** 标签是指标导出的 HTTP 请求的返回代码。

1.5.8. 添加 *高级*配置

添加 **advanced** 配置部分，以根据您的需要更新每个可观察性组件的保留。

编辑 **MultiClusterObservability** 自定义资源，并使用以下命令添加 **advanced** 部分： **oc edit mco observability -o yaml**。您的 YAML 文件可能类似以下内容：

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

有关可添加到**高级**配置中的所有参数的描述，请参阅 Observability API。

1.5.9. 从控制台更新 *MultiClusterObservability* 自定义资源副本

如果工作负载增加，增加可观察 pod 的副本数。从您的 hub 集群中进入到 Red Hat OpenShift Container Platform 控制台。找到 **MultiClusterObservability** 自定义资源，并更新您要更改副本的组件的 **replicas** 参数值。更新的 YAML 可能类似以下内容：

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

有关 **mco observability** 自定义资源中的参数的更多信息，请参阅 Observability API。

1.5.10. 自定义路由认证

如果要自定义 OpenShift Container Platform 路由认证，则必须在 **alt_names** 部分添加路由。要确保 OpenShift Container Platform 路由可以访问，请添加以下信息：**alertmanager.apps.<domainname>**, **observatorium-api.apps.<domainname>**, **rbac-query-proxy.apps.<domainname>**。

注意 ：用户负责证书轮转和更新。

1.5.11. 自定义用于访问对象存储的证书

完成以下步骤以自定义用于访问对象存储的证书：

1. 通过在对象存储 secret 中添加证书来编辑 **http_config** 部分。查看以下示例：

```
thanos.yaml: |
  type: s3
  config:
    bucket: "thanos"
    endpoint: "minio:9000"
    insecure: false
    access_key: "minio"
    secret_key: "minio123"
    http_config:
      tls_config:
        ca_file: /etc/minio/certs/ca.crt
        insecure_skip_verify: false
```

2. 在 **open-cluster-management-observability** 命名空间中添加对象存储 secret。secret 必须包含您在前面的 secret 示例中定义的 **ca.crt**。如果要启用 Mutual TLS，则需要在前面的 secret 中提供 **public.crt** 和 **private.key**。查看以下示例：

```
thanos.yaml: |
  type: s3
  config:
    ...
    http_config:
      tls_config:
        ca_file: /etc/minio/certs/ca.crt (1)
        cert_file: /etc/minio/certs/public.crt
        key_file: /etc/minio/certs/private.key
        insecure_skip_verify: false
```

(1) **thanos-object-storage** secret 的证书和密钥值的路径。

3. 通过更新 **MultiClusterObservability** 自定义资源中的 **TLSSecretName** 参数来配置 secret 名称。查看以下示例，其中 secret 名称为 **tls-certs-secret** ：

```
metricObjectStorage:
  key: thanos.yaml
  name: thanos-object-storage
  tlsSecretName: tls-certs-secret
```

此 secret 可以挂载到需要访问对象存储的所有组件的 **tlsSecretMountPath** 资源中，它包括以下组件：**receiver**,**store**,**ruler**,**compact**。

1.5.12. 查看和查找数据

通过从 hub 集群访问 Grafana 来查看来自受管集群的数据。您可以查询特定的警报并为查询添加过滤器。

例如，对于来自单一节点集群中的 *cluster_infrastructure_provider*，使用以下查询表达式：**cluster_infrastructure_provider{clusterType="SNO"}**

备注：

- 如果在单一节点受管集群中启用了可观察性，则不要设置 **ObservabilitySpec.resources.CPU.limits** 参数。当您设置 CPU 限制时，observability pod 会计算为受管集群的容量。如需更多信息，请参阅管理工作负载分区。

1.5.12.1. 查看历史数据

当您查询历史数据时，手动设置查询参数选项来控制仪表板中显示的数据量。完成以下步骤：

1. 在 hub 集群中，选择位于控制台标头中的 **Grafana 链接**。
2. 选择 **Edit Panel** 来编辑集群仪表板。
3. 在 Grafana 中的 Query 前端数据源中点 *Query* 选项卡。
4. 选择 **$datasource**。
5. 如果要查看更多数据，请增加 *Step* 参数的值。如果 *Step* 参数部分为空，则会自动计算。
6. 找到 *Custom query parameters* 字段，然后选择 **max_source_resolution=auto**。
7. 要验证是否显示数据，请刷新 Grafana 页面。

您的查询数据会出现在 Grafana 仪表板中。

1.5.12.2. 查看 etcd 表

在 Grafana 中查看 hub 集群仪表板中的 etcd 表，以了解 etcd 作为数据存储的稳定性。

从 hub 集群中选择 Grafana 链接来查看从 hub 集群收集的 *etcd* 表数据。此时会显示跨受管集群的 *领导选举更改*。

1.5.12.3. 查看 Kubernetes API 服务器仪表板的集群团队服务级别概述

在 Grafana 中的 hub 集群仪表板中查看集群 fleet Kubernetes API 服务级别概述。

进入 Grafana 仪表板后，选择 **Kubernetes** > **Service-Level Overview** > **API Server** 来访问受管仪表板菜单。此时会显示 *Fleet Overview* 和 *Top Cluster* 的详情。

查看过去 7 或 30 天、终止和非关闭集群以及 API 服务器请求持续时间，超过或满足目标 *service-level objective* (SLO)值的集群总数。

1.5.12.4. 查看 Kubernetes API 服务器仪表板的集群服务级别概述。

在 Grafana 中的 hub 集群仪表板中查看 Kubernetes API 服务级别概述表。

进入 Grafana 仪表板后，选择 **Kubernetes** > **Service-Level Overview** > **API Server** 来访问受管仪表板菜单。此时会显示 *Fleet Overview* 和 *Top Cluster* 的详情。

查看过去七或 30 天、剩余停机时间和趋势的错误预算。

1.5.13. 禁用可观察性

您可以禁用可观察性，在 Red Hat Advanced Cluster Management hub 集群中停止数据收集。

1.5.13.1. 在所有集群中禁用可观察性

通过删除所有受管集群中的可观察性组件来禁用可观察性。

通过将 **enableMetrics** 设置为 **false** 来更新 **multicluster-observability-operator** 资源。更新的资源可能类似如下：

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

1.5.13.2. 在单个集群中禁用可观察性

通过删除特定受管集群中的可观察性组件来禁用可观察性。将 **observability: disabled** 标签添加到 **managedclusters.cluster.open-cluster-management.io** 自定义资源中。

在 Red Hat Advanced Cluster Management 控制台 *Clusters* 页面中，将 **observability=disabled** 标签添加到指定的集群中。

备注：当一个带有可观察性组件的受管集群被分离时，**metric-collector** 部署会被删除。

要了解更多有关警报转发的信息，请参阅 Prometheus AlertManager 文档。有关使用可观察性服务监控控制台数据的更多信息，请参阅 观察环境简介。

1.6. 设计您的 GRAFANA 仪表板

您可以通过创建一个 **grafana-dev** 实例来设计 Grafana 仪表板。

- 设置 Grafana 开发人员实例
- 设计您的 Grafana 仪表板
  - 使用 ConfigMap 设计 Grafana 仪表板
- 在 Grafana 中使用受管集群标签
  - 添加受管集群标签
  - 启用受管集群标签
  - 禁用受管集群标签
- 卸载 Grafana 开发者实例

1.6.1. 设置 Grafana 开发人员实例

首先，克隆 **stolostron/multicluster-observability-operator/** 存储库，以便您可以运行 **tools** 文件夹中的脚本。确保使用最当前的 **grafana-dev** 实例。

完成以下步骤以设置 Grafana 开发人员实例：

1. 运行 **setup-grafana-dev.sh** 来设置 Grafana 实例。运行脚本时，会创建以下资源：**secret/grafana-dev-config**、**deployment.apps/grafana-dev**、**service/grafana-dev**、**ingress.extensions/grafana-dev**、**persistentvolumeclaim/grafana-dev**：

```
./setup-grafana-dev.sh --deploy
secret/grafana-dev-config created
deployment.apps/grafana-dev created
service/grafana-dev created
serviceaccount/grafana-dev created
clusterrolebinding.rbac.authorization.k8s.io/open-cluster-management:grafana-crb-dev
created
route.route.openshift.io/grafana-dev created
persistentvolumeclaim/grafana-dev created
oauthclient.oauth.openshift.io/grafana-proxy-client-dev created
deployment.apps/grafana-dev patched
service/grafana-dev patched
route.route.openshift.io/grafana-dev patched
oauthclient.oauth.openshift.io/grafana-proxy-client-dev patched
clusterrolebinding.rbac.authorization.k8s.io/open-cluster-management:grafana-crb-dev
patched
```

2. 使用 **switch-to-grafana-admin.sh** 脚本将用户角色切换到 Grafana 管理员。

   a. 选择 Grafana URL, **https://grafana-dev-open-cluster-management-observability.{OPENSHIFT_INGRESS_DOMAIN}** 并登录。

   b. 然后，运行以下命令来添加切换的用户作为 Grafana 管理员。例如，在使用 **kubeadmin** 登录后，运行以下命令：

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

Grafana 开发人员实例已设置。

1.6.2. 设计您的 Grafana 仪表板

设置 Grafana 实例后，您可以设计仪表板。完成以下步骤以刷新 Grafana 控制台并设计您的仪表板：

1. 在 Grafana 控制台中，通过在导航面板中选择 **Create** 图标来创建仪表板。选择 **Dashboard**，然后单击 **Add new panel**。
2. 在 *New Dashboard/Edit Panel* 视图中导航到 *Query* 选项卡。
3. 从数据源选择器中选择 **Observatorium** 并输入 PromQL 查询来配置查询。
4. 在 Grafana 仪表板标头中点击仪表板标头中的 **Save** 图标。
5. 添加一个描述性名称并点 **Save**。

1.6.2.1. 使用 ConfigMap 设计 Grafana 仪表板

使用 ConfigMap 设计 Grafana 仪表板。您可以使用 **generate-dashboard-configmap-yaml.sh** 脚本在本地生成仪表板 ConfigMap，并在本地保存 ConfigMap:

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

如果您没有运行上述脚本的权限，请完成以下步骤：

1. 选择一个仪表板并点 **Dashboard settings** 图标。
2. 在导航框中，点 **JSON Model** 图标。
3. 复制仪表板 JSON 数据，并将它粘贴到 ***data*** 部分。
4. 修改 ***name*** 并替换 ***$your-dashboard-name***。在 **data.$your-dashboard-name.json.$$your_dashboard_json** 的 **uid** 项中输入一个 UUID（universally unique identifier）。您可以使用 uuidegen 等程序来创建 UUID。ConfigMap 可能类似以下文件：

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

备注：

- 如果在 **grafana-dev** 实例中创建了仪表板，您可以使用仪表板的名称，并在脚本中将其作为参数传递。例如，在 **grafana-dev** 实例中创建一个名为 *Demo Dashboard* 的仪表板。通过 CLI，您可以运行以下脚本：

  ```
  ./generate-dashboard-configmap-yaml.sh "Demo Dashboard"
  ```

  运行脚本后，您可能会收到以下信息：

  ```
  Save dashboard <demo-dashboard> to ./demo-dashboard.yaml
  ```

- 如果您的仪表板不在 *General* 文件夹中，您可以在此 ConfigMap 的 **annotations** 部分中指定文件夹名称：

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

  完成 ConfigMap 的更新后，您可以安装它，将仪表板导入到 Grafana 实例。

通过 CLI 或 OpenShift Container Platform 控制台应用 YAML 文件来验证是否已创建 YAML 文件。创建 **open-cluster-management-observability** 命名空间中的 ConfigMap。通过 CLI 运行以下命令：

```
oc apply -f demo-dashboard.yaml
```

-

在 OpenShift Container Platform 控制台中，使用 **demo-dashboard.yaml** 文件创建 ConfigMap。控制面板位于 *Custom* 文件夹中。

1.6.3. 在 Grafana 中使用受管集群标签

当在 hub 集群中启用可观察性时，**observability-managed-cluster-label-allowlist** ConfigMap 会在 **open-cluster-management-observability** 命名空间中创建。ConfigMap 包含由 **observabilty-rbac-query-proxy** pod 维护的受管集群标签列表，用于在 *ACM - Cluster Overview* Grafana 仪表板中填充要过滤的标签名称列表。默认情况下，可观察性会忽略 **observability-managed-cluster-label-allowlist** ConfigMap 中的标签子集。

当集群导入到受管集群团队或修改时，**observability-rbac-query-proxy** pod 会监视对受管集群标签的任何更改，并自动更新 **observability-managed-cluster-label-allowlist** ConfigMap 以反映更改。ConfigMap 仅包含唯一的标签名称，这些名称包含在 **ignore_labels** 或 **labels** 列表中。您的 **observability-managed-cluster-label-allowlist** ConfigMap 可能类似以下 YAML 文件：

```
data:
  managed_cluster.yaml: |
    ignore_labels:
      - clusterID
      - cluster.open-cluster-management.io/clusterset
      - feature.open-cluster-management.io/addon-application-manager
      - feature.open-cluster-management.io/addon-cert-policy-controller
      - feature.open-cluster-management.io/addon-cluster-proxy
      - feature.open-cluster-management.io/addon-config-policy-controller
      - feature.open-cluster-management.io/addon-governance-policy-framework
      - feature.open-cluster-management.io/addon-iam-policy-controller
      - feature.open-cluster-management.io/addon-observability-controller
      - feature.open-cluster-management.io/addon-search-collector
      - feature.open-cluster-management.io/addon-work-manager
      - installer.name
      - installer.namespace
      - local-cluster
      - name
    labels:
      - cloud
      - vendor
```

启用的标签会在 *ACM - Clusters Overview* Grafana 仪表板上的下拉过滤器中显示。值来自 **acm_managed_cluster_labels** 指标，具体取决于所选 **label** 键值。

ConfigMap 的 **ignore_labels** 键列表中列出的任何标签都会从 *ACM - Clusters Overview* Grafana 仪表板上的下拉过滤器中删除。

1.6.3.1. 添加受管集群标签

当您向 **observability-managed-cluster-label-allowlist** ConfigMap 中添加受管集群标签时，标签将作为 Grafana 中的过滤器选项提供。为 hub 集群或与受管集群团队关联的受管集群对象添加唯一标签。例如，如果您将标签 **department=finance** 添加到受管集群，则 ConfigMap 会被更新，并可能类似以下更改：

```
data:
  managed_cluster.yaml: |
    ignore_labels:
      - clusterID
```

```
    - cluster.open-cluster-management.io/clusterset
    - feature.open-cluster-management.io/addon-application-manager
    - feature.open-cluster-management.io/addon-cert-policy-controller
    - feature.open-cluster-management.io/addon-cluster-proxy
    - feature.open-cluster-management.io/addon-config-policy-controller
    - feature.open-cluster-management.io/addon-governance-policy-framework
    - feature.open-cluster-management.io/addon-iam-policy-controller
    - feature.open-cluster-management.io/addon-observability-controller
    - feature.open-cluster-management.io/addon-search-collector
    - feature.open-cluster-management.io/addon-work-manager
    - installer.name
    - installer.namespace
    - local-cluster
    - name
  labels:
    - cloud
    - department
    - vendor
```

1.6.3.2. 启用受管集群标签

通过从 **observability-managed-cluster-label-allowlist** ConfigMap 中的 **ignore_labels** 列表中删除标签来启用已禁用的受管集群标签。

例如，启用 **local-cluster** 和 **name** 标签。您的 **observability-managed-cluster-label-allowlist** ConfigMap 可能类似以下内容：

```
data:
  managed_cluster.yaml: |
    ignore_labels:
      - clusterID
      - installer.name
      - installer.namespace
    labels:
      - cloud
      - vendor
      - local-cluster
      - name
```

ConfigMap 在 30 秒后重新同步，以确保更新了集群标签。更新 ConfigMap 后，检查 **open-cluster-management-observability** 命名空间中的 **observability-rbac-query-proxy** pod 日志，以验证列出标签的位置。pod 日志中可能会显示以下信息：

```
enabled managedcluster labels: <label>
```

在 Grafana 仪表板中，验证标签是否在 *Label* 下拉菜单中选择。

1.6.3.3. 禁用受管集群标签

从 *Label* 下拉菜单过滤器中列出的受管集群标签排除受管集群标签。将标签名称添加到 **ignore_labels** 列表中。例如，如果您将 **local-cluster** 和 **name** 重新添加到 **ignore_labels** 列表中，您的 YAML 可能类似以下文件：

```
data:
```

```
  managed_cluster.yaml: |
    ignore_labels:
      - clusterID
      - installer.name
      - installer.namespace
      - local-cluster
      - name
    labels:
      - cloud
      - vendor
```

检查 **open-cluster-management-observability** 命名空间中的 **observability-rbac-query-proxy** pod 日志，以验证列出标签的位置。pod 日志中可能会显示以下信息：

```
disabled managedcluster label: <label>
```

1.6.4. 卸载 Grafana 开发者实例

卸载实例时，相关资源也会被删除。运行以下命令:

```
./setup-grafana-dev.sh --clean
secret "grafana-dev-config" deleted
deployment.apps "grafana-dev" deleted
serviceaccount "grafana-dev" deleted
route.route.openshift.io "grafana-dev" deleted
persistentvolumeclaim "grafana-dev" deleted
oauthclient.oauth.openshift.io "grafana-proxy-client-dev" deleted
clusterrolebinding.rbac.authorization.k8s.io "open-cluster-management:grafana-crb-dev" deleted
```

返回到 Observing 环境简介。

1.7. 在 RED HAT INSIGHTS 中使用可观察性

Red Hat Insights 与 Red Hat Advanced Cluster Management observability 集成，并被启用来帮助识别集群中的现有或潜在问题。Red Hat Insights 可帮助您识别、确定和解决稳定性、性能、网络和安全风险。Red Hat OpenShift Container Platform 通过 OpenShift Cluster Manager 提供集群健康监控。OpenShift Cluster Manager 会以匿名的形式收集有关集群健康、使用和大小的聚合信息。如需更多信息，请参阅 Red Hat Insights 产品文档。

当您创建或导入 OpenShift 集群时，受管集群中的匿名数据会自动发送到红帽。这些信息用于创建提供集群健康信息的智能分析工具。Red Hat Advanced Cluster Management 管理员可以使用这个健康信息根据严重性创建警报。

需要的访问权限： 集群管理员

1.7.1. 先决条件

- 确保启用了 Red Hat Insights。如需更多信息，请参阅修改全局集群 pull secret 以禁用远程健康报告。
- 安装 OpenShift Container Platform 版本 4.0 或更高版本。
- hub 集群用户注册到 OpenShift Cluster Manager，必须能够管理 OpenShift Cluster Manager 中的所有 Red Hat Advanced Cluster Management 受管集群。

1.7.2. 来自 Red Hat Advanced Cluster Management 控制台的 Red Hat Insights

继续阅读以查看集成的功能描述：

- 当您从 *Clusters* 页面选择集群时，您可以从 *Status* 卡中选择**识别出的问题的数量**。*Status* 卡显示有关 *节点*、*应用程序*、*策略违规*和 *识别出的问题*的信息。*Identified issues* 卡包括了 Red Hat insights 中的信息。*Identified issues* 状态按严重性显示问题的数量。问题严重性的分类级别如下：*Critical*、*Major*、*Low* 和 *Warning*

- 在点数量后会显示 *Potential issue* 侧面板。面板中显示了所有问题的摘要和图表信息。您还可以使用搜索功能搜索推荐的补救方法。补救选项显示漏洞的 *Description*、与漏洞关联的 *Category*，以及 *Total risk*。

- 在 *Description* 部分中，您可以选择到这个漏洞的链接。通过选择 *How to remediate* 选项卡来查看相关的步骤来解决您的漏洞。您还可以单击 *Reason* 选项卡来查看漏洞发生的原因。

如需更多信息，请参阅管理 Insights **PolicyReports**。

1.8. 管理 INSIGHT POLICYREPORTS

Red Hat Advanced Cluster Management for Kubernetes **PolicyReports** 是 **insights-client** 生成的违反情况。**PolicyReports** 用于定义和配置发送到事件管理系统的警报。出现违反情况时，来自 **PolicyReport** 的警报将发送到事件管理系统。

查看以下部分以了解如何管理和查看 Insights **PolicyReports** ：

- 搜索 insight 策略报告
- 从控制台查看发现的问题

1.8.1. 搜索 insight 策略报告

您可以在受管集群中搜索具有冲突的特定 Insights **PolicyReport**。

登录 Red Hat Advanced Cluster Management hub 集群后，点控制台标头中的 *Search* 图标进入 *Search* 页面。输入以下查询：**kind:policyreport**。

注： **PolicyReport** 名称与集群名称匹配。

您还可以通过洞察策略违反和类别进一步指定查询。当您选择一个 **PolicyReport** 名称时，您会被重定向到关联的集群的 *Details* 页面。*Insights* 侧栏会自动显示。

如果禁用了搜索服务，且您要搜索洞察信息，请在 hub 集群中运行以下命令：

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

1.8.2. 从控制台查看发现的问题

您可以查看特定集群中确定的问题。

登录 Red Hat Advanced Cluster Management 集群后，从导航菜单中选择 **Overview**。选择一个严重性来查看与该严重性关联的 **PolicyReports**。您可以在 *Cluster issues* 概况卡中查看集群问题的详细信息和严重程度。

另外，您还可以从导航菜单中选择 **Clusters**。从表中选择一个受管集群来查看更多详细信息。在 *Status* 卡中查看确定的问题数量。

选择潜在问题的数量来查看严重性图，并推荐对问题进行补救。点漏洞的链接来查看与漏洞相关的 *How to remediate* 和 *Reason* 信息。

注 ： 在解决了这个问题后，Red Hat Advanced Cluster Management 每 30 分钟会接收 Red Hat Insights，Red Hat Insights 每两小时更新一次。

务必验证从 **PolicyReport** 发送警报消息的组件。进入到 *Governance* 页面，再选择特定的**策略报告**。选择 *Status* 选项卡，再单击 **View details** 链接来查看 **PolicyReport** YAML 文件。

找到 **source** 参数，该参数会通知您发送违反情况的组件。值选项为 **grc** 和 **insights**。

了解如何为 **PolicyReports** 创建自定义警报规则，如需更多信息，请参阅 配置 AlertManager。