

1. 安装 Elasticsearch Operator

登录到 OpenShift Container Platform Web 控制台，导航到 Operators→OperatorHub。在过滤器框中输入 Elasticsearch 以找到 Elasticsearch Operator。然后单击 Install 进行安装，如下图所示：

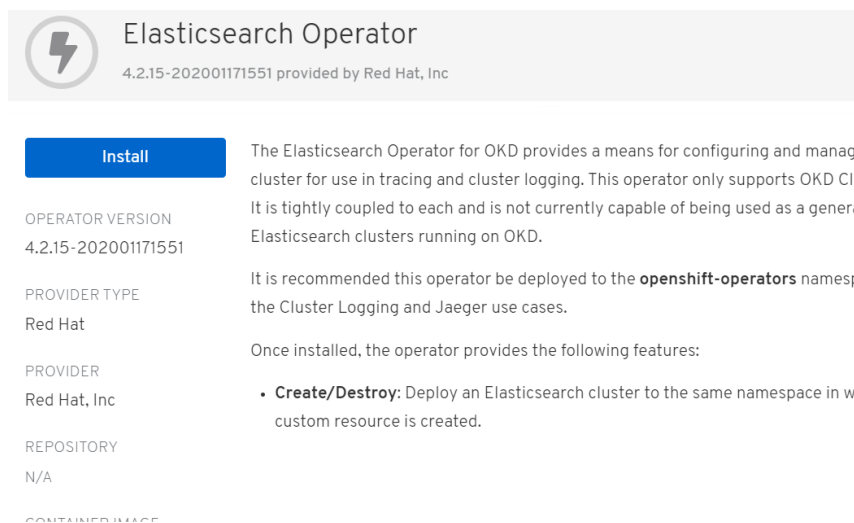


图 8-18 安装 Elasticsearch Operator

在“Create Operator Subscription”页面上，选择 All namespaces on the cluster (default)。这会将操作员安装在默认的 openshift-operators 项目中，并使该 Operator 可用于集群中的所有项目，此外，选择 preview Update Channel、Automatic Approval Strategy，然后单击 Subscribe，如下图所示：

[OperatorHub](#) > Operator Subscription

Create Operator Subscription

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic up

Installation Mode *

- ☒ All namespaces on the cluster (default)
Operator will be available in all namespaces.
- ☐ A specific namespace on the cluster
Operator will be available in a single namespace only.

Update Channel *

- ☐ 4.2
- ☐ 4.2-s390x
- ☐ 4.3
- ☒ preview

Approval Strategy *

- ☒ Automatic
- ☐ Manual



Elasticsearch Operator
provided by Red Hat, Inc

Provided APIs



Elasticsearch
An Elasticsearch cluster instance

Subscribe


Cancel

图 8-19 创建 Operator 订阅

2. 安装 Jaeger Operator

导航到 Operators→OperatorHub。在过滤器框中键入 Jaeger，以找到 Jaeger Operator。

单击 Red Hat 提供的 Jaeger Operator，以显示有关该 Operator 的信息，然后单击 Install，如下图所示：



Jaeger Operator

1.13.1 provided by Red Hat, Inc.

Install

OPERATOR VERSION
1.13.1

PROVIDER TYPE
Red Hat

PROVIDER
Red Hat, Inc.

REPOSITORY
<https://github.com/jaeger/tracing/jaeger-operator>

CONTAINER IMAGE
registry.redhat.io/distributed-tracing/jaeger-rhel7-operator:1.13.1

CREATED AT
2019-07-27 10:00:00

Jaeger, inspired by [Dapper](#) and [OpenZipkin](#), is a distributed tracing system released as open source by Uber Technologies. It is used for monitoring and troubleshooting microservices-based distributed systems.

Core capabilities

Jaeger is used for monitoring and troubleshooting microservices-based distributed systems, including:

- Distributed context propagation
- Distributed transaction monitoring
- Root cause analysis
- Service dependency analysis
- Performance / latency optimization
- OpenTracing compatible data model
- Multiple storage backends: Elasticsearch, Memory.

Operator features

- **Multiple modes** - Supports [allInOne](#) and [production modes of deployment](#).
- **Configuration** - The Operator manages [configuration information](#) when installing Jaeger instances.
- **Storage** - [Configure storage](#) used by Jaeger. By default, [memory](#) is used. Other options include [elasticsearch](#). The operator can delegate creation of an Elasticsearch cluster to the [Elasticsearch Operator](#).

图 8-20 安装 Jaeger Operator

在“Create Operator Subscription”页面上，选择 All namespaces on the cluster (default)、选择 Automatic Approval Strategy，然后点击 Subscribe，如下图 8-21 所示：

OperatorHub > Operator Subscription

Create Operator Subscription

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Installation Mode *

- ☒ All namespaces on the cluster (default)
Operator will be available in all namespaces.
- ☐ A specific namespace on the cluster
Operator will be available in a single namespace only.

Update Channel *

- ☒ stable

Approval Strategy *

- ☒ Automatic
- ☐ Manual

Provided APIs

- Jaeger**
A configuration file for a Jaeger custom resource.

Buttons: [Subscribe](#) [Cancel](#)

图 8-21 创建 Operator 订阅

3. 安装 Kiali Operator

登录到 OpenShift Container Platform Web 控制台。导航到 Operators→OperatorHub。在过滤器框中键入 Kiali 以找到 Kiali Operator。单击 Red Hat 提供的 Kiali Operator 并进行安装，然后点击 Install，如下图 8-22 所示：

Kiali Operator

1.0.9 provided by Red Hat

[Install](#)

OPERATOR VERSION
1.0.9

PROVIDER TYPE
Red Hat

PROVIDER
Red Hat

REPOSITORY
<https://github.com/kiali/kiali>

CONTAINER IMAGE
registry.redhat.io/openshift-service-mesh/kiali-rhel7-operator:1.0.9

CREATED AT
2019-12-17T21:15:24Z

About the managed application

A Microservice Architecture breaks up the monolith into many smaller pieces that are composed together. Patterns to secure the communication between services like fault tolerance (via timeout, retry, circuit breaking, etc.) have come up as well as distributed tracing to be able to see where calls are going.

A service mesh can now provide these services on a platform level and frees the application writers from those tasks. Routing decisions are done at the mesh level.

Kiali works with Istio, in OpenShift or Kubernetes, to visualize the service mesh topology, to provide visibility into features like circuit breakers, request rates and more. It offers insights about the mesh components at different levels, from abstract Applications to Services and Workloads.

See <https://www.kiali.io> to read more.

Accessing the UI

By default, the Kiali operator exposes the Kiali UI as a Route on OpenShift or Ingress on Kubernetes.

On OpenShift, the default root context path is '/' and on Kubernetes it is '/kiali' though you can change this by configuring the 'web_root' setting in the Kiali CR.

About this Operator

图 8-22 安装 Kiali Operator

在“Create Operator Subscription”页面上，选择 All namespaces on the cluster (default)。

选择 stable Update Channel、选择 Automatic Approval Strategy，然后点击 Subscribe，如下图所示：

OperatorHub > Operator Subscription

Create Operator Subscription

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Installation Mode *

- ☒ All namespaces on the cluster (default)
Operator will be available in all namespaces.
- ☐ A specific namespace on the cluster
Operator will be available in a single namespace only.

Update Channel *

- ☒ stable

Approval Strategy *

- ☒ Automatic
- ☐ Manual

Subscribe **Cancel**

Kiali Operator
provided by Red Hat

Provided APIs

- Kiali**
A configuration file for a Kiali installation.

图 8-23 创建 Operator 订阅

4. 安装 Red Hat OpenShift Service Mesh Operator

登录到 OpenShift Container Platform Web 控制台。导航到 Operators→OperatorHub。在过滤器框中键入 Red Hat OpenShift Service Mesh，以查找 Red Hat OpenShift Service Mesh Operator，并进行安装，如下图所示：

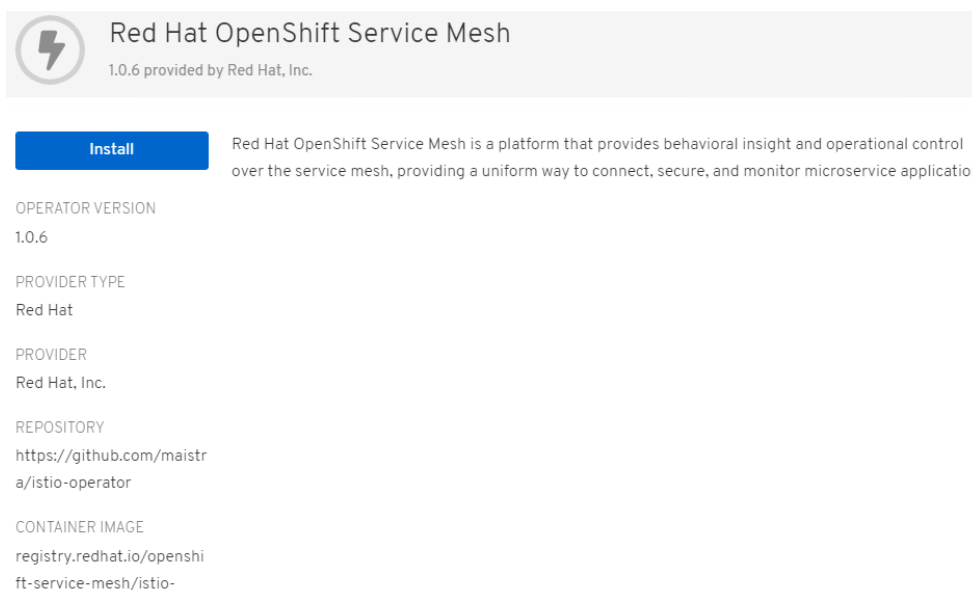


图 8-24 安装 Red Hat OpenShift Service Mesh Operator

在“Create Operator Subscription”页面上，选择 All namespaces on the cluster (default)。选择 1.0 Update Channel、选择 Automatic Approval Strategy，然后点击 Subscribe，如下图 8-25 所示：

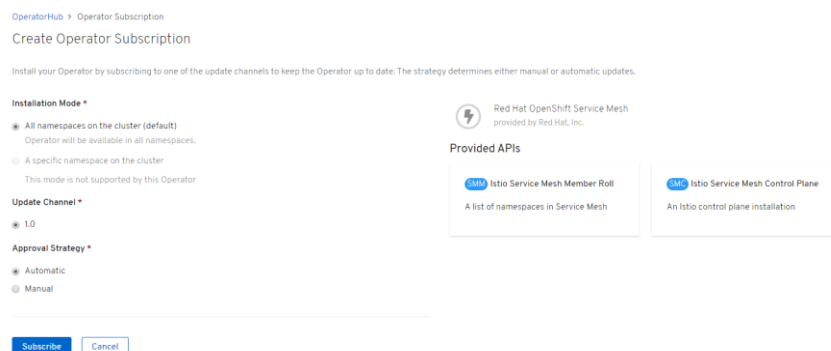


图 8-25 创建 Operator 订阅

几个 Operator 安装完毕后，如下图 8-26 所示：

Installed Operators					
Installed Operators are represented by Cluster Service Versions within this namespace. For more information, see the Operator Lifecycle Manager documentation . Or create an Operator and Cluster Service Version using the Operator SDK .					
					Filter by name...
Name	Namespace	Deployment	Status	Provided APIs	
Elasticsearch Operator 4.1.31-202001140447 provided by Red Hat, Inc.	default	elasticsearch-operator	Copied Up to date	Elasticsearch	
Jaeger Operator 1.13.1 provided by Red Hat, Inc.	default	jaeger-operator	Copied Up to date	Jaeger	
Kiali Operator 1.0.9 provided by Red Hat	default	kiali-operator	Copied Up to date	Kiali Monitoring Dashboard	
Red Hat OpenShift Service Mesh 1.0.6 provided by Red Hat, Inc.	default	istio-operator	Copied Up to date	Istio Service Mesh Member Roll Istio Service Mesh Control Plane	

图 8-26 安装成功的几个 Operator

5. 安装 OpenShift Service Mesh 控制平面

OpenShift Service Mesh 控制平面既可以在 OpenShift Web 界面上安装，也可以通过命令行进行安装。

使用具有 `cluster-admin` 权限的用户登录 OpenShift:

```
[root@oc132-lb weixinyucluster]# oc whoami
```

```
system:admin
```

创建一个名为 `istio-system` 的新项目。

```
oc new-project istio-system
```

接下来，根据红帽提供的示例，书写 `istio-installation.yaml`，如下所示：

```
# cat istio-installation.yaml
```

```
apiVersion: maistra.io/v1

kind: ServiceMeshControlPlane

metadata:
  name: full-install

spec:

  istio:

    global:

      proxy:

        resources:

          requests:

            cpu: 100m

            memory: 128Mi

          limits:

            cpu: 500m

            memory: 128Mi
```

gateways:

istio-egressgateway:

autoscaleEnabled: false

istio-ingressgateway:

autoscaleEnabled: false

mixer:

policy:

autoscaleEnabled: false

telemetry:

autoscaleEnabled: false

resources:

requests:

cpu: 100m

memory: 1G

limits:

cpu: 500m

memory: 4G

pilot:

autoscaleEnabled: false

traceSampling: 100

kiali:

enabled: true

grafana:

enabled: true

```
tracing:

  enabled: true

jaeger:

  template: all-in-one
```

关于具体的参数含义，由于篇幅有限，请参照红帽官方文档。

执行以下命令，部署控制平面。

```
# oc create -n istio-system -f istio-installation.yaml

servicemeshcontrolplane.maistra.io/full-install created
```

执行以下命令，查看控制平面安装状态。

```
$ oc get smcp -n istio-system

NAME          READY
basic-install  True
```

安装完成以后，确认控制平面的 pod 在 Istio-system 项目中成功部署，如下图 8-27 所示：

NAME	READY	STATUS	RESTARTS	AGE
grafana-8644d85f8c-mw749	2/2	Running	0	3h34m
istio-citadel-7494699648-wb86z	1/1	Running	0	3h37m
istio-egressgateway-f9d9c479d-gq9f4	1/1	Running	0	3h35m
istio-galley-6f5859ccf-cfq82	1/1	Running	0	3h36m
istio-ingressgateway-8556d8d864-k4tpj	1/1	Running	0	3h35m
istio-pilot-6948f5d86b-sm8ml	2/2	Running	0	3h35m
istio-policy-55c6789777-zxz5w	2/2	Running	0	3h36m
istio-sidecar-injector-6484bd7665-66tz5	1/1	Running	0	3h34m
istio-telemetry-77b5bcfcb5-lwx88	2/2	Running	0	3h36m
jaeger-6966d9545b-465ms	1/1	Running	0	130m
kiali-55bc44c96b-dq9wn	1/1	Running	0	119m
prometheus-7f454b6b8b-bsn9r	2/2	Running	0	3h37m

图 8-27 Istio 控制平面安装完成

6. 创建 Red Hat OpenShift Service Mesh member roll

ServiceMeshMemberRoll 列出了属于控制平面的项目。只有 ServiceMeshMemberRoll 中列出的项目才受控制平面的影响。在将项目添加到特定控制平面部署的成员卷之前，该

项目不属于服务网格。我们必须在与 ServiceMeshControlPlane 相同的项目中创建一个名为 default 的 ServiceMeshMemberRoll 资源。我们可以通过 OpenShift Web 页面创建 ServiceMeshMemberRoll，也可以通过命令行创建。这里我们通过命令行进行展示。

首先创建 servicemeshmemberroll-default.yaml，示例如下所示：

```
# cat servicemeshmemberroll-default.yaml

apiVersion: maistra.io/v1

kind: ServiceMeshMemberRoll

metadata:

  name: default

  namespace: istio-system

spec:

  members:

    # a list of projects joined into the service mesh

    - demo1

    - davidproject

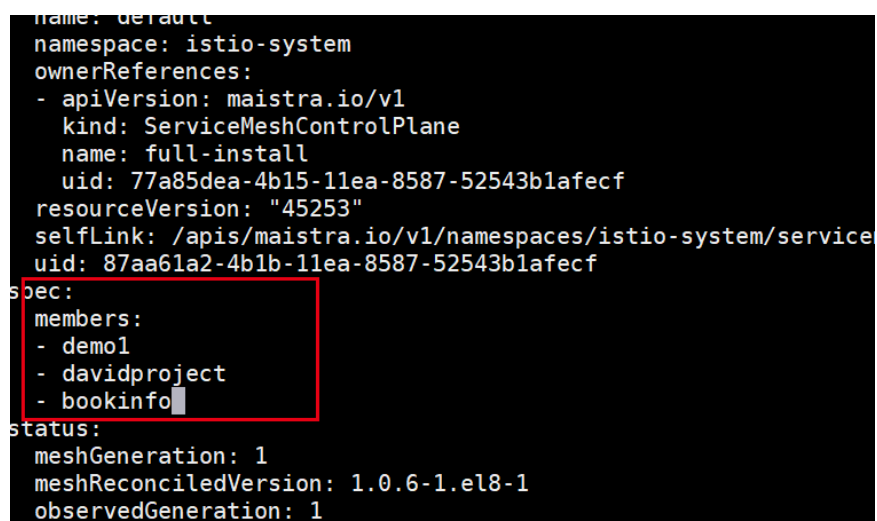
# oc create -n istio-system -f servicemeshmemberroll-default.yaml

servicemeshmemberroll.maistra.io/default created
```

如果想修改项目列表的话，可以使用如下命令行调整：

```
#oc edit smmr -n istio-system
```

在下图红框的位置进行修改，增加项目名称，如下图 8-28 所示：



```
name: default
namespace: istio-system
ownerReferences:
- apiVersion: maistra.io/v1
  kind: ServiceMeshControlPlane
  name: full-install
  uid: 77a85dea-4b15-11ea-8587-52543b1afecf
resourceVersion: "45253"
selfLink: /apis/maistra.io/v1/namespaces/istio-system/service
uid: 87aa61a2-4b1b-11ea-8587-52543b1afecf
spec:
  members:
  - demo1
  - davidproject
  - bookinfo
status:
  meshGeneration: 1
  meshReconciledVersion: 1.0.6-1.el8-1
  observedGeneration: 1
```

图 8-28 修改项目列表

所以，我们要将一个项目中的 pod 纳入 Istio 管理，需要在项目中创建 pod 之前，将项目的名称添加到 smmr 中。加入以后，不代表项目中新建的 pod 会自动被注入 Sidecar，因为上述操作并没有在 deployments 上打标签。将项目名称加入到 smmr 列表，还是需要在 pod 的 deployments 中增加以下注释，才能完成 Sidecar 注入。

```
# oc describe deployments details-v1 |grep -i inject
```

```
Annotations: sidecar.istio.io/inject: true
```

7. 更新 Mixer policy 策略

默认情况下，Mixer 策略处于禁用状态，因此需要手工启动，运行以下命令以检查当前的 Mixer 策略实施状态：

```
# oc get cm -n istio-system istio -o jsonpath='{.data.mesh}' | grep disablePolicyChecks
```

```
disablePolicyChecks: true
```

如果 disablePolicyChecks: true，则编辑 Service Mesh ConfigMap：

```
#oc edit cm -n istio-system istio
```

将配置文件中对应配置改成如下设置（共有两处），然后保存退出。

```
\ndisablePolicyChecks: false
```

重新检查 Mixer 策略状态以确保将其设置为 false。

```
# oc get cm -n istio-system istio -o jsonpath='{.data.mesh}' | grep disablePolicyChecks
```

```
disablePolicyChecks: false
```