



Advanced Configuration Options For OpenShift

NetApp Solutions

NetApp
March 11, 2022

This PDF was generated from https://docs.netapp.com/us-en/netapp-solutions/containers/rh-os-n_LB_MetalLB.html on March 11, 2022. Always check docs.netapp.com for the latest.

Table of Contents

- Advanced Configuration Options For OpenShift 1
 - Exploring load balancer options: Red Hat OpenShift with NetApp 1
 - Creating Private Image Registries 20

Advanced Configuration Options For OpenShift

Exploring load balancer options: Red Hat OpenShift with NetApp

In most cases, Red Hat OpenShift makes applications available to the outside world through routes. A service is exposed by giving it an externally reachable hostname. The defined route and the endpoints identified by its service can be consumed by an OpenShift router to provide this named connectivity to external clients.

However in some cases, applications require the deployment and configuration of customized load balancers to expose the appropriate services. One example of this is NetApp Astra Control Center. To meet this need, we have evaluated a number of custom load balancer options. Their installation and configuration are described in this section.

The following pages have additional information about load balancer options validated in the Red Hat OpenShift with NetApp solution:

- [MetalLB](#)
- [F5 BIG-IP](#)

Next: [Solution validation/use cases: Red Hat OpenShift with NetApp](#).

Installing MetalLB load balancers: Red Hat OpenShift with NetApp

This page lists the installation and configuration instructions for the MetalLB load balancer.

MetalLB is a self-hosted network load balancer installed on your OpenShift cluster that allows the creation of OpenShift services of type load balancer in clusters that do not run on a cloud provider. The two main features of MetalLB that work together to support LoadBalancer services are address allocation and external announcement.

MetalLB configuration options

Based on how MetalLB announces the IP address assigned to LoadBalancer services outside of the OpenShift cluster, it operates in two modes:

- **Layer 2 mode.** In this mode, one node in the OpenShift cluster takes ownership of the service and responds to ARP requests for that IP to make it reachable outside of the OpenShift cluster. Because only the node advertises the IP, it has a bandwidth bottleneck and slow failover limitations. For more information, see the documentation [here](#).
- **BGP mode.** In this mode, all nodes in the OpenShift cluster establish BGP peering sessions with a router and advertise the routes to forward traffic to the service IPs. The prerequisite for this is to integrate MetalLB with a router in that network. Owing to the hashing mechanism in BGP, it has certain limitation when IP-to-Node mapping for a service changes. For more information, refer to the documentation [here](#).



For the purpose of this document, we are configuring MetalLB in layer-2 mode.

Installing The MetalLB Load Balancer

1. Download the MetalLB resources.

```
[netapp-user@rhel7 ~]$ wget
https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/namespace.yaml
[netapp-user@rhel7 ~]$ wget
https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/metallb.yaml
```

2. Edit file `metallb.yaml` and remove `spec.template.spec.securityContext` from controller Deployment and the speaker DaemonSet.

Lines to be deleted:

```
securityContext:
  runAsNonRoot: true
  runAsUser: 65534
```

3. Create the `metallb-system` namespace.

```
[netapp-user@rhel7 ~]$ oc create -f namespace.yaml
namespace/metallb-system created
```

4. Create the MetalLB CR.

```
[netapp-user@rhel7 ~]$ oc create -f metallb.yaml
podsecuritypolicy.policy/controller created
podsecuritypolicy.policy/speaker created
serviceaccount/controller created
serviceaccount/speaker created
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created
role.rbac.authorization.k8s.io/config-watcher created
role.rbac.authorization.k8s.io/pod-lister created
role.rbac.authorization.k8s.io/controller created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller
created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker
created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
rolebinding.rbac.authorization.k8s.io/controller created
daemonset.apps/speaker created
deployment.apps/controller created
```

5. Before configuring the MetalLB speaker, grant the speaker DaemonSet elevated privileges so that it can perform the networking configuration required to make the load balancers work.

```
[netapp-user@rhel7 ~]$ oc adm policy add-scc-to-user privileged -n metallb-system -z speaker
clusterrole.rbac.authorization.k8s.io/system:openshift:scc:privileged
added: "speaker"
```

6. Configure MetalLB by creating a ConfigMap in the metallb-system namespace.

```
[netapp-user@rhel7 ~]$ vim metallb-config.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 10.63.17.10-10.63.17.200

[netapp-user@rhel7 ~]$ oc create -f metallb-config.yaml
configmap/config created
```

7. Now when loadbalancer services are created, MetalLB assigns an externalIP to the services and advertises the IP address by responding to ARP requests.



If you wish to configure MetalLB in BGP mode, skip step 6 above and follow the procedure in the MetalLB documentation [here](#).

Next: [Solution validation/use cases: Red Hat OpenShift with NetApp](#).

Installing F5 BIG-IP Load Balancers

F5 BIG-IP is an Application Delivery Controller (ADC) that offers a broad set of advanced production-grade traffic management and security services like L4-L7 load balancing, SSL/TLS offload, DNS, firewall and many more. These services drastically increase the availability, security and performance of your applications.

F5 BIG-IP can be deployed and consumed in various ways, on dedicated hardware, in the cloud, or as a virtual appliance on-premises. Refer to the documentation [here](#) to explore and deploy F5 BIG-IP as per requirement.

For efficient integration of F5 BIG-IP services with Red Hat OpenShift, F5 offers the BIG-IP Container Ingress

Service (CIS). CIS is installed as a controller pod that watches OpenShift API for certain Custom Resource Definitions (CRDs) and manages the F5 BIG-IP system configuration. F5 BIG-IP CIS can be configured to control service types LoadBalancers and Routes in OpenShift.

Further, for automatic IP address allocation to service the type LoadBalancer, you can utilize the F5 IPAM controller. The F5 IPAM controller is installed as a controller pod that watches OpenShift API for LoadBalancer services with an ipamLabel annotation to allocate the IP address from a preconfigured pool.

This page lists the installation and configuration instructions for F5 BIG-IP CIS and IPAM controller. As a prerequisite, you must have an F5 BIG-IP system deployed and licensed. It must also be licensed for SDN services, which are included by default with the BIG-IP VE base license.



F5 BIG-IP can be deployed in standalone or cluster mode. For the purpose of this validation, F5 BIG-IP was deployed in standalone mode, but, for production purposes, it is preferred to have a cluster of BIG-IPs to avoid a single point of failure.



An F5 BIG-IP system can be deployed on dedicated hardware, in the cloud, or as a virtual appliance on-premises with versions greater than 12.x for it to be integrated with F5 CIS. For the purpose of this document, the F5 BIG-IP system was validated as a virtual appliance, for example using the BIG-IP VE edition.

Validated releases

Technology	Software version
Red Hat OpenShift	4.6 EUS, 4.7
F5 BIG-IP VE edition	16.1.0
F5 Container Ingress Service	2.5.1
F5 IPAM Controller	0.1.4
F5 AS3	3.30.0

Installation

1. Install the F5 Application Services 3 extension to allow BIG-IP systems to accept configurations in JSON instead of imperative commands. Go to [F5 AS3 GitHub repository](#), and download the latest RPM file.
2. Log into F5 BIG-IP system, navigate to iApps > Package Management LX and click Import.
3. Click Choose File and select the downloaded AS3 RPM file, click OK, and then click Upload.



4. Confirm that the AS3 extension is installed successfully.



- Next configure the resources required for communication between OpenShift and BIG-IP systems. First create a tunnel between OpenShift and the BIG-IP server by creating a VXLAN tunnel interface on the BIG-IP system for OpenShift SDN. Navigate to Network > Tunnels > Profiles, click Create, and set the Parent Profile to vxlan and the Flooding Type to Multicast. Enter a name for the profile and click Finished.



- Navigate to Network > Tunnels > Tunnel List, click Create, and enter the name and local IP address for the tunnel. Select the tunnel profile that was created in the previous step and click Finished.



- Log into the Red Hat OpenShift cluster with cluster-admin privileges.

8. Create a hostsubnet on OpenShift for the F5 BIG-IP server, which extends the subnet from the OpenShift cluster to the F5 BIG-IP server. Download the host subnet YAML definition.

```
wget https://github.com/F5Networks/k8s-bigip-ctlr/blob/master/docs/config_examples/openshift/f5-kctlr-openshift-hostsubnet.yaml
```

9. Edit the host subnet file and add the BIG-IP VTEP (VXLAN tunnel) IP for the OpenShift SDN.

```
apiVersion: v1
kind: HostSubnet
metadata:
  name: f5-server
  annotations:
    pod.network.openshift.io/fixed-vnid-host: "0"
    pod.network.openshift.io/assign-subnet: "true"
# provide a name for the node that will serve as BIG-IP's entry into the
cluster
host: f5-server
# The hostIP address will be the BIG-IP interface address routable to
the
# OpenShift Origin nodes.
# This address is the BIG-IP VTEP in the SDN's VXLAN.
hostIP: 10.63.172.239
```



Change the hostIP and other details as applicable to your environment.

10. Create the HostSubnet resource.

```
[admin@rhel-7 ~]$ oc create -f f5-kctlr-openshift-hostsubnet.yaml

hostsubnet.network.openshift.io/f5-server created
```

11. Get the cluster IP subnet range for the host subnet created for the F5 BIG-IP server.


```
[admin@rhel-7 ~]$ oc get hostssubnet
```

NAME	HOST	HOST IP
SUBNET	EGRESS CIDRS	EGRESS IPS
f5-server	f5-server	10.63.172.239
10.131.0.0/23		
ocp-vmw-nszws-master-0	ocp-vmw-nszws-master-0	10.63.172.44
10.128.0.0/23		
ocp-vmw-nszws-master-1	ocp-vmw-nszws-master-1	10.63.172.47
10.130.0.0/23		
ocp-vmw-nszws-master-2	ocp-vmw-nszws-master-2	10.63.172.48
10.129.0.0/23		
ocp-vmw-nszws-worker-r8fh4	ocp-vmw-nszws-worker-r8fh4	10.63.172.7
10.130.2.0/23		
ocp-vmw-nszws-worker-tvr46	ocp-vmw-nszws-worker-tvr46	10.63.172.11
10.129.2.0/23		
ocp-vmw-nszws-worker-wdxhg	ocp-vmw-nszws-worker-wdxhg	10.63.172.24
10.128.2.0/23		
ocp-vmw-nszws-worker-wg8r4	ocp-vmw-nszws-worker-wg8r4	10.63.172.15
10.131.2.0/23		
ocp-vmw-nszws-worker-wtgfw	ocp-vmw-nszws-worker-wtgfw	10.63.172.17
10.128.4.0/23		

12. Create a self IP on OpenShift VXLAN with an IP in OpenShift's host subnet range corresponding to the F5 BIG-IP server. Log into the F5 BIG-IP system, navigate to Network > Self IPs and click Create. Enter an IP from the cluster IP subnet created for F5 BIG-IP host subnet, select the VXLAN tunnel, and enter the other details. Then click Finished.

Network » Self IPs » New Self IP...

Configuration

Name	10.131.0.60
IP Address	10.131.0.60
Netmask	255.252.0.0
VLAN / Tunnel	openshift_vxla ▾
Port Lockdown	Allow All ▾
Traffic Group	<input type="checkbox"/> Inherit traffic group from current partition / path traffic-group-local-only (non-floating) ▾
Service Policy	None ▾

Cancel Repeat Finished

13. Create a partition in the F5 BIG-IP system to be configured and used with CIS. Navigate to System > Users > Partition List, click Create, and enter the details. Then click Finished.

System » Users : Partition List » New Partition...

Properties

Partition Name	<input type="text" value="ocp-vmw"/>
Partition Default Route Domain	<input type="text" value="0"/>
Description	<div></div> <div><input type="checkbox"/> Extend Text Area <input type="checkbox"/> Wrap Text</div>

Redundant Device Configuration

Device Group	<input checked="" type="checkbox"/> Inherit device group from root folder <input type="text" value="None"/>
Traffic Group	<input checked="" type="checkbox"/> Inherit traffic group from root folder <input type="text" value="traffic-group-1 (floating)"/>



F5 recommends that no manual configuration be done on the partition that is managed by CIS.

14. Install the F5 BIG-IP CIS using the operator from OperatorHub. Log into the Red Hat OpenShift cluster with cluster-admin privileges and create a secret with F5 BIG-IP system login credentials, which is a prerequisite for the operator.

```
[admin@rhel-7 ~]$ oc create secret generic bigip-login -n kube-system
--from-literal=username=admin --from-literal=password=admin

secret/bigip-login created
```

15. Install the F5 CIS CRDs.

```
[admin@rhel-7 ~]$ oc apply -f
https://raw.githubusercontent.com/F5Networks/k8s-bigip-
ctlr/master/docs/config_examples/crd/Install/customresourcedefinitions.y
ml

customresourcedefinition.apiextensions.k8s.io/virtualservers.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/tlsprofiles.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/transportservers.cis.f5.co
m created
customresourcedefinition.apiextensions.k8s.io/externaldnss.cis.f5.com
created
customresourcedefinition.apiextensions.k8s.io/ingresslinks.cis.f5.com
created
```

16. Navigate to Operators > OperatorHub, search for the keyword F5, and click the F5 Container Ingress Service tile.

OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through [Red Hat Marketplace](#). You can install Operators on your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the [Developer Catalog](#) providing a self-service experience.

The screenshot shows the OperatorHub interface. On the left is a sidebar with categories: All Items, AI/Machine Learning, Application Runtime, Big Data, Cloud Provider, Database, Developer Tools, Development Tools, Drivers And Plugins, Integration & Delivery, Logging & Tracing, Modernization & Migration, and Monitoring. The main area has a search bar with 'F5' entered and a '1 items' count. Below the search bar, a single result is displayed: the F5 logo, followed by the text 'F5 Container Ingress Services provided by F5 Networks Inc.' and 'Operator to install F5 Container Ingress Services (CIS) for BIG-IP.'

17. Read the operator information and click Install.

 **F5 Container Ingress Services** 1.8.0 provided by F5 Networks Inc. ×

Install

Latest version
1.8.0

Capability level
☒ Basic Install
☐ Seamless Upgrades
☐ Full Lifecycle
☐ Deep Insights
☐ Auto Pilot

Provider type
Certified

Provider
F5 Networks Inc.

Repository
<https://github.com/F5Networks/k8s-bigip-ctlr>

Container image
registry.connect.redhat.com/f5networks/k8s-bigip-ctlr

Introduction

This Operator installs F5 Container Ingress Services (CIS) for BIG-IP in your Cluster. This enables to configure and deploy CIS using Helm Charts.

F5 Container Ingress Services for BIG-IP

F5 Container Ingress Services (CIS) integrates with container orchestration environments to dynamically create L4/L7 services on F5 BIG-IP systems, and load balance network traffic across the services. Monitoring the orchestration API server, CIS is able to modify the BIG-IP system configuration based on changes made to containerized applications.

Documentation

Refer to F5 documentation

- CIS on OpenShift (<https://clouddocs.f5.com/containers/latest/userguide/openshift/>) - OpenShift Routes (<https://clouddocs.f5.com/containers/latest/userguide/routes.html>)

Prerequisites

Create BIG-IP login credentials for use with Operator Helm charts. A basic way be,

```
oc create secret generic <SECRET-NAME> -n kube-system --from-literal=username=<USERNAME> --from-literal=password=<PASSWORD>
```

18. On the Install operator screen, leave all default parameters, and click Install.

Install Operator

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.

Update channel *

☒ beta

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.

Installed Namespace *

PR openshift-operators

Approval strategy *

- ☒ Automatic
- ☐ Manual

Install

Cancel



F5 Container Ingress Services
provided by F5 Networks Inc.

Provided APIs

FBIC F5BigIpCtrlr

This CRD provides kind `F5BigIpCtrlr` to configure and deploy F5 BIG-IP Controller.

19. It takes a while to install the operator.



20. After the operator is installed, the Installation Successful message is displayed.

21. Navigate to Operators > Installed Operators, click F5 Container Ingress Service, and then click Create Instance under the F5BigIpCtrlr tile.

[Installed Operators](#) > Operator details



F5 Container Ingress Services
1.8.0 provided by F5 Networks Inc.

[Details](#)

[YAML](#)

[Subscription](#)

[Events](#)

[F5BigIpCtrlr](#)

Provided APIs

FBIC F5BigIpCtrlr

This CRD provides kind `F5BigIpCtrlr` to configure and deploy F5 BIG-IP Controller.

[+ Create instance](#)

22. Click YAML View and paste the following content after updating the necessary parameters.



Update the parameters `bigip_partition`, ``openshift_sdn_name``, `bigip_url` and `bigip_login_secret` below to reflect the values for your setup before copying the content.

```

apiVersion: cis.f5.com/v1
kind: F5BigIpCtlr
metadata:
  name: f5-server
  namespace: openshift-operators
spec:
  args:
    log_as3_response: true
    agent: as3
    log_level: DEBUG
    bigip_partition: ocp-vmw
    openshift_sdn_name: /Common/openshift_vxlan
    bigip_url: 10.61.181.19
    insecure: true
    pool-member-type: cluster
    custom_resource_mode: true
    as3_validation: true
    ipam: true
    manage_configmaps: true
  bigip_login_secret: bigip-login
  image:
    pullPolicy: Always
    repo: f5networks/cntr-ingress-svcs
    user: registry.connect.redhat.com
  namespace: kube-system
  rbac:
    create: true
  resources: {}
  serviceAccount:
    create: true
  version: latest

```

23. After pasting this content, click Create. This installs the CIS pods in the kube-system namespace.

Pods Create Pod

Filter Name Search by name...

Name ↑	Status ↓	Ready ↓	Restarts ↓	Owner ↓	Memory ↓	CPU ↓
 f5-server-f5-bigip-ctlr-5d7578667d-qxdgj	 Running	1/1	0	 f5-server-f5-bigip-ctlr-5d7578667d	61.1 MiB	0.003 cores



Red Hat OpenShift, by default, provides a way to expose the services via Routes for L7 load balancing. An inbuilt OpenShift router is responsible for advertising and handling traffic for these routes. However, you can also configure the F5 CIS to support the Routes through an external F5 BIG-IP system, which can run either as an auxiliary router or a replacement to the self-hosted OpenShift router. CIS creates a virtual server in the BIG-IP system that acts as a router for the OpenShift routes, and BIG-IP handles the advertisement and traffic routing. Refer to the documentation here for information on parameters to enable this feature. Note that these parameters are defined for OpenShift Deployment resource in the apps/v1 API. Therefore, when using these with the F5BigIpCtrl resource cis.f5.com/v1 API, replace the hyphens (-) with underscores (_) for the parameter names.

24. The arguments that are passed to the creation of CIS resources include `ipam: true` and `custom_resource_mode: true`. These parameters are required for enabling CIS integration with an IPAM controller. Verify that the CIS has enabled IPAM integration by creating the F5 IPAM resource.

```
[admin@rhel-7 ~]$ oc get f5ipam -n kube-system
```

NAMESPACE	NAME	AGE
kube-system	ipam.10.61.181.19.ocp-vmw	43s

25. Create the service account, role and rolebinding required for the F5 IPAM controller. Create a YAML file and paste the following content.


```
[admin@rhel-7 ~]$ vi f5-ipam-rbac.yaml

kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctrlr-clusterrole
rules:
  - apiGroups: ["fic.f5.com"]
    resources: ["ipams","ipams/status"]
    verbs: ["get", "list", "watch", "update", "patch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctrlr-clusterrole-binding
  namespace: kube-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: ipam-ctrlr-clusterrole
subjects:
  - apiGroup: ""
    kind: ServiceAccount
    name: ipam-ctrlr
    namespace: kube-system
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ipam-ctrlr
  namespace: kube-system
```

26. Create the resources.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-rbac.yaml

clusterrole.rbac.authorization.k8s.io/ipam-ctrlr-clusterrole created
clusterrolebinding.rbac.authorization.k8s.io/ipam-ctrlr-clusterrole-
binding created
serviceaccount/ipam-ctrlr created
```

27. Create a YAML file and paste the F5 IPAM deployment definition provided below.



Update the ip-range parameter in spec.template.spec.containers[0].args below to reflect the ipamLabels and IP address ranges corresponding to your setup.



ipamLabels [range1 and range2 in below example] are required to be annotated for the services of type LoadBalancer for the IPAM controller to detect and assign an IP address from the defined range.

```
[admin@rhel-7 ~]$ vi f5-ipam-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    name: f5-ipam-controller
  name: f5-ipam-controller
  namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: f5-ipam-controller
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: f5-ipam-controller
    spec:
      containers:
      - args:
        - --orchestration=openshift
        - --ip-range='{ "range1": "10.63.172.242-10.63.172.249",
"range2": "10.63.170.111-10.63.170.129" }'
        - --log-level=DEBUG
        command:
        - /app/bin/f5-ipam-controller
        image: registry.connect.redhat.com/f5networks/f5-ipam-
controller:latest
        imagePullPolicy: IfNotPresent
        name: f5-ipam-controller
        dnsPolicy: ClusterFirst
        restartPolicy: Always
        schedulerName: default-scheduler
        securityContext: {}
        serviceAccount: ipam-ctrlr
        serviceAccountName: ipam-ctrlr
```

28. Create the F5 IPAM controller deployment.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-deployment.yaml  
  
deployment/f5-ipam-controller created
```

29. Verify the F5 IPAM controller pods are running.

```
[admin@rhel-7 ~]$ oc get pods -n kube-system
```

NAME	READY	STATUS	RESTARTS
AGE			
f5-ipam-controller-5986cff5bd-2bvn6	1/1	Running	0
30s			
f5-server-f5-bigip-ctlr-5d7578667d-qxdgj	1/1	Running	0
14m			

30. Create the F5 IPAM schema.

```
[admin@rhel-7 ~]$ oc create -f  
https://raw.githubusercontent.com/F5Networks/f5-ipam-  
controller/main/docs/_static/schemas/ipam_schema.yaml  
  
customresourcedefinition.apiextensions.k8s.io/ipams.fic.f5.com
```

Verification

1. Create a service of type LoadBalancer

```
[admin@rhel-7 ~]$ vi example_svc.yaml
```

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    cis.f5.com/ipamLabel: range1
  labels:
    app: f5-demo-test
  name: f5-demo-test
  namespace: default
spec:
  ports:
  - name: f5-demo-test
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: f5-demo-test
  sessionAffinity: None
  type: LoadBalancer
```

```
[admin@rhel-7 ~]$ oc create -f example_svc.yaml
```

```
service/f5-demo-test created
```

2. Check if the IPAM controller assigns an external IP to it.

```
[admin@rhel-7 ~]$ oc get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
f5-demo-test	LoadBalancer	172.30.210.108	10.63.172.242
80:32605/TCP	27s		

3. Create a deployment and use the LoadBalancer service that was created.

```
[admin@rhel-7 ~]$ vi example_deployment.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: f5-demo-test
  name: f5-demo-test
spec:
  replicas: 2
  selector:
    matchLabels:
      app: f5-demo-test
  template:
    metadata:
      labels:
        app: f5-demo-test
    spec:
      containers:
      - env:
        - name: service_name
          value: f5-demo-test
        image: nginx
        imagePullPolicy: Always
        name: f5-demo-test
        ports:
        - containerPort: 80
          protocol: TCP
```

```
[admin@rhel-7 ~]$ oc create -f example_deployment.yaml
```

```
deployment/f5-demo-test created
```

4. Check if the pods are running.

```
[admin@rhel-7 ~]$ oc get pods
```

NAME	READY	STATUS	RESTARTS	AGE
f5-demo-test-57c46f6f98-47wvp	1/1	Running	0	27s
f5-demo-test-57c46f6f98-cl2m8	1/1	Running	0	27s

5. Check if the corresponding virtual server is created in the BIG-IP system for the service of type LoadBalancer in OpenShift. Navigate to Local Traffic > Virtual Servers > Virtual Server List.



Next: [Solution Validation/Use Cases: Red Hat OpenShift with NetApp.](#)

Creating Private Image Registries

For most deployments of Red Hat OpenShift, using a public registry like [Quay.io](#) or [DockerHub](#) meets most customer's needs. However there are times when a customer may want to host their own private or customized images.

This procedure documents creating a private image registry which is backed by a persistent volume provided by Astra Trident and NetApp ONTAP.



Astra Control Center requires a registry to host the images the Astra containers require. The following section describes the steps to setup a private registry on Red Hat OpenShift cluster and pushing the images required to support the installation of Astra Control Center.

Creating A private image registry

1. Remove the default annotation from the current default storage class and annotate the Trident-backed storage class as default for the OpenShift cluster.

```
[netapp-user@rhel7 ~]$ oc patch storageclass thin -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class": "false"}}}'
storageclass.storage.k8s.io/thin patched

[netapp-user@rhel7 ~]$ oc patch storageclass ocp-trident -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class": "true"}}}'
storageclass.storage.k8s.io/ocp-trident patched
```

2. Edit the imageregistry operator by entering the following storage parameters in the `spec` section.

```
[netapp-user@rhel7 ~]$ oc edit
configs.imageregistry.operator.openshift.io

storage:
  pvc:
    claim:
```

3. Enter the following parameters in the `spec` section for creating a OpenShift route with a custom hostname. Save and exit.

```
routes:
- hostname: astra-registry.apps.ocp-vmw.cie.netapp.com
  name: netapp-astra-route
```



The above route config is used when you want a custom hostname for your route. If you want OpenShift to create a route with a default hostname, you can add the following parameters to the `spec` section: `defaultRoute: true`.

Custom TLS certificates

When you are using a custom hostname for the route, by default, it uses the default TLS configuration of the OpenShift Ingress operator. However, you can add a custom TLS configuration to the route. To do so, complete the following steps.

- a. Create a secret with the route's TLS certificates and key.

```
[netapp-user@rhel7 ~]$ oc create secret tls astra-route-tls -n
openshift-image-registry -cert/home/admin/netapp-astra/tls.crt
--key=/home/admin/netapp-astra/tls.key
```

- b. Edit the imageregistry operator and add the following parameters to the `spec` section.

```
[netapp-user@rhel7 ~]$ oc edit
configs.imageregistry.operator.openshift.io

routes:
- hostname: astra-registry.apps.ocp-vmw.cie.netapp.com
  name: netapp-astra-route
  secretName: astra-route-tls
```

4. Edit the imageregistry operator again and change the management state of the operator to the `Managed` state. Save and exit.

```
oc edit configs.imageregistry/cluster

managementState: Managed
```

5. If all the prerequisites are satisfied, PVCs, pods, and services are created for the private image registry. In a few minutes, the registry should be up.

```
[netapp-user@rhel7 ~]$oc get all -n openshift-image-registry
```

NAME	READY	STATUS
pod/cluster-image-registry-operator-74f6d954b6-rb7zr	1/1	Running
3		90d
pod/image-pruner-1627257600-f5cpj	0/1	Completed
0		2d9h
pod/image-pruner-1627344000-swqx9	0/1	Completed
0		33h
pod/image-pruner-1627430400-rv5nt	0/1	Completed
0		9h
pod/image-registry-6758b547f-6pnj8	1/1	Running
0		76m
pod/node-ca-bwb5r	1/1	Running
0		90d
pod/node-ca-f8w54	1/1	Running
0		90d
pod/node-ca-gjx7h	1/1	Running
0		90d
pod/node-ca-lcx4k	1/1	Running
0		33d
pod/node-ca-v7zmx	1/1	Running
0		7d21h
pod/node-ca-xpppp	1/1	Running
0		89d

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/image-registry	ClusterIP	172.30.196.167	<none>
5000/TCP			15h
service/image-registry-operator	ClusterIP	None	<none>
60000/TCP			90d

NAME	DESIRED	CURRENT	READY	UP-TO-DATE
AVAILABLE		AGE		
daemonset.apps/node-ca	6	6	6	6
kubernetes.io/os=linux	90d			

NAME	READY	UP-TO-DATE
AVAILABLE		
AGE		
deployment.apps/cluster-image-registry-operator	1/1	1
90d		
deployment.apps/image-registry	1/1	1
15h		

NAME	DESIRED
CURRENT READY AGE	
replicaset.apps/cluster-image-registry-operator-74f6d954b6	1 1
1 90d	
replicaset.apps/image-registry-6758b547f	1 1
1 76m	
replicaset.apps/image-registry-78bfbd7f59	0 0
0 15h	
replicaset.apps/image-registry-7fcc8d6cc8	0 0
0 80m	
replicaset.apps/image-registry-864f88f5b	0 0
0 15h	
replicaset.apps/image-registry-cb47fffb	0 0
0 10h	

NAME	COMPLETIONS	DURATION	AGE
job.batch/image-pruner-1627257600	1/1	10s	2d9h
job.batch/image-pruner-1627344000	1/1	6s	33h
job.batch/image-pruner-1627430400	1/1	5s	9h

NAME	SCHEDULE	SUSPEND	ACTIVE	LAST
SCHEDULE AGE				
cronjob.batch/image-pruner	0 0 * * *	False	0	9h
90d				

NAME	HOST/PORT
PATH SERVICES PORT TERMINATION WILDCARD	
route.route.openshift.io/public-routes	astra-registry.apps.ocp-vmw.cie.netapp.com
image-registry	<all> reencrypt None

6. If you are using the default TLS certificates for the ingress operator OpenShift registry route, you can fetch the TLS certificates using the following command.

```
[netapp-user@rhel7 ~]$ oc extract secret/router-ca --keys=tls.crt -n openshift-ingress-operator
```

7. To allow OpenShift nodes to access and pull the images from the registry, add the certificates to the docker client on the OpenShift nodes. Create a configmap in the `openshift-config` namespace using the TLS certificates and patch it to the cluster image config to make the certificate trusted.

```
[netapp-user@rhel7 ~]$ oc create configmap astra-ca -n openshift-config
--from-file=astra-registry.apps.ocp-vmw.cie.netapp.com=tls.crt

[netapp-user@rhel7 ~]$ oc patch image.config.openshift.io/cluster
--patch '{"spec":{"additionalTrustedCA":{"name":"astra-ca"}}}'
--type=merge
```

8. The OpenShift internal registry is controlled by authentication. All the OpenShift users can access the OpenShift registry, but the operations that the logged in user can perform depends on the user permissions.

- a. To allow a user or a group of users to pull images from the registry, the user(s) must have the registry-viewer role assigned.

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-viewer
ocp-user

[netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-viewer
ocp-user-group
```

- b. To allow a user or group of users to write or push images, the user(s) must have the registry-editor role assigned.

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-editor
ocp-user

[netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-editor
ocp-user-group
```

9. For OpenShift nodes to access the registry and push or pull the images, you need to configure a pull secret.

```
[netapp-user@rhel7 ~]$ oc create secret docker-registry astra-registry-
credentials --docker-server=astra-registry.apps.ocp-vmw.cie.netapp.com
--docker-username=ocp-user --docker-password=password
```

10. This pull secret can then be patched to serviceaccounts or be referenced in the corresponding pod definition.

- a. To patch it to service accounts, run the following command.

```
[netapp-user@rhel7 ~]$ oc secrets link <service_account_name> astra-
registry-credentials --for=pull
```

- b. To reference the pull secret in the pod definition, add the following parameter to the `spec` section.

```
imagePullSecrets:
- name: astra-registry-credentials
```

11. To push or pull an image from workstations apart from OpenShift node, complete the following steps.

- a. Add the TLS certificates to the docker client.

```
[netapp-user@rhel7 ~]$ sudo mkdir /etc/docker/certs.d/astra-registry.apps.ocp-vmw.cie.netapp.com

[netapp-user@rhel7 ~]$ sudo cp /path/to/tls.crt
/etc/docker/certs.d/astra-registry.apps.ocp-vmw.cie.netapp.com
```

- b. Log into OpenShift using the `oc login` command.

```
[netapp-user@rhel7 ~]$ oc login --token=sha256~D49SpB_lesSrJYwrM0LIO
-VRcjWHu0a27vKa0 --server=https://api.ocp-vmw.cie.netapp.com:6443
```

- c. Log into the registry using OpenShift user credentials with the `podman/docker` command.

podman

```
[netapp-user@rhel7 ~]$ podman login astra-registry.apps.ocp-vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t) --tls
-verify=false
```

+

NOTE: If you are using `kubeadmin` user to log into the private registry, then use token instead of password.

docker

```
[netapp-user@rhel7 ~]$ docker login astra-registry.apps.ocp-vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t)
```

+

NOTE: If you are using `kubeadmin` user to log into the private registry, then use token instead of password.

- d. Push or pull the images.

podman

```
[netapp-user@rhel7 ~]$ podman push astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest  
[netapp-user@rhel7 ~]$ podman pull astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```

docker

```
[netapp-user@rhel7 ~]$ docker push astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest  
[netapp-user@rhel7 ~]$ docker pull astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```

Next: [Solution Validation/Use Cases: Red Hat OpenShift with NetApp.](#)

Copyright Information

Copyright © 2022 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.