Hewlett Packard Enterprise

Deployment Guide

Deploying F5 BIG-IP and F5 Container Ingress Services on HPE Synergy for Red Hat OpenShift Container Platform

Contents

Overview

In today's digital world, organizations are under increasing pressure to deliver applications faster while reducing costs. As these applications grow more complex, it puts stress on IT infrastructure, IT teams, and processes. To remain competitive, organizations must adapt quickly and developers need to be more effective, efficient, and agile. Container technology provides the right application platform to help organizations become more responsive and iterate across multiple IT environments as well as develop, deploy, and manage applications faster and scale the applications and infrastructure over time. But implementing a containerized environment across existing infrastructure is a complex undertaking that can require weeks or months to mobilize, particularly for enterprises.

Businesses can address challenges and complexity with scale and automation by deploying F5® container solutions with application performance and security services. This enables advanced application services and native, self-service Ingress control. F5 container solutions scale out apps in containerized environments with orchestration solutions for developers, system teams, and operations. F5 open-source container solutions offer speed and agility when deploying application performance and security services on premises and across multi-cloud services. You can also dynamically configure Ingress control, HTTP routing, and load balancing within the container or Platform-as-a-Service (PaaS) orchestration environment. F5 delivers a rich set of network and application metrics in a data-stream format for export to 3rd-party analytics such as Prometheus.

Red Hat® OpenShift Container Platform on HPE Synergy provides an end-to-end fully integrated container solution that, once assembled, can be configured within hours. This eliminates the complexities associated with implementing a container platform across an enterprise data center and provides the automation of hardware and software configuration to quickly provision and deploy a containerized environment at scale. Red Hat OpenShift Container Platform provides organizations with a reliable platform for deploying and scaling container-based applications and HPE Synergy provides the flexible infrastructure you need to run that container platform to dynamically provision and scale applications, whether they run in VMs or containers, or are hosted on-premises, in the cloud, or somewhere in between.

Ultimately, maintaining manageability while driving scale into container environments means bringing enterprise grade application delivery into the core solution stack. Combining Red Hat OpenShift Container Platform and F5 BIG-IP® and F5 Container Ingress Services on HPE Synergy Composable Infrastructure creates a solution stack that can be programmatically deployed, altered, managed and scaled over time. This document should be used in conjunction with the HPE Synergy and Red Hat OpenShift Reference Configurations

Target audience: Data center managers, enterprise architects, and implementation personnel wishing to learn more about how to deploy F5 container solutions into Red Hat OpenShift Container Platform on HPE Synergy Composable Infrastructure. Familiarity with HPE Synergy, Red Hat OpenShift Container Platform, container-based solutions, Ansible Engine and/or Ansible Tower, and core networking knowledge is assumed.

Solution overview

Hewlett Packard Enterprise and Red Hat began a collaboration on a Reference Architecture (RA) in 2018 which sought to greatly simplify the planning, deployment and consumption of Red Hat OpenShift Container Platform on HPE Synergy while enabling the use of persistent storage in container environments. The solution deploys Red Hat OpenShift Container Platform as a combination of virtual and physical resources. The OpenShift Master, Infrastructure, and etcd functions are deployed as virtual machines running on three (3) HPE Synergy 480 Gen10 Compute Modules managed by HPE OneView and running Red Hat Virtualization Host 4.2 (RHVH) or VMware vSphere® 6.7. Red Hat OpenShift Container Platform worker nodes can be deployed on bare metal on six (6) HPE Synergy 480 Gen10 Compute Modules running Red Hat Enterprise Linux 7.6 or as virtual machines when iSCSI storage is utilized. The hypervisor is installed via a Kickstart file over PXE, and post-

installation configuration steps are performed using Ansible. HPE Nimble Storage or HPE 3PAR StoreServ Storage provides support for both ephemeral and persistent container volumes. Figure 1 provides a high-level, functional view of the RA.

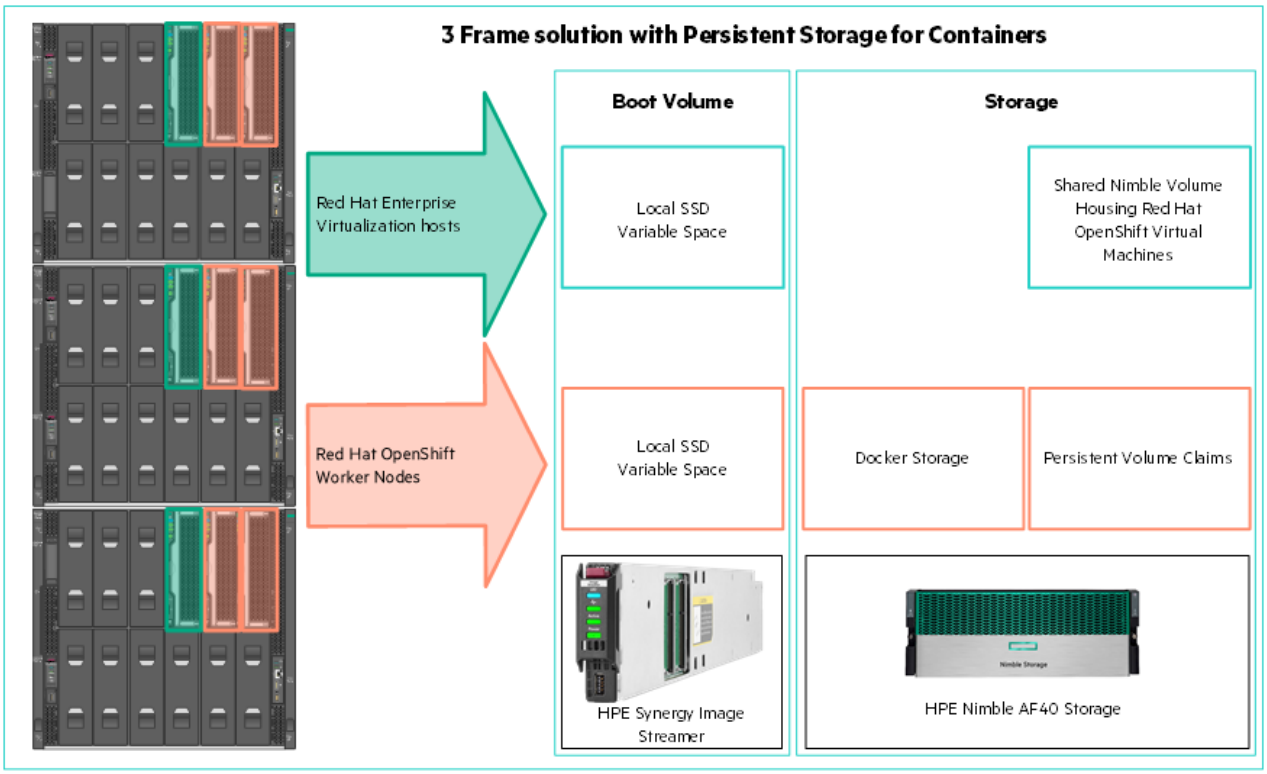

Figure 1. Solution design by function and storage type

Figure 2 shows a three (3) frame implementation with a single HPE Nimble Storage array and HPE networking.

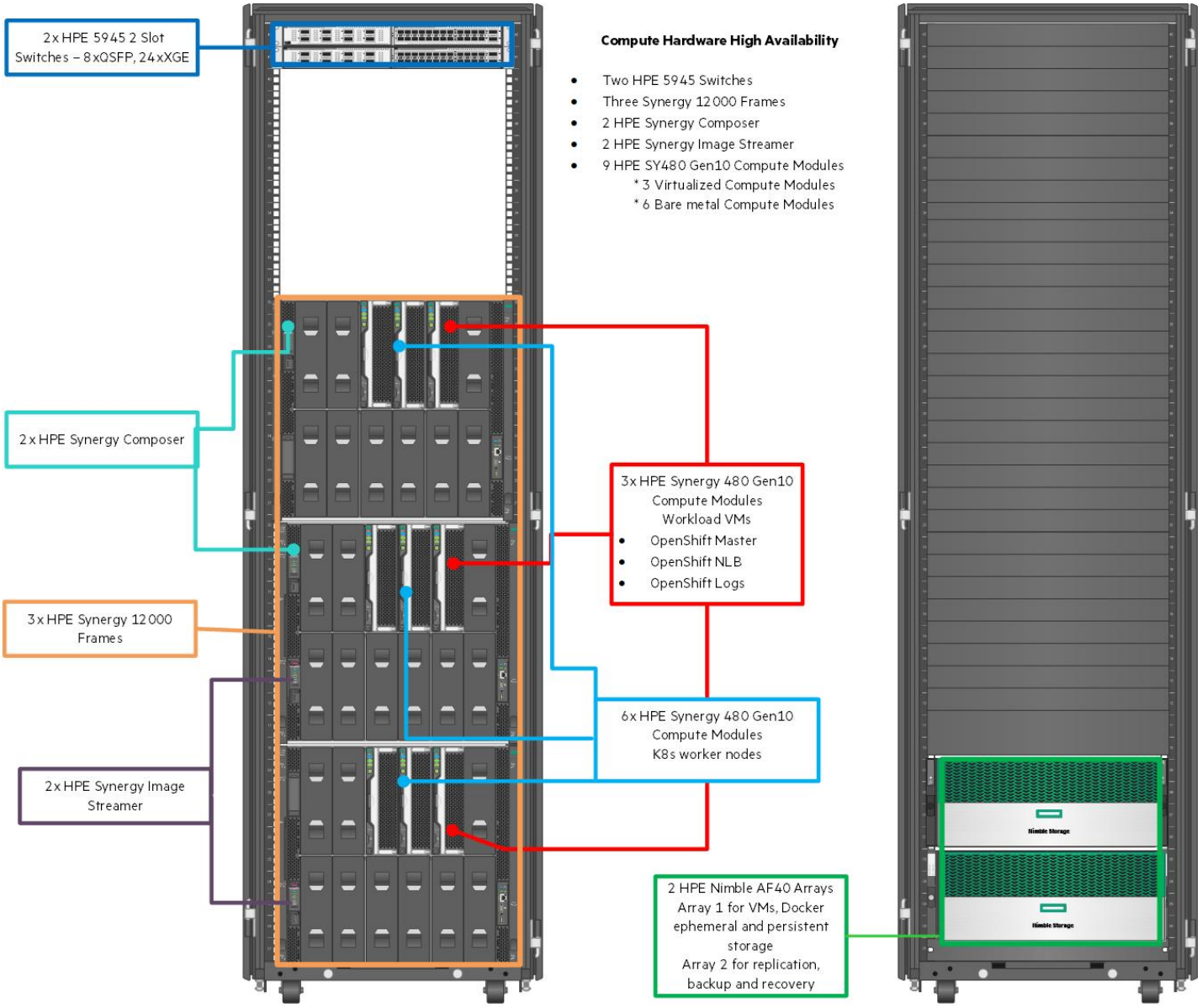

Figure 2. Shows the front facing layout of the hardware within the racks

The RA has been continually updated since initial release to reflect new versions of software, hardware and firmware recipes as well as to integrate value added features. This document describes the addition of F5 Networks as an enterprise load balancing solution and Container Ingress Services for Red Hat OpenShift Container Platform as a part of the overall RA.

Software versions

Table 1 describes the versions of important software utilized in the creation of this solution. This software is considered a prerequisite for the solution and it is assumed that appropriate licensing and subscriptions are in place and that the advice in this document will be combined with the appropriate Deployment Guide for the solution found at https://github.com/hewlettpackard/hpe-solutions-openshift.

Table 1. Major software versions used in solution creation

Component	Version
Red Hat Enterprise Linux Server	7.6
VMware ESXi	6.7
VMware vCenter™ Server Appliance	6.7
Red Hat OpenShift Container Platform	3.11
F5 BIG-IP Virtual Edition	14.1.0.5-0.0

Solution configuration

In order to successfully integrate F5 BIG-IP into the HPE Synergy and Red Hat OpenShift Reference Architecture the following steps should be carried out. For the purposes of this document, VMware vSphere was chosen as the hypervisor for the architecture. Deployment specifics will vary when Red Hat Enterprise Virtualization is used.

1. Follow the appropriate HPE Deployment Guide to install and configure required systems and services to support Red Hat OpenShift.
2. Download and install the BIG-IP Virtual Edition.
3. Configure the BIG-IP to load balance the masters.
4. Install Red Hat OpenShift using the appropriate deployment guide.
5. Configure BIG-IP container integration.
6. Create a sample virtual server.

The steps to configure the environment and the infrastructure as well as to install core networking are described in detail in the deployment guides found at https://github.com/hewlettpackard/hpe-solutions-openshift. The steps will not be repeated here.

Download and install the BIG-IP Virtual Edition

If you do not have an F5 account, register for one and download the BIG-IP Virtual Edition software for your hypervisor. Follow the documentation found at https://techdocs.f5.com/content/kb/en-us/products/big-ip_ltm/manuals/product/bigip-ve-setup-vmware-esxi-13-1-0/_jcr_content/pdfAttach/download/file.res/BIG-IP_Virtual_Edition_and_VMware_ESXi__Setup.pdf.

BIG-IP Configuration

BIG-IP DNS improves the performance and availability of your global applications by sending users to the closest or best-performing physical, virtual, or cloud environment. It also hyperscales and secures your DNS[1] infrastructure from DDoS[2] attacks.

A load balancer is a device that acts as a reverse proxy and distributes network or application traffic across a number of servers. Load balancers are used to increase both the capacity (concurrent users) and reliability of applications. They improve the overall performance of applications by decreasing the burden on servers associated with managing and maintaining application and network sessions, as well as by performing application-specific tasks.

Load balancers are generally grouped into two categories: Layer 4 and Layer 7. Layer 4 load balancers act upon data found in network and transport layer protocols (IP, TCP, FTP, UDP). Layer 7 load balancers distribute requests based upon data found in application layer protocols such as HTTP.

[1] Domain Name System
[2] Distributed Denial of Service

BIG-IP to load balance Red Hat OpenShift Container Platform masters

Red Hat OpenShift Container Platform features a Native High Availability (HA) mode that provides high availability for the masters without the need for Red Hat Enterprise Linux (RHEL) clustering. When installed in Native HA mode, the stateless API component (atomic-openshift-master-api) of the master is split out from the stateful controller component (atomic-openshift-master-controllers). This way the API service can be configured in an active-active mode while the controller service runs as active-passive. Once you have an active-active master API (which includes the Web Console), all that is needed to have a fully HA orchestration infrastructure, is a proper load balancer.

Out of the box, Red Hat OpenShift Container Platform has the ability to install a HA Proxy instance on a host you designate as a lightweight load balancer between masters. However, this only creates another single point of failure. The preferred method is to integrate an enterprise load balancer (LB) such as an F5 BIG-IP. This integration and configuration is depicted in Figure 3 below.

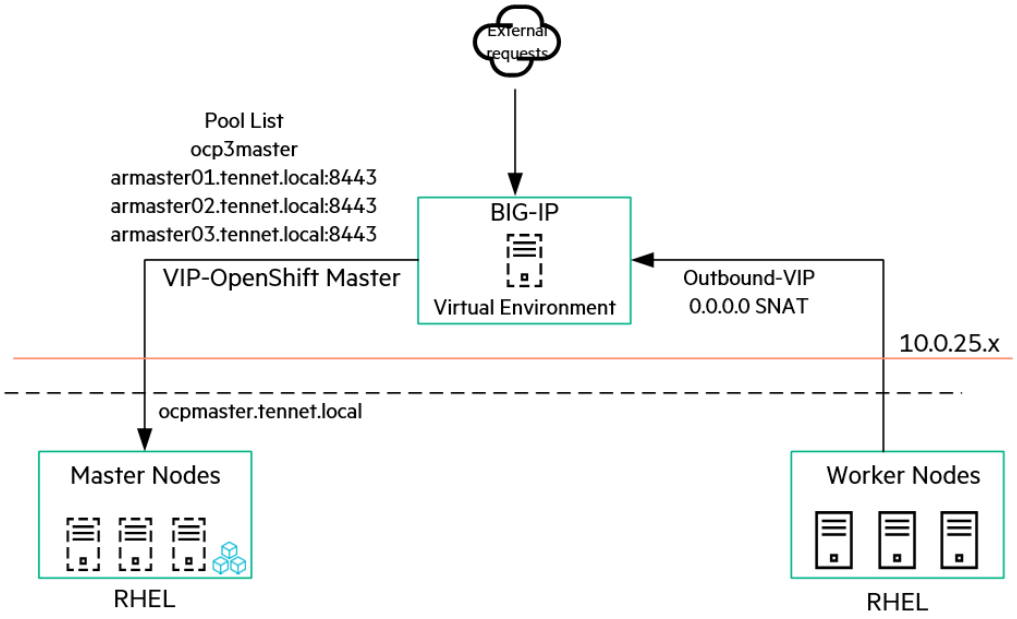

Figure 3. BIG-IP load balancing API requests and providing external access for the nodes

The Virtual IP (VIP) must listen and proxy back to all master hosts on port 8443. This means that the BIG-IP does not terminate SSL, but simply proxies encrypted traffic through to the masters, which then handle termination. This has the advantage of being a fairly simple implementation on the BIG-IP side, and a slightly simpler setup process on the Red Hat OpenShift installation than terminating on the BIG-IP. The drawback of this method is that a self-signed certificate is being presented, so users of the Web Console or API will see untrusted or unknown certificate errors.

Creating BIG-IP virtual server

This section describes the configuration of the F5 BIG-IP virtual server. Prior to beginning, the installer should deploy the appliance on an existing virtual host.

1. Login to the newly created BIG-IP virtual machine and log on to the web console. Navigate to **Local Traffic > Nodes > Create**.

A sample node list, after adding the nodes is shown in Figure 4.

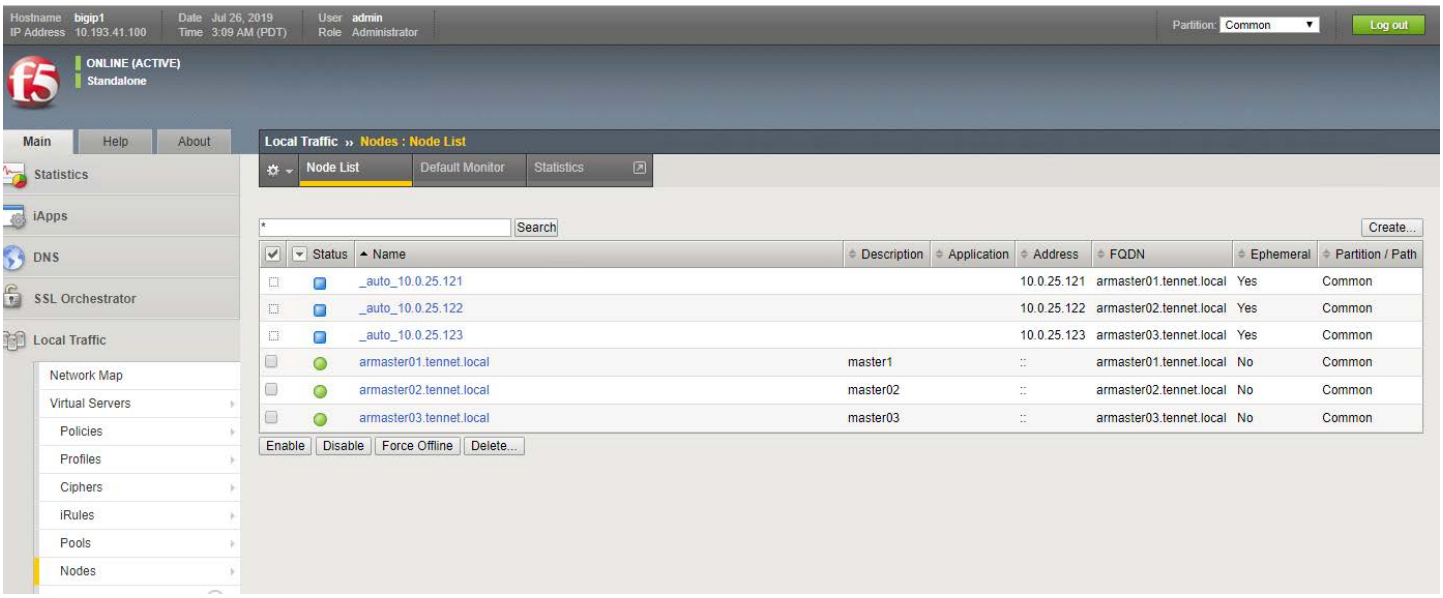

Figure 4. Node list displaying the newly created nodes

2. Once a node is created, create a pool list, From the BIG-IP web, navigate to **Local Traffic > Pools > Pool List** and click on the **Create** button to create a new pool. Provide information in the required fields and then select the created nodes as shown in Figure 5.

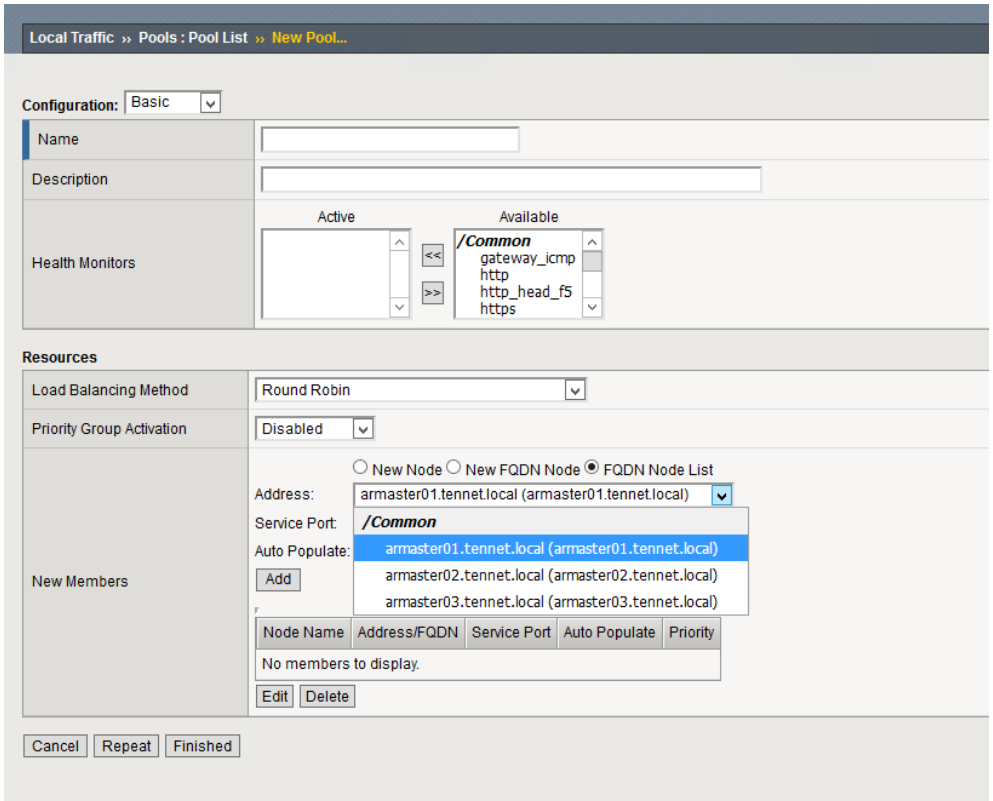

Figure 5. Creating a pool list

3. Figure 6 shows the properties of the newly created pool. In this example it has been named “ocp3-master”.

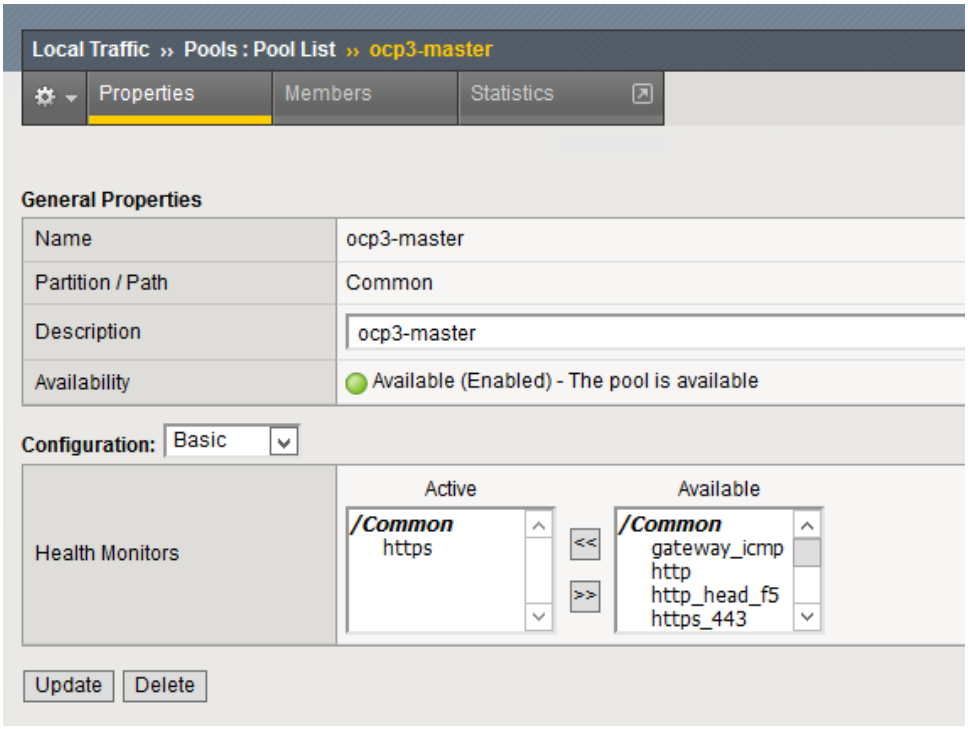

Figure 6. Properties of the pool list

4. Figure 7 shows the members of the newly created pool, ocp3-master.

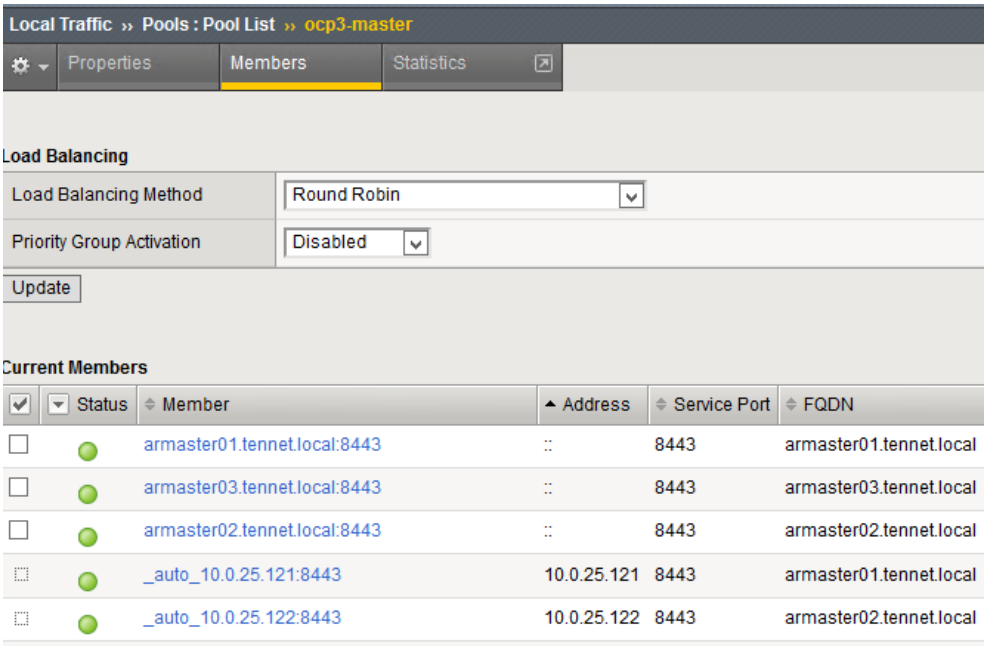

Figure 7. Members of the pool list “ocp3-master”

5. Verify the network mapping by navigate to **Local Traffic** > **Network Map**. Sample configuration is shown in Figure 8.

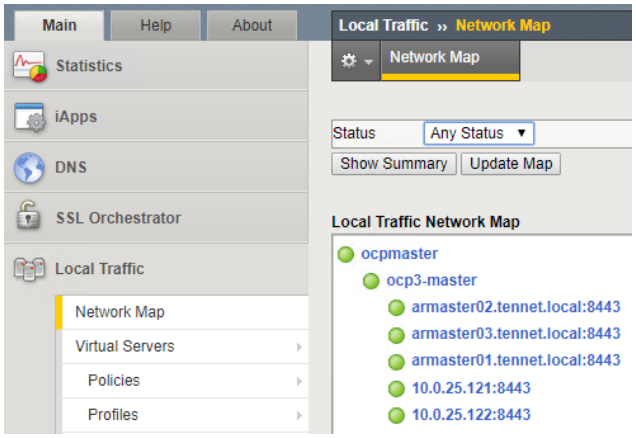

Figure 8. Network map for local traffic

6. The next step is to create a new Virtual Server. From the BIG-IP web console, navigate to **Local Traffic > Virtual Servers > Create** as in Figure 9.

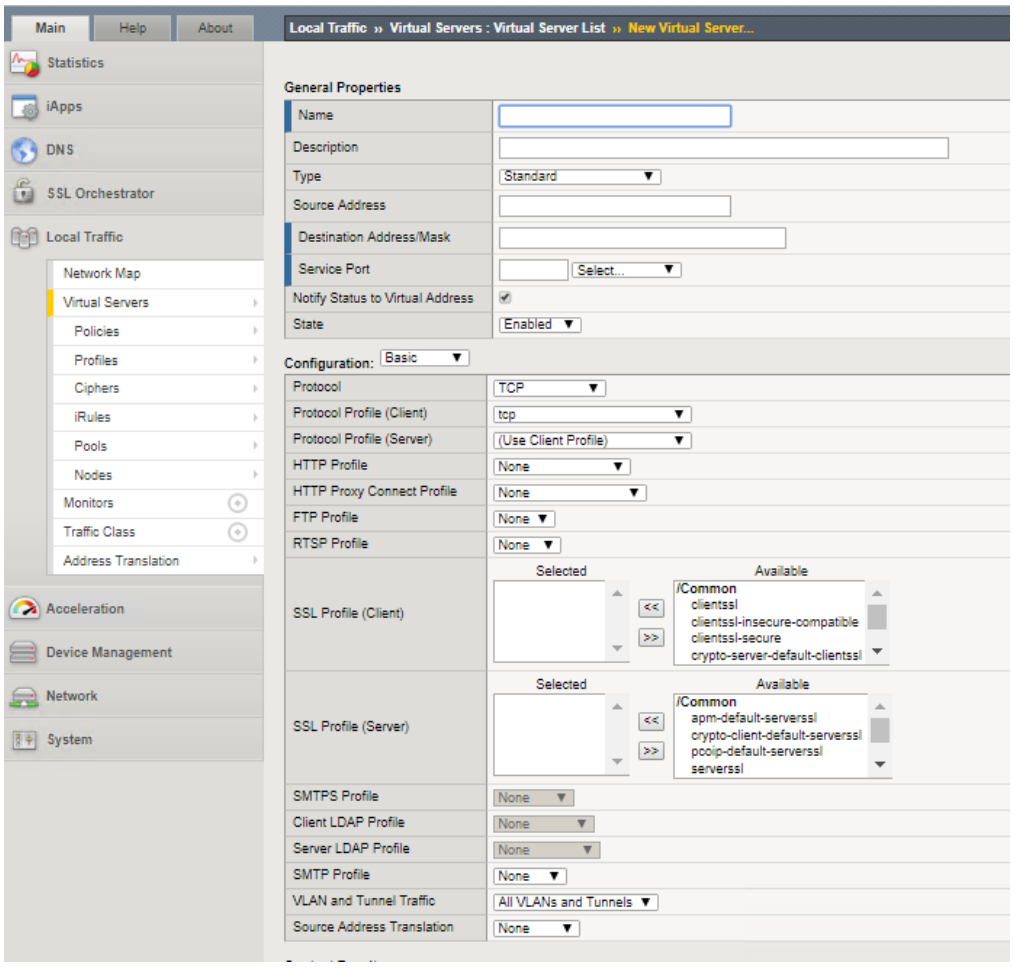

Figure 9. Creating a Virtual Server

7. Fill in the necessary fields including **Name**, **Source Address**, **Destination Address/Mask**, **Service Port** and **Source Address Translation**. Source Address Translation should be set to Auto Map enabled. An example configuration is shown in Figure 10.

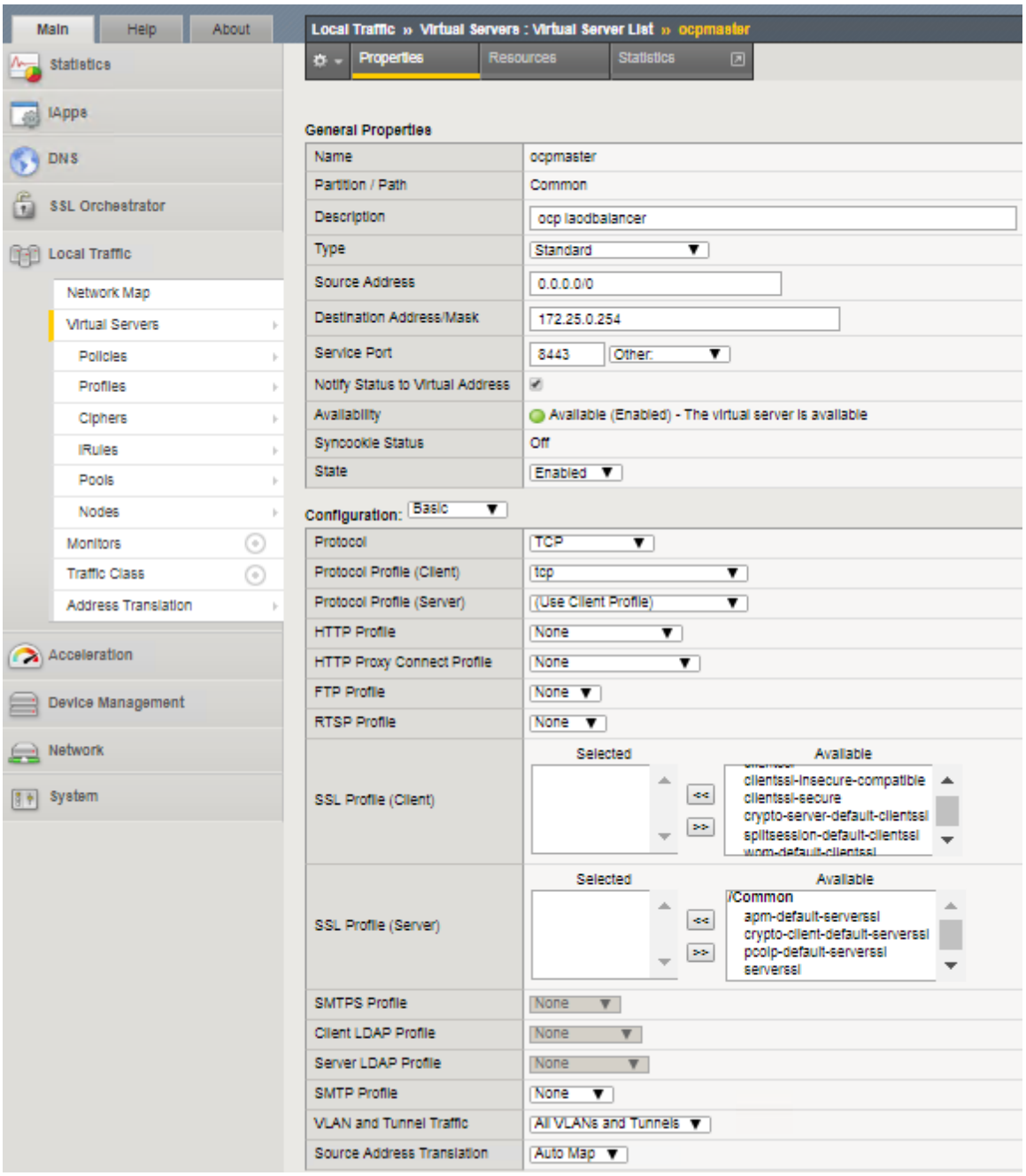

Figure 10. Defining Virtual Server properties

8. Navigate to **Local Traffic > Virtual Servers** and then select your **Virtual Server** and then **Resources**. When done, select the correct resource load balance pool. In the example shown in Figure 11, the default pool is ocp3-master.

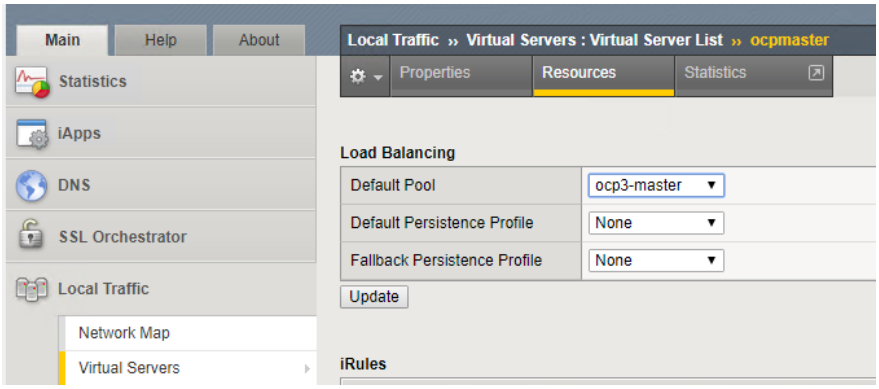

Figure 11. Selecting the appropriate Default Pool

9. The next step is to create VLANs. Navigate to **Network > VLANs > VLAN List** then click **Create**. Provide the values for the required fields as shown in Figure 12. Separate VLANs should exist for the management, data center and Red Hat OpenShift networks.

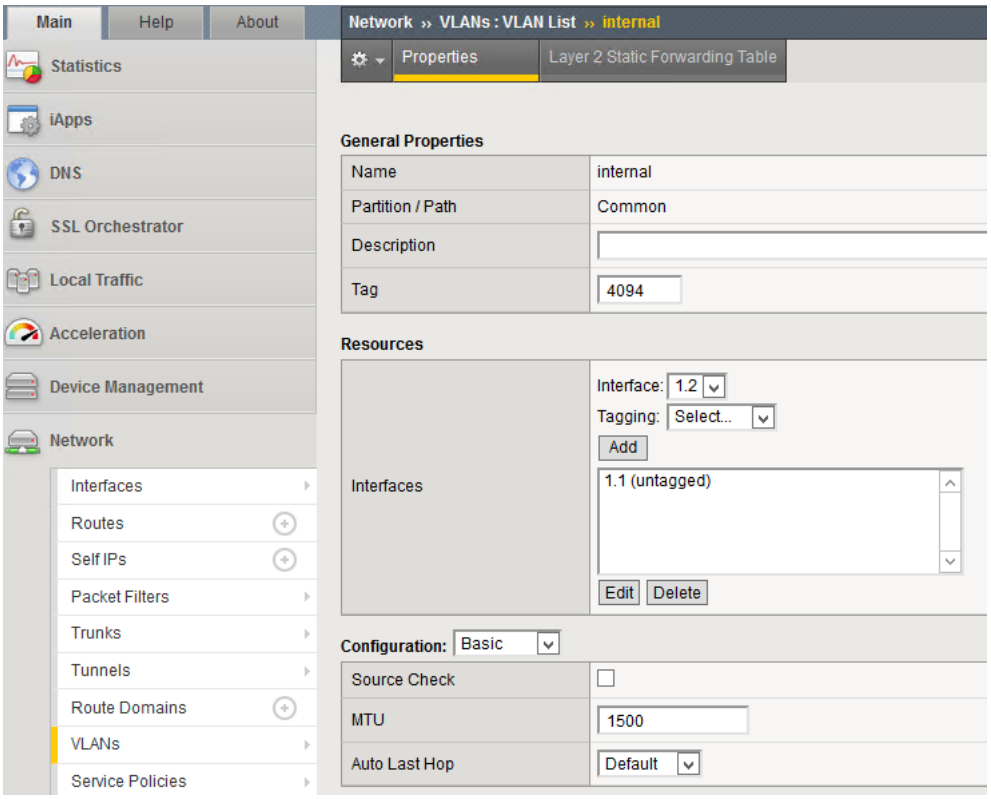

Figure 12. Defining VLANs.

10. Create a new Self IP using the newly created VLAN. A sample Self IP configuration is shown in Figure 13.

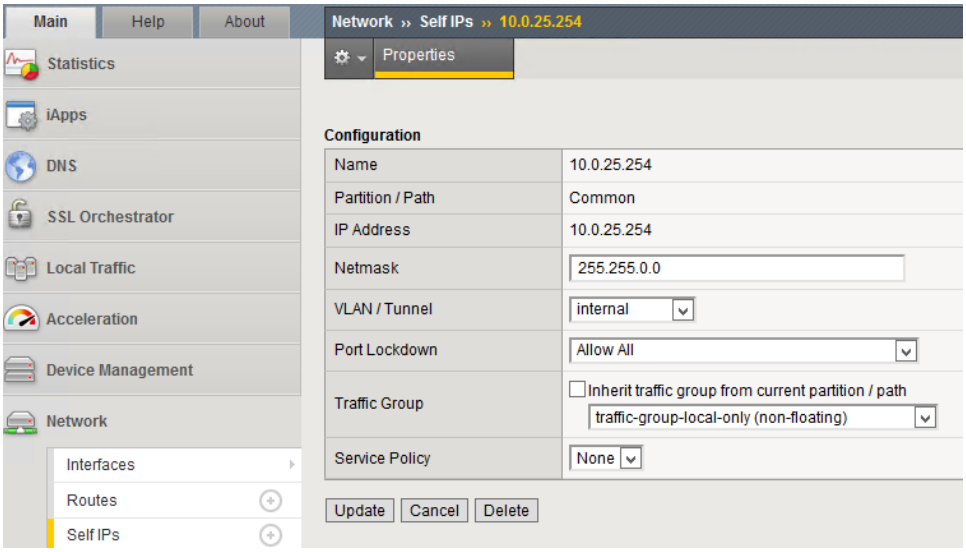

Figure 13. Creating a Self IP

Prepare Red Hat OpenShift Container Platform Virtual Machines

Utilize the playbooks provided in the appropriate Deployment Guide to create the virtual machines that will be used to install Red Hat OpenShift Container Platform. Deployment Guides and Ansible playbooks can be found at https://github.com/hewlettpackard/hpe-solutions-openshift. Follow the instructions in the guide to clone the repositories to your local Ansible Engine.

Edit the inventory file provided as part of the GitHub clone and provide the address of the BIG-IP address instead of the HA Proxy load balancer address. A sample inventory file is shown below.

```
[OSEv3:children]
masters
nodes
etcd
infra

[OSEv3:vars]

openshift_disable_check=disk_availability,memory_availability,package_availability,docker_image_availa
bility,docker_storage,package_version
ansible_ssh_user=root
ansible_become=true
openshift_release="3.11"
oreg_auth_user="{{ vault_rhsub_user }}"
oreg_auth_password="{{ vault_rhsub_pass }}"
openshift_deployment_type=openshift-enterprise
openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login': 'true', 'challenge': 'true',
'kind': 'HTPasswdPasswordIdentityProvider',}]
public_hosted_zone=labs.ocp41.loc
#load_balancer_hostname=
load_balancer_hostname=ocpmaster.tennet.local
#openshift_master_cluster_hostname=ocpmaster.tennet.local
```

```
openshift_master_cluster_hostname=<your value>
#openshift_master_cluster_public_hostname=ocpmaster.tennet.local
openshift_master_cluster_public_hostname=<your value>
#
#openshift_public_hostname=ocpmaster.tennet.local
openshift_public_hostname=<your value>
openshift_master_cluster_method=native
openshift_master_bootstrap_auto_approve=true
openshift_use_openshift_sdn=true
osm_cluster_network_cidr=12.0.0.0/8
osm_host_subnet_length=9
openshift_registry_selector='node-role.kubernetes.io/infra=true'
openshift_router_selector='node.role.kubernetes.io/infra=true'
openshift_enable_service_catalog=true
oreg_url='registry.access.redhat.com/openshift3/ose-${component}:${version}'

# host group for masters

[masters]
armaster01.tennet.local
armaster02.tennet.local

# host group for nodes
[nodes]
armaster01.tennet.local openshift_node_group_name='node-config-master'
armaster02.tennet.local openshift_node_group_name='node-config-master'
armaster01.tennet.local openshift_node_group_name='node-config-infra'
armaster02.tennet.local openshift_node_group_name='node-config-infra'
arworker02.tennet.local openshift_node_group_name='node-config-compute'

#Since we are providing a pre-configured LB VIP, no need for this group
#[lb]

# host group for infra

[infra]
armaster01.tennet.local
armaster02.tennet.local

# host group for etcd

[etcd]
armaster01.tennet.local
armaster02.tennet.local
```

Once the inventory file has been edited, use the commands found in the appropriate Deployment Guide to install Red Hat OpenShift Container Platform. This will involve installing the appropriate repository and running the prerequisites check play and the deploy cluster play found on the Ansible Engine under /usr/share/ansible/openshift-ansible/playbooks.

Once installation is complete ensure that you follow the guide and add a Red Hat OpenShift user and grant the user the cluster-admin role.

BIG-IP container integration

F5 Container Ingress Services enable the use of a BIG-IP device in a Red Hat OpenShift Container Platform implementation. Because Red Hat OpenShift has native Kubernetes integration, the F5 integration with Red Hat OpenShift utilizes the same controller as the F5 integration with Kubernetes. The F5 Container Ingress Services configure BIG-IP objects for applications in an OpenShift cluster, serving North-South traffic. This deployment guide provides step-by-step instructions for integrating an HA pair of BIG-IP devices into an OpenShift Cluster Network.

At a high level, the following tasks need to be completed in order to add BIG-IP devices to an OpenShift Container Platform cluster network.

1. Create a new Red Hat OpenShift HostSubnet
2. Create a VXLAN profile
3. Create a VXLAN tunnel
4. Create a Self IP in the VXLAN
5. Verify creation of the BIG-IP objects

Create a new Red Hat OpenShift HostSubnet

1. Define a HostSubnet manifest using valid YAML or JSON. You can upload the files individually using `oc create` commands. These will handle health monitor traffic. Also create one HostSubnet to pass client traffic. An `f5-kctlr-openshift-hostsubnet.yaml` file example appears below.

```
apiVersion: v1
kind: HostSubnet
metadata:
# provide a name
  name: f5-bigip-node01
  annotations:
    pod.network.openshift.io/fixed-vnid-host: "0"
    pod.network.openshift.io/assign-subnet: "true"
# provide a name for the BIG-IP device's host Node
#Provide a name
host: f5-bigip-node01
# Provide an IP address to serve as the BIG-IP VTEP in the OpenShift SDN
# self ip of BIG ip
hostIP: 10.0.25.254
```

2. After the appropriate yaml file has been created, upload the HostSubnet files to the OpenShift API server using the following command.

```
# oc create -f f5-kctlr-openshift-hostsubnet.yaml
```

3. Verify the creation of Host Subnets using the following command.

```
# oc get clusternetwork
NAME      CLUSTER NETWORKS   SERVICE NETWORK   PLUGIN NAME
default   12.0.0.0/8:9       172.30.0.0/16     redhat/openshift-ovs-subnet
```

4. Use **oc get** to retrieve information about the newly-created HostSubnets. Be sure to record the hostIP and subnet for each. These will be needed when setting up the VXLAN on your BIG-IP devices.

```
# oc get hostsubnet
NAME                      HOST                      HOST IP       SUBNET
armaster01.tennet.local   armaster01.tennet.local   10.0.25.121   12.1.0.0/23
armaster02.tennet.local   armaster02.tennet.local   10.0.25.122   12.0.0.0/23
arworker02.tennet.local   arworker02.tennet.local   10.0.25.125   12.2.0.0/23
f5-bigip-node01           f5-bigip-node01           10.0.25.254   12.3.0.0/23
```

Create a VXLAN profile

1. The installer should create a VXLAN profile that uses multi-cast flooding. From the BIG-IP web console navigate to **Network > Tunnels > Profiles > VXLAN** and then click **Create**. A window will open as in Figure 14.

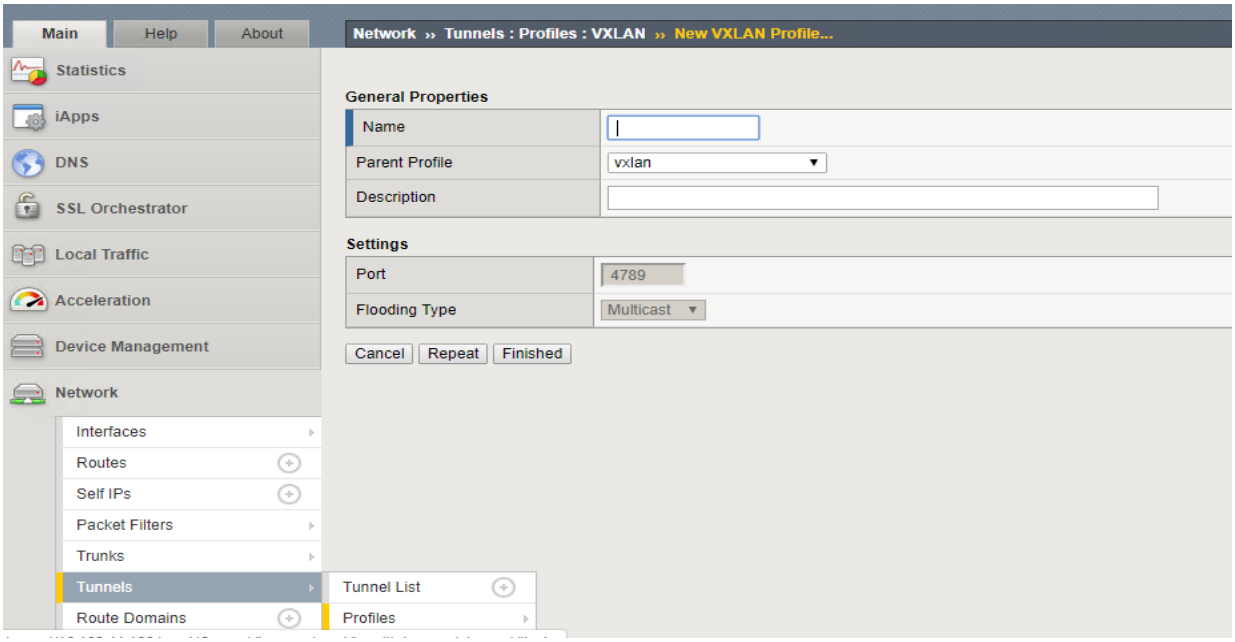

Figure 14. Creating VXLAN Profile

An example VXLAN Profile is shown in Figure 15.

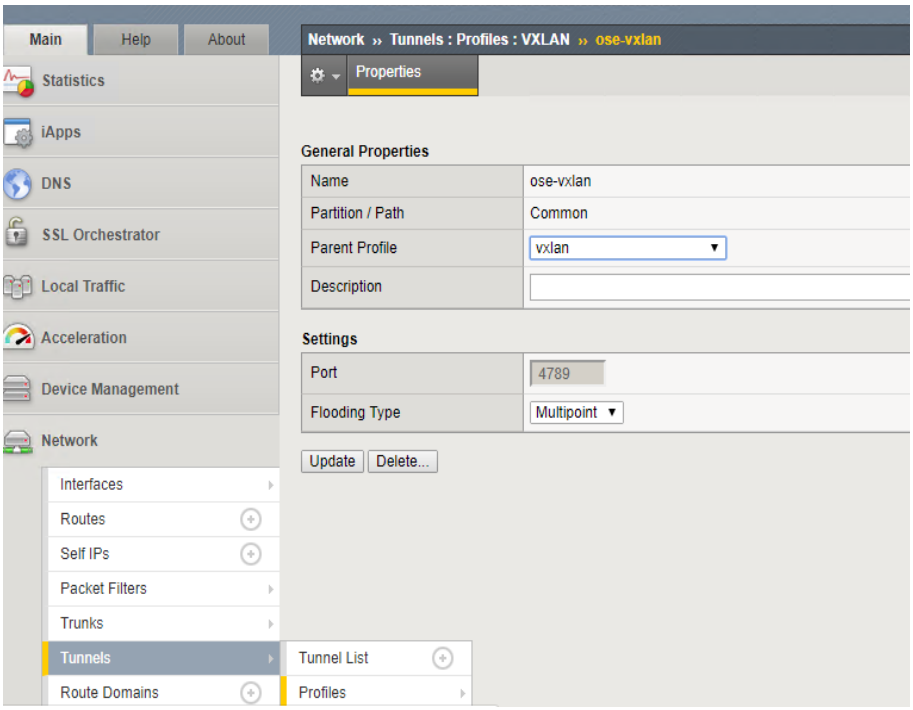

Figure 15. Example VXLAN profile

Create a VXLAN tunnel

1. To create a VXLAN tunnel, navigate to **Network > Tunnels > Tunnel List** and then click **Create**. Fill in the fields such as Local Address and VXLAN Profile that were created in step 2. Figure 16 shows the New Tunnel screen.

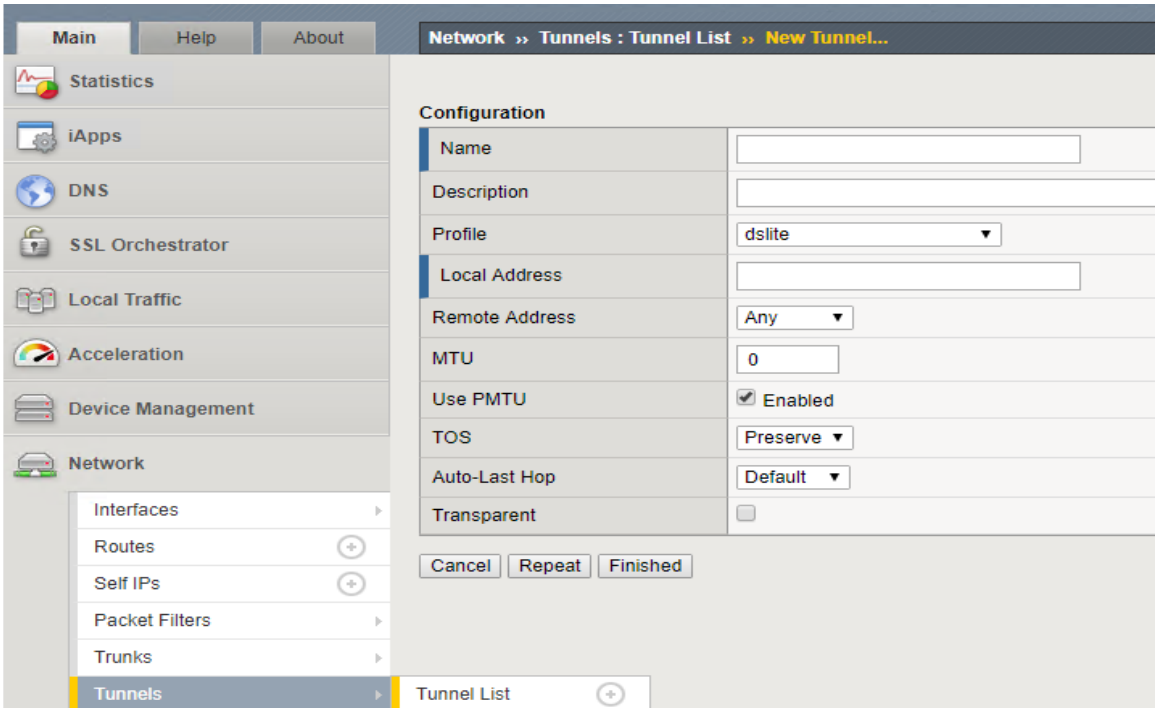

Figure 16. Creating a new VXLAN tunnel

An example VXLAN tunnel is shown in Figure 17.

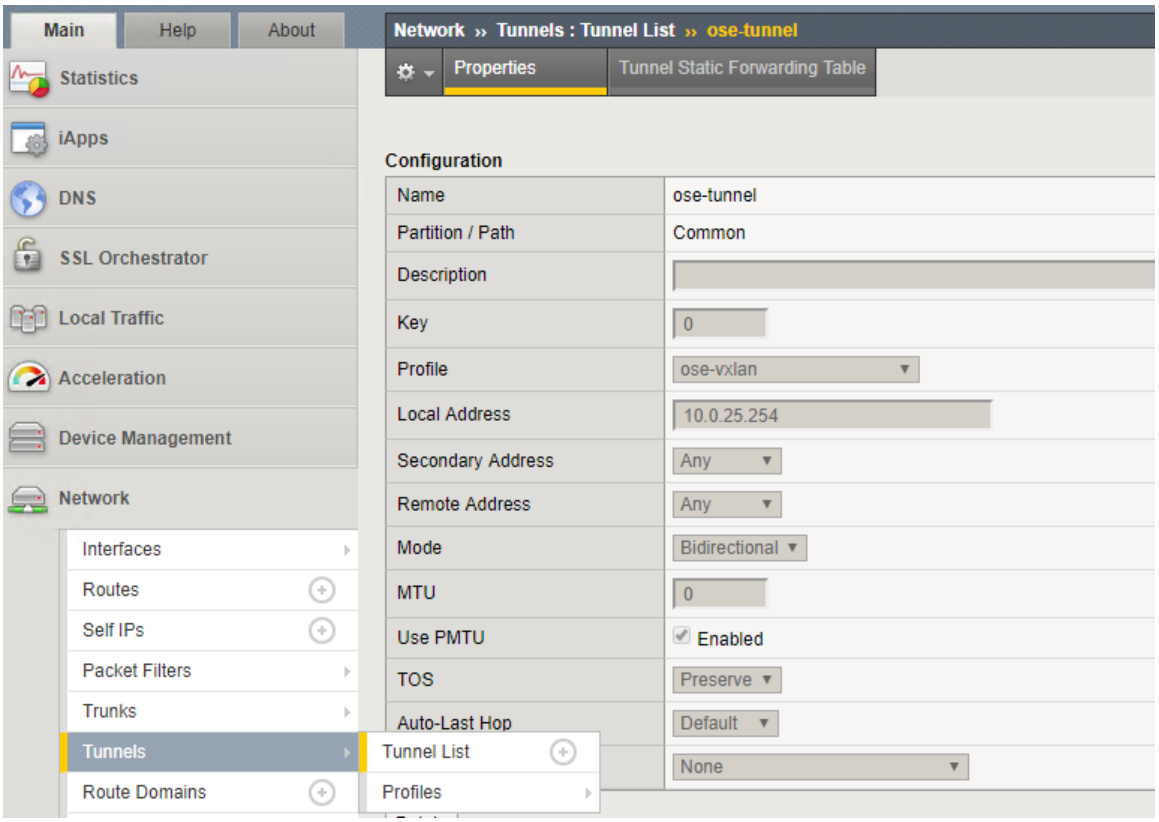

Figure 17. An example VXLAN tunnel

Create a Self IP in the VXLAN

The installer should create a Self IP address in the VXLAN. The subnet mask you assign to the Self IP must match the one that the Red Hat OpenShift SDN assigns to nodes. If you use the BIG-IP configuration utility to create a self IP, you may need to provide the full netmask instead of the CIDR notation. Be sure to specify a floating traffic group (for example, traffic-group-1). Otherwise, the Self IP will use the BIG-IP system's default.

```
# oc get hostsubnet
NAME                      HOST                      HOST IP       SUBNET
armaster01.tennet.local   armaster01.tennet.local   10.0.25.121   12.1.0.0/23
armaster02.tennet.local   armaster02.tennet.local   10.0.25.122   12.0.0.0/23
arworker02.tennet.local   arworker02.tennet.local   10.0.25.125   12.2.0.0/23
f5-bigip-node01           f5-bigip-node01           10.0.25.254   12.3.0.0/23
```

1. Create a Self IP in the same host subnet of the Red Hat OpenShift cluster. In the example above the subnet was 12.3.0.0.
2. From the BIG-IP web console navigate to **Network > Self IPs** and click **Create**. Provide all the necessary subnet information related to the host subnet created in the OCP SDN. A sample Self IP configuration is shown in Figure 18.

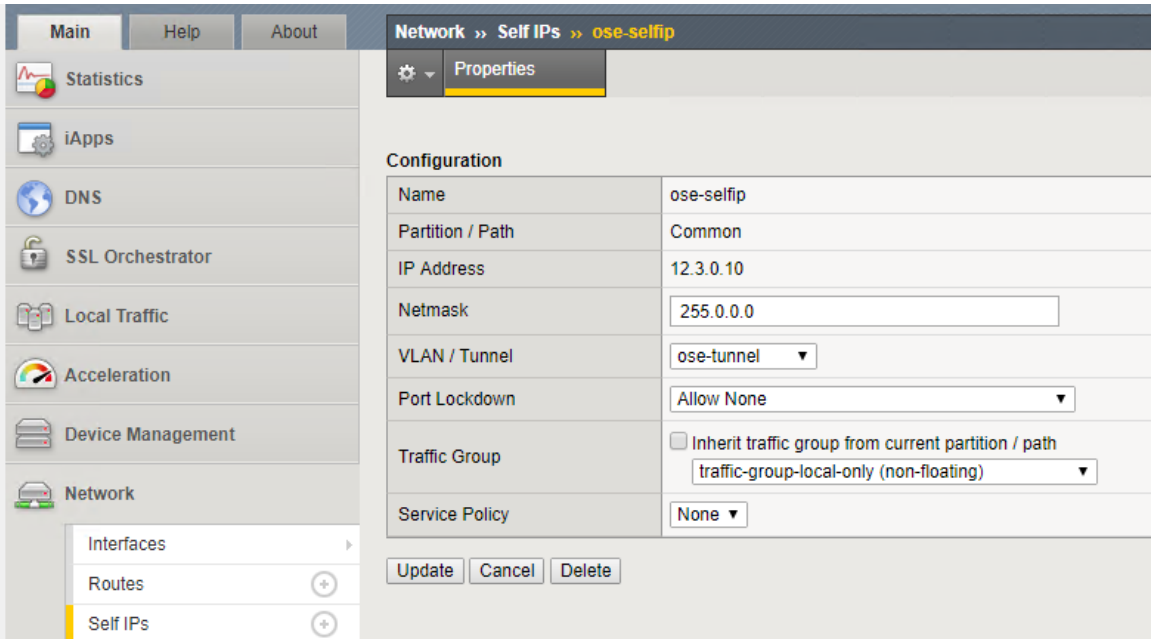

Figure 18. Creating a Self IP

After adding internal, external and ose-tunnel Self IPs the list of SelfIPs should resemble Figure 19.

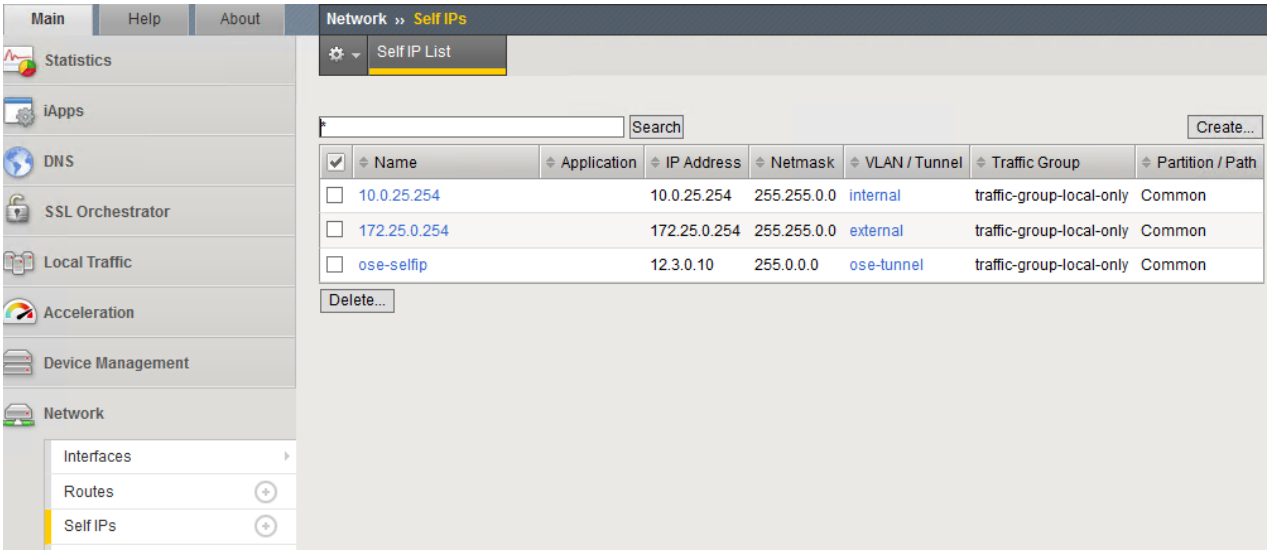

Figure 19. List of Self IPs

Create a new partition on BIG-IP system

1. From the BIG-IP web console, navigate to **System > Users > Partition List > New Partition** and then click **Create**. Provide the required information. An example is shown in Figure 20.

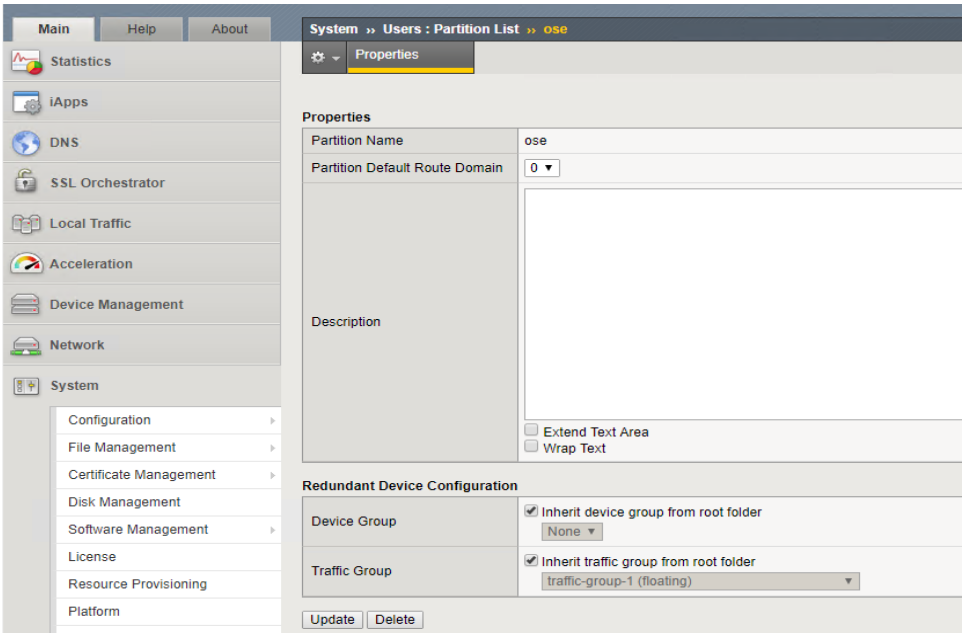

Figure 20. Creating a partition

Secure your BIG-IP credentials

1. Create a generic Secret containing your BIG-IP login information using the following command.

```
# kubectl create secret generic bigip-login --namespace kube-system --from-literal=username=admin --from-literal=password=admin
```

2. Verify the secret by running the following command.

```
# kubectl describe secret bigip-login
```

Set up RBAC Authentication

You can create RBAC resources in the project in which the F5 Container Ingress Services will run. Each Controller that manages a device in a cluster or active-standby pair can use the same Service Account, Cluster Role, and Cluster Role Binding.

1. Create a Service Account for the F5 Container Ingress Services by running the following command.

```
# oc create serviceaccount bigip-ctlr -n kube-system serviceaccount "bigip-ctlr"
```

2. Create a Cluster Role and Cluster Role Binding with the required permissions. Upload the Cluster Role and Cluster Role Binding to the API server. An example YAML file appears below.

```
# For use in OpenShift clusters
apiVersion: v1
kind: ClusterRole
metadata:
  annotations:
    authorization.openshift.io/system-only: "true"
  name: system:bigip-ctlr
```

```
  rules:
  - apiGroups: ["", "extensions"]
    resources: ["nodes", "services", "endpoints", "namespaces", "ingresses", "routes" ]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["", "extensions"]
    resources: ["configmaps", "events", "ingresses/status"]
    verbs: ["get", "list", "watch", "update", "create", "patch" ]
  - apiGroups: ["", "extensions"]
    resources: ["secrets"]
    resourceNames: ["<secret-containing-bigip-login>"]
    verbs: ["get", "list", "watch"]
  ---
  apiVersion: v1
  kind: ClusterRoleBinding
  metadata:
      name: bigip-ctlr-role
  userNames:
  - system:serviceaccount:kube-system:bigip-ctlr
  subjects:
  - kind: ServiceAccount
    name: bigip-ctlr
  roleRef:
    name: system:bigip-ctlr
```

3. To run the YAML file use the following command. For this example, the YAML file was named f5-kctlr-openshift-clusterrole.yaml.

```
# oc create -f f5-kctlr-openshift-clusterrole.yaml
```

Deploy the F5 Container Ingress Services

Follow the steps below to use a Deployment to deploy F5 Container Ingress Services into the cluster. You need to deploy a controller for both F5 BIG-IP nodes.

1. Provide a unique metadata.name for the services.
2. Provide a unique --bigip-url in the Deployment.
3. Use the same --bigip-partition in all Deployments.

The following example file is named f5-k8s-bigip-ctlr_openshift_basic.yaml.

```
  apiVersion: extensions/v1beta1
  kind: Deployment
  metadata:
    name: k8s-bigip-ctlr
  spec:
    replicas: 1
    template:
      metadata:
        name: k8s-bigip-ctlr
        labels:
          app: k8s-bigip-ctlr
      spec:
        # Name of the Service Account bound to a Cluster Role with the required
        # permissions
        serviceAccountName: bigip-ctlr
        containers:
```

```
      - name: k8s-bigip-ctlr
        # replace the version as needed
        image: "f5networks/k8s-bigip-ctlr"
        env:
          - name: BIGIP_USERNAME
            valueFrom:
              secretKeyRef:
                # Replace with the name of the Secret containing your login
                # credentials
                name: bigip-login
                key: username
          - name: BIGIP_PASSWORD
            valueFrom:
              secretKeyRef:
                # Replace with the name of the Secret containing your login
                # credentials
                name: bigip-login
                key: password
        command: ["/app/bin/k8s-bigip-ctlr"]
        args: [
          # See the k8s-bigip-ctlr documentation for information about
          # all config options
          # https://clouddocs.f5.com/products/connectors/k8s-bigip-ctlr/latest
          "--bigip-username=${BIGIP_USERNAME}",
          "--bigip-password=${BIGIP_PASSWORD}",
          "--bigip-url=10.0.25.254",
          "--bigip-partition=ose",
          "--pool-member-type=cluster",
          "--openshift-sdn-name=/Common/ose-tunnel",
          "--manage-routes=true",
          "--route-vserver-addr=172.25.0.25",
          "--route-label=f5router_label",
          "--insecure=true",
          ]
      imagePullSecrets:
        # Secret that gives access to a private Docker registry
        - name: f5-docker-images
        # Secret containing the BIG-IP system login credentials
        - name: bigip-login
```

4. To run the file, use the following command.

```
# oc create -f f5-k8s-bigip-ctlr_openshift_basic.yaml
```

5. Use the following commands to check the status.

```
# oc get deployments --namespace=kube-system
NAME             DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
k8s-bigip-ctlr   3         3         3            3           17h

# oc get pods --namespace=kube-system
NAME                                     READY     STATUS    RESTARTS   AGE
k8s-bigip-ctlr-79fc77bbdc-62d8m          1/1       Running   0          5h
k8s-bigip-ctlr-79fc77bbdc-nwk4l          1/1       Running   0          5h
k8s-bigip-ctlr-79fc77bbdc-nwzr4          1/1       Running   0          17h
```

```
master-api-armaster01.tennet.local          1/1       Running   2          2d
master-api-armaster02.tennet.local          1/1       Running   1          2d
master-controllers-armaster01.tennet.local  1/1       Running   2          2d
master-controllers-armaster02.tennet.local  1/1       Running   1          2d
master-etcd-armaster01.tennet.local         1/1       Running   2          3d
master-etcd-armaster02.tennet.local         1/1       Running   2          3d
```

6. Once the Red Hat OpenShift deployment has completed successfully, log into the console using the BIG-IP Domain name. In this case ocpmaster.tennet.lcoal, as shown in Figure 21.

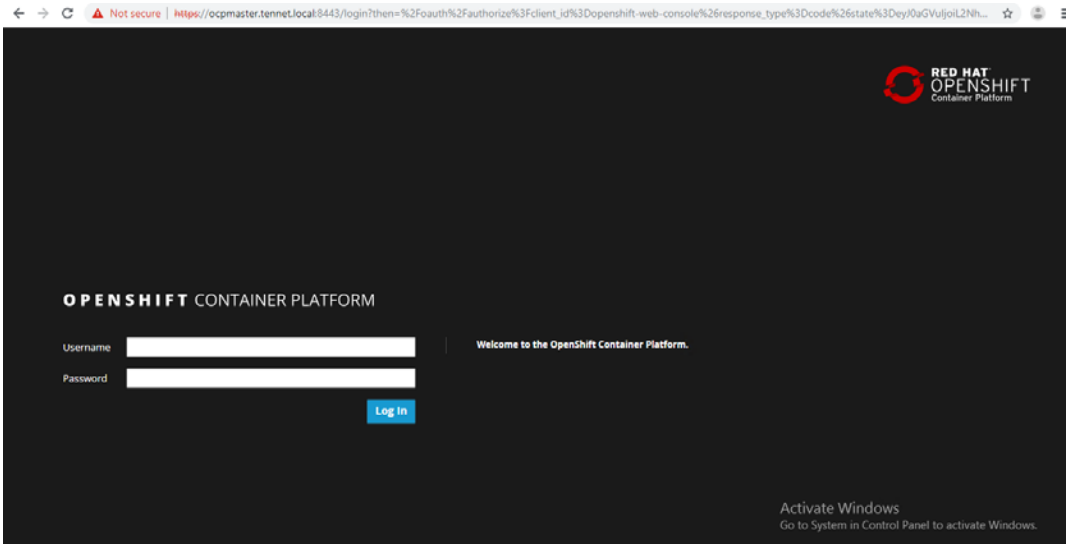

Figure 21. Red Hat OpenShift login console

The console is shown in Figure 22.

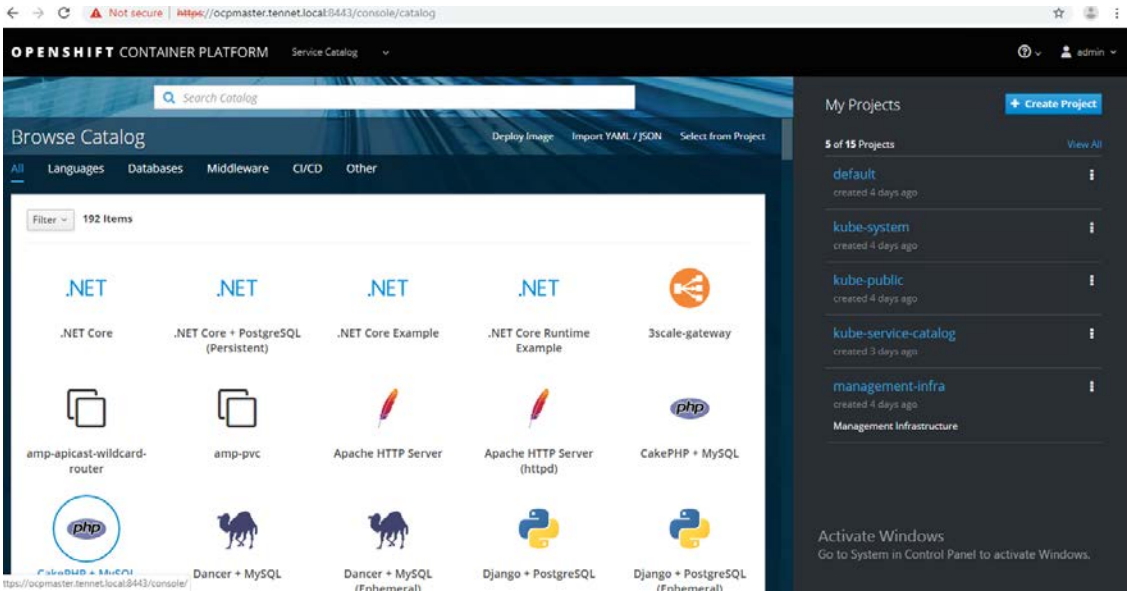

Figure 22. Red Hat OpenShift Container Platform main page post login

Change Tracker

Version	Release Date	Changes
1.0	10/21/2019	Initial release

Resources and additional links

HPE Installation Resources for Red Hat OpenShift Container Platform on HPE Synergy, https://github.com/hewlettpackard/hpe-solutions-openshift

HPE Reference Architectures, hpe.com/info/ra

F5 Container Ingress Services, https://www.f5.com/products/automation-and-orchestration/container-ingress-services

F5 container integration documentation, https://clouddocs.f5.com/containers/v2/openshift/kctlr-openshift-app-install.html

To help us improve our documents, please provide feedback at hpe.com/contact/feedback.

Share Now

Sign up for updates

OCP3844, October 2019, Version 1.0