W. Caban, *Architecting and Operating OpenShift Clusters*
https://doi.org/10.1007/978-1-4842-4985-7_11

11. OCP 4.1 UPI Mode Bare-Metal with PXE Boot Deployment

William Caban [1]

(1) Columbia, MD, USA

Some of the advanced networking optimizations discussed in Chapter 10 are intended to be used with bare-metal deployments of OpenShift. Furthermore, some of the capabilities are now included in OpenShift 4.1. [1]

This chapter provides supplementary information that goes into the details of an installation of OpenShift 4.1 in bare-metal deployment using the User Provisioned Infrastructure (UPI) mode that was discussed in Chapter 6 .

NOTE At the time of this writing, the UPI mode is still in beta, but it has been validated to work with bare-metal deployments.

UPI Mode

With the Installer Provisioned Infrastructure (IPI) mode, covered in Chapter 6 , the *openshift-installer* takes care of configuring ancillary services like internal and external load balancers, DNS records, and the provisioning of the base Operating System (OS); with the UPI mode, all those ancillary configurations need to be in place before starting the deployment.

The installation of OCP 4.1 with UPI mode varies based on the infrastructure target. For example, using UPI mode in a VMware environment, vs. an AWS environment, vs. Bare-Metal, has different steps. The core prerequisites are the same but the infrastructure-specific requirements will vary.

BARE-METAL WITH PXE BOOT EXAMPLE

This chapter covers UPI mode for Bare-Metal using PXE Boot for provisioning the OS during the installation. The diagram for the documented deployment is as shown in Figure 11-1.

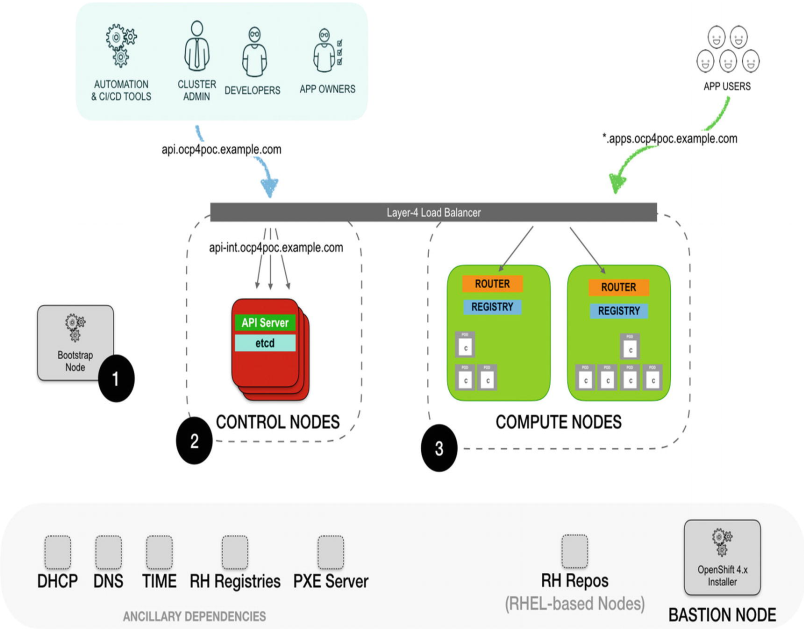

Figure 11-1 OCP 4.1 UPI standard deployment

The basic deployment for OCP 4.x is a high availability (HA) configuration with three *Master* or *Control Nodes* and at least two *Workers* or *Compute Nodes*. The Bootstrap Node is only used during the initial deployment of the *Master* or *Control Nodes*. See Table 11-1 for details on the reference environment.

Table 11-1 Reference Environment

Node Name	IP Address	Mac Address
bootstrap	192.168.1.10	02:01:01:01:01:01
master-0	192.168.1.11	02:00:00:00:01:01

Node Name	IP Address	Mac Address
master-1	192.168.1.12	02:00:00:00:01:02
master-2	192.168.1.13	02:00:00:00:01:03
worker-0	192.168.1.15	02:00:00:00:02:01
worker-1	192.168.1.16	02:00:00:00:02:02

NOTE The reference configuration uses `ocp4poc` as the cluster name and `example.com` as the base domain, hence the use of `ocp4poc.example.com` as the domain for the cluster.

UPI Bare-Metal with PXE Boot

There are two ways to install the Red Hat Enterprise Linux CoreOS (RHCOS). One is using an ISO image which then requires manual entry of parameters to load the Ignition configuration files, and the other option is using the PXE Boot install in which case all the Ignition parameters are passed using the PXE APPEND configuration fields.

PREREQUISITES

The deployment of OpenShift 4.1 using UPI mode with PXE Boot bare-metal has the following prerequisites:

- Designate a cluster name (i.e., cluster name = `ocp4poc`).
- Designate a base domain (i.e., base domain = `example.com`) for the subdomain dedicated to the cluster.
  - The cluster subdomain will be composed of <cluster-name>.<based-domain>.
    - That is, `ocp4poc.example.com`
- Fully resolvable FQDN forward and reverse DNS entries for all the Nodes (including the Bootstrap node).
  - Special *etcd* service entries are required.
  - Special *Kubernetes API* internal and external entries.
- Set up a Load Balancer in pass-through mode for Kubernetes API (tcp/6443), Machine Server Config (tcp/22623), and OpenShift Routers HTTP and HTTPS (tcp/80, tcp/443) .

> **NOTE** At the moment of this writing, when using UPI mode in bare-metal with PXE Boot, the Red Hat Enterprise Linux CoreOS (RHCOS) uses reverse DNS resolution for assigning the hostname to the Nodes.

DNS Configuration (Example)

Following the reference information from Table 11-1, the corresponding DNS configuration must include the entry layout in Table 11-2.

Table 11-2 Reference DNS Configuration

Role	FQDN	

Role	FQDN	
bootstrap	bootstrap.<cluster_name>.<base_domain>	192.168.1.10
master-0	master-0.<cluster_name>.<base_domain>	192.168.1.11
master-1	master-1.<cluster_name>.<base_domain>	192.168.1.12
master-2	master-2.<cluster_name>.<base_domain>	192.168.1.13
worker-0	worker-0.<cluster_name>.<base_domain>	192.168.1.15
worker-1	worker-1.<cluster_name>.<base_domain>	192.168.1.16

Role	FQDN	
Kubernetes API (tcp/6443)	api.<cluster_name>.<base_domain>	External Load Balancer for Master Nodes
	api-int.<cluster_name>.<base_domain>	Internal Load Balancer for Master Nodes
etcd	etcd-0.<cluster_name>.<base_domain>	192.168.1.11
	etcd-1.<cluster_name>.<base_domain>	192.168.1.12
	etcd-2.<cluster_name>.<base_domain>	192.168.1.13

Role	FQDN	
etcd SRV	etcd-server-ssl._tcp.<cluster_name>.<base_domain> For each Master Node, OpenShift requires a SRV DNS record for etcd server on that machine with priority 0, weight 10, and port 2380.	
Wildcard Subdomain for Apps	*.apps.<cluster_name>.<base_domain>	192.168.1.15, 192.168.1.16

The reference configurations in Listings 11-1 and 11-2 are for the *Bind DNS* server. When using other DNS servers, a similar configuration is required.

```
; /var/named/ocp4poc.example.com
$TTL 1D
@   IN SOA  bastion.ocp4poc.example.com.  root.ocp4poc.example.com. (
            2019052001  ; serial
            1D          ; refresh
            2H          ; retry
            1W          ; expiry
            2D )        ; minimum

@           IN NS       bastion.ocp4poc.example.com.
@           IN A        192.168.1.1
```

```
; Ancillary services
lb          IN A        192.168.1.200
lb-ex       IN A        10.10.10.10

; Bastion or Jumphost
bastion     IN A        192.168.1.1

; OCP Cluster
bootstrap   IN A        192.168.1.10

master-0    IN A        192.168.1.11
master-1    IN A        192.168.1.12
master-2    IN A        192.168.1.13

worker-0    IN A        192.168.1.15
worker-1    IN A        192.168.1.16

etcd-0      IN A        192.168.1.11
etcd-1      IN A        192.168.1.12
etcd-2      IN A        192.168.1.13

_etcd-server-ssl._tcp.ocp4poc.example.com.  IN
SRV  0   0   2380    etcd-0.ocp4poc.example.com.
                                            IN
SRV  0   0   2380    etcd-1.ocp4poc.example.com.
                                            IN
SRV  0   0   2380    etcd-2.ocp4poc.example.com.

api         IN CNAME    lb-ext   ; external LB
interface
api-int     IN CNAME    lb       ; internal LB
interface

apps        IN CNAME    lb-ext
*.apps      IN CNAME    lb-ext
```

Listing 11-1 Forward DNS Record

> **NOTE** The configuration of the etcd server records is required for the OpenShift installation. The api (external VIP pointing to the Control Nodes) and api-int (internal VIP pointing to the Control Nodes) records must exist pointing to the correct VIP.

```
;  /var/named/1.168.192.in-addr.arpa
$TTL 1h
$ORIGIN 1.168.192.IN-ADDR.ARPA.

@   1h  IN
SOA  bastion.ocp4poc.example.com.  root.ocp4poc.exampl
e.com. (
            2019052901  ; serial
            2H          ; refresh
            15          ; retry
            1W          ; expiry
            2H )        ; minimum

            IN NS       bastion.ocp4poc.example.com.

1       IN PTR  bastion.ocp4poc.example.com.

10      IN PTR  bootstrap.ocp4poc.example.com.

11      IN PTR  master-0.ocp4poc.example.com.
12      IN PTR  master-1.ocp4poc.example.com.
13      IN PTR  master-2.ocp4poc.example.com.

15      IN PTR  worker-0.ocp4poc.example.com.
16      IN PTR  worker-1.ocp4poc.example.com.

100     IN PTR  lb.ocp4poc.example.com.
```

Listing 11-2 Reverse DNS Record

Load Balancer Configuration (Examples)

The load balancer configuration is divided into external-facing configuration and cluster-facing configuration. The external-facing configuration should resolve to the external IP of the load balancer. The cluster-facing configuration should resolve to the internal IP of the load balancer. All the ports must be configured in pass-through mode. The ports required by OpenShift and that should be configured in the load balancer are listed in Table 11-3.

Table 11-3 Reference Load Balancer Configuration

Service VIP	Backend	F
Kubernetes API	bootstrap.ocp4poc.example.com:6443 master-0.ocp4poc.example.com:6443 master-1.ocp4poc.example.com:6443 master-2.ocp4poc.example.com:6443	6 T f E N s r a c b i p c
Machine Server	bootstrap.ocp4poc.example.com:22623 master-0.ocp4poc.example.com:22623 master-1.ocp4poc.example.com:22623 master-2.ocp4poc.example.com:22623	2 T f E N s r a c b i p c

Service VIP	Backend	P
Ingres HTTP	worker-0.ocp4poc.example.com:80 worker-1.ocp4poc.example.com:80	8
	worker-1.ocp4poc.example.com:80	
Ingress HTTPS	worker-0.ocp4poc.example.com:443 worker-1.ocp4poc.example.com:443	4
	api-int.<cluster_name>.<base_domain>	I L E f N

NGINX and HAProxy are Open Source projects commonly used as load balancers. A reference load balancer configuration using NGINX is presented in Listing 11-3.

```
# ngnix.conf
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log;
```

```
pid /run/nginx.pid;

events {
    worker_connections 1024;
}

# Pass-through
stream {
    upstream ocp4poc-k8s-api {
        # Kubernetes API
        server bootstrap.ocp4poc.example.com:6443;

        server master-0.ocp4poc.example.com:6443;
        server master-1.ocp4poc.example.com:6443;
        server master-2.ocp4poc.example.com:6443;
    }

    upstream ocp4poc-machine-config {
        # Machine-Config
        server bootstrap.ocp4poc.example.com:22623;

        server master-0.ocp4poc.example.com:22623;
        server master-1.ocp4poc.example.com:22623;
        server master-2.ocp4poc.example.com:22623;
    }

    server {
        listen 6443;
        proxy_pass ocp4poc-k8s-api;
    }

    server {
        listen 22623 ;
        proxy_pass ocp4poc-machine-config;

    }

    # Passthrough required for the routers
    upstream ocp4poc-http {
        # Worker Nodes running OCP Router
        server worker-0.ocp4poc.example.com:80;
        server worker-1.ocp4poc.example.com:80;
    }
```

```
    upstream ocp4poc-https {
        # Worker Nodes running OCP Router
        server worker-0.ocp4poc.example.com:443;
        server worker-1.ocp4poc.example.com:443;
    }
    server {
        listen 443;
        proxy_pass ocp4poc-http;
    }

    server {
        listen 80 ;
        proxy_pass ocp4poc-https;

    }
}
```

Listing 11-3 Load Balancer with NGINX (Example)

A reference load balancer configuration using HAProxy is presented in Listing 11-4.

```
# haproxy.cfg
defaults
    mode                    http
    log                     global
    option                  httplog
    option                  dontlognull
    option forwardfor       except 127.0.0.0/8
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          300s
    timeout server          300s
    timeout http-keep-alive 10s
    timeout check           10s
    maxconn                 20000

frontend openshift-api-server
    bind *:6443
```

```
    default_backend openshift-api-server
    mode tcp
    option tcplog
backend openshift-api-server
    balance source
    mode tcp
    server bootstrap 192.168.1.10:6443 check
    server master-0 192.168.1.11:6443 check
    server master-1 192.168.1.12:6443 check
    server master-2 192.168.1.13:6443 check

frontend machine-config-server
    bind *:22623
    default_backend machine-config-server
    mode tcp
    option tcplog

backend machine-config-server
    balance source
    mode tcp
    server bootstrap 192.168.1.10:22623 check
    server master-0 192.168.1.11:22623 check
    server master-1 192.168.1.12:22623 check
    server master-2 192.168.1.13:22623 check

frontend ingress-http
    bind *:8080
    default_backend ingress-http
    mode tcp
    option tcplog

backend ingress-http
    balance source
    mode tcp
    server worker-0 192.168.1.15:80 check
    server worker-1 192.168.1.15:80 check

frontend ingress-https
    bind *:8443
    default_backend ingress-https
    mode tcp
```

```
    option tcplog
backend ingress-https
    balance source
    mode tcp
    server worker-0 192.168.1.15:443 check
    server worker-1 192.168.1.15:443 check
```

Listing 11-4 Load Balancer with HAProxy (Example)

DHCP with PXE Boot Configuration (Example)

Listing 11-5 is a reference configuration of DHCP using *DNSmasq*, sending the PXE Boot server information to the Nodes.

```
# OCP4 PXE BOOT Lab
### dnsmasq configurations
# disable DNS /etc/dnsmasq.conf set port=0
#
no-dhcp-interface=eth0
interface=eth1

#domain=ocp4poc.example.com

#### DHCP (dnsmasq --help dhcp)
dhcp-range=eth1,192.168.1.10,192.168.1.200,24h
dhcp-option=option:netmask,255.255.255.0
dhcp-option=option:router,192.168.1.1
dhcp-option=option:dns-server,192.168.1.1
dhcp-option=option:ntp-server,204.11.201.10

# Bootstrap
dhcp-host=02:01:01:01:01:01,192.168.1.10

# master-0, master-1, master-2
dhcp-host=02:00:00:00:01:01,192.168.1.11
dhcp-host=02:00:00:00:01:02,192.168.1.12
dhcp-host=02:00:00:00:01:03,192.168.1.13

# worker-0, worker-1
dhcp-host=02:00:00:00:02:01,192.168.1.15
```

```
dhcp-host=02:00:00:00:02:01,192.168.1.16

#### PXE
enable-tftp
tftp-root=/var/lib/tftpboot,eth1
dhcp-boot=pxelinux.0
```

Listing 11-5 DHCP for PXE Boot with DNSmasq

PXE Boot Configuration (Example)

Listing 11-6 is a reference configuration of using DNSmasq as the PXE Boot server.

```
UI vesamenu.c32
DEFAULT LOCAL
PROMPT 0
TIMEOUT 200
ONTIMEOUT LOCAL

MENU TITLE PXE BOOT MENU

LABEL WORKER-BIOS
  MENU LABEL ^1 WORKER (BIOS)
  KERNEL rhcos/rhcos-kernel
  APPEND rd.neednet=1 initrd=rhcos/rhcos-initramfs.img
console=tty0 coreos.inst=yes
coreos.inst.install_dev=sda
coreos.inst.ignition_url=http://192.168.1.1:8000/worke
r.ign
coreos.inst.image_url=http://192.168.1.1:8000/metal/rh
cos-410.8.20190516.0-metal-bios.raw.gz ip=eth1:dhcp

LABEL MASTER-BIOS
  MENU LABEL ^2 MASTER (BIOS)
  KERNEL rhcos/rhcos-kernel
  APPEND rd.neednet=1 initrd=rhcos/rhcos-initramfs.img
console=tty0 coreos.inst=yes
coreos.inst.install_dev=sda
coreos.inst.ignition_url=http://192.168.1.1:8000/maste
r.ign
```

```
coreos.inst.image_url=http://192.168.1.1:8000/metal/rh
cos-410.8.20190516.0-metal-bios.raw.gz ip=eth1:dhcp

LABEL BOOTSTRAP-BIOS
  MENU LABEL ^3 BOOTSTRAP (BIOS)
  KERNEL rhcos/rhcos-kernel
  APPEND rd.neednet=1 initrd=rhcos/rhcos-initramfs.img
console=tty0 coreos.inst=yes
coreos.inst.install_dev=sda
coreos.inst.ignition_url=http://192.168.1.1:8000/boots
trap.ign
coreos.inst.image_url=http://192.168.1.1:8000/metal/rh
cos-410.8.20190516.0-metal-bios.raw.gz ip=eth1:dhcp

LABEL LOCAL
  MENU LABEL ^7 Boot from Local Disk
  MENU DEFAULT
  LOCALBOOT 0
LABEL RECOVERY1
  MENU LABEL ^8 Recovery (initqueue)
  KERNEL rhcos/rhcos-kernel
  APPEND rd.break=initqueue  rd.neednet=1
initrd=rhcos/rhcos-initramfs.img console=tty0
ip=eth1:dhcp

LABEL RECOVERY2
  MENU LABEL ^9 Recovery (pre-mount)
  KERNEL rhcos/rhcos-kernel
  APPEND rd.break=pre-mount  rd.neednet=1
initrd=rhcos/rhcos-initramfs.img console=tty0
ip=eth1:dhcp
```

Listing 11-6 DNSmasq as PXE Boot Server

Preparing the Installation

The bare-metal deployment of OpenShift 4.1 using UPI mode with PXE Boot requires special attention to the hardware configuration in use, especially the BIOS configuration and NIC interface configured for the PXE Boot.

NOTE The examples in this chapter use a Bastion Node in the same network as the Cluster Nodes, but this is not strictly necessary. They can be on different networks as long as the reachability exists.

CONSIDERATIONS WITH UPI MODE WITH PXE BOOT

At the time of this writing, there are several considerations to have when using UPI Mode with PXE Boot:

- When using a physical server with multiple NICs
  - The `PXE APPEND` command **must specify** the exact NIC to use during the PXE boot. For example, use a syntax similar to `ip=eth2:dhcp` and NOT a generic DHCP entry like `ip=dhcp`.
  - If the `PXE APPEND` uses the `ip=dhcp`, the DNS information from the **last** NIC to come up will be used as the entry for `/etc/resolv.conf`.
  - If the **last** NIC to come up has a self-assigned IP and does not receive a DNS, the resulting `/etc/resolv.conf` will be empty. When this happens, the *Node* will attempt to use the `localhost [::1]` as the DNS and the installation will fail. To work around this, during the installation
    - When possible, avoid having NICs with active link that are not receiving valid IPs.
    - Pass the `nameserver=<nameserver_ip>` with the `PXE APPEND` command.
  - When the server has many NICs, it is possible for the `NetworkManager-wait-online.service` to time out before the DHCP request over each NIC timeout. When this happens, a cascaded failure may be triggered. To avoid this situation, a recommended patch is to increase the timeout of this *NetworkManager* service and avoid the situation.

- At the time of this writing, using the `PXE APPEND` to disable IPv6 using the `ipv6.disable` is not supported.

- When customizing Ignition files to write custom files or configurations in the Node, the permissions must be specified in OCTAL mode (i.e., 384), NOT in DECIMAL mode (i.e., 600).

- If there is no valid reverse DNS resolution during the installation, the Masters (and all the Nodes) will register as `localhost.localdomain` into the *Kubernetes etcd*. When this happens, *Kubernetes* will fail to identify the existence of multiple masters and the installation process will fail.

DOWNLOADING RHCOS AND INSTALLATION BINARIES

The installation requires the download of the Red Hat Enterprise Linux CoreOS (RHCOS) corresponding to the 4.1 version, the OpenShift 4.1 client, and the OCP 4.1 openshift-installer. These are available from the corresponding mirror repositories:

- Obtain the latest RHCOS images from `https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.1/latest/`

- Obtain the latest OpenShift client and installer binaries from `https://mirror.openshift.com/pub/openshift-v4/clients/ocp/`

For the UPI mode using PXE Boot, the required images are as shown in Figure 11-2 (the specific subrelease and release will be different after GA).

```
images/
├── openshift-client-linux-4.1.0-rc.7.tar.gz
├── openshift-install-linux-4.1.0-rc.7.tar.gz
├── rhcos-410.8.20190516.0-installer-initramfs.img
├── rhcos-410.8.20190516.0-installer-kernel
├── rhcos-410.8.20190516.0-metal-bios.raw.gz
└── rhcos-410.8.20190516.0-metal-uefi.raw.gz
```

Figure 11-2 RHCOS and OCP 4.1 installation binaries (example)

PREPARING THE PXE BOOT IMAGES

Copy the RHCOS PXE Boot images to the PXE server similar to #1 on Figure 11-3. Copy the RHCOS Operating System Images to the web server to be used by the PXE installation similar to #2 on Figure 11-3.

```
# RHCOS PXE Boot Images 1
mkdir /var/lib/tftpboot/rhcos
cp ./images/rhcos-410.8.20190516.0-installer-initramfs.img /var/lib/tftpboot/rhcos/rhcos-initramfs.img
cp ./images/rhcos-410.8.20190516.0-installer-kernel /var/lib/tftpboot/rhcos/rhcos-kernel

# RHCOS OS Images 2
mkdir /usr/share/nginx/html/metal/
cp -f ./images/rhcos-410.8.20190516.0-metal-bios.raw.gz /usr/share/nginx/html/metal/
cp -f ./images/rhcos-410.8.20190516.0-metal-uefi.raw.gz /usr/share/nginx/html/metal/
```

Figure 11-3 Installing RHCOS PXE Boot and OS Images

Installation

At high level, the installation process consists of creating the *install-config.yaml* configuration, generating the Ignition files, and using those Ignition configurations to bootstrap the cluster.

Any customization required for the initial installation of the cluster must be done to those Ignition files. There are three initial Ignition files:

- **bootstrap.ign**: This Ignition file contains all the information the Bootstrap Node will use to render the cluster configuration and generate the *MachineConfig* configuration files for the Master Nodes.

- **master.ign**: This is the Ignition file the Master Nodes will use to install the RHCOS image into the bare-metal server. It also contains the information on how to obtain the Master Node configuration from the Bootstrap Node.

- **worker.ign**: This is the Ignition file the Worker Nodes will use to install the RHCOS image into the bare-metal server. It also contains the information on how to obtain the Worker Node configuration from the Master Nodes.

The discovery of the Kubernetes API to retrieve the state of the deployment process, the discovery of the API to retrieve the configuration for the Nodes, the discovery of the etcd database, and other access required by the Ignition process are highly dependent on the existence of the specific DNS entries discussed previously in this chapter.

CREATING THE CONFIGURATION

The OpenShift 4.1 installer UPI mode requires the creation of the install-config.yaml file which will be used to generate the Ignition files (Listing 11-7).

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: ocp4poc
networking:
  clusterNetworks:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

Listing 11-7 Sample install-config.yaml

The `pullSecret` must be obtained from `https://try.openshift.com`

The SSH key is the public SSH key from the key pair that is going to be used by the administration during the installation.

GENERATING THE IGNITION FILES

Create a folder for the installation, copy the `install-config.yaml` file into it, and proceed to generate the Ignition files, as shown in Figure 11-4.

```
# Creating installation folder
mkdir ocp4poc

# Copy installation configuration
cp ./install-config.yaml ocp4poc

# Generating Ignition files
./openshift-install create ignition-configs --dir=ocp4poc
```

Figure 11-4 Generating Ignition files

When using UPI PXE Boot with a system with multiple NIC, it is recommended to increase the timeout of the `NetworkManager-wait-online.service` (see Listing 11-8).

```
{
"systemd": {
   "units": [
    {
     "name": "NetworkManager-wait-online.service",
     "dropins": [{
       "name": "timeout.conf",
       "contents": "
[Service]\nExecStart=\nExecStart=/usr/bin/nm-online -s
-q --timeout=300"
     }]
    }
   ]
  }
}
```

Listing 11-8 Increase Network Manager timeout patch

By default, OCP 4.1 UPI only creates a local user in the Bootstrap Node. There is no local user in Master and Worker Nodes. To create a local user, follow Listing 11-9.

```
{
```

```
  "passwd": {
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [
          "ssh-rsa ...."
        ]
      }
    ]
  }
}
```

Listing 11-9 Adding local user

The patches from Listings 11-8 and 11-9 must be merged with the original Ignition file of the corresponding Node.

> **NOTE** At the moment of this writing, OpenShift does not provide a tool to edit the Ignition files and apply customization. Currently the administrator must rely on third-party tools to edit and merge the corresponding JSON files.

Copy the resulting Ignition files to the web server that will be used by the PXE Boot process—for example, `cp -f ./ocp4poc/*.ign /usr/share/nginx/html/`

BOOTSTRAP AND MASTER NODES

The first Node to be installed is the Bootstrap Node. When using the PXE configuration from Listing 11-6, the PXE Boot menu will be similar to Figure 11-5.

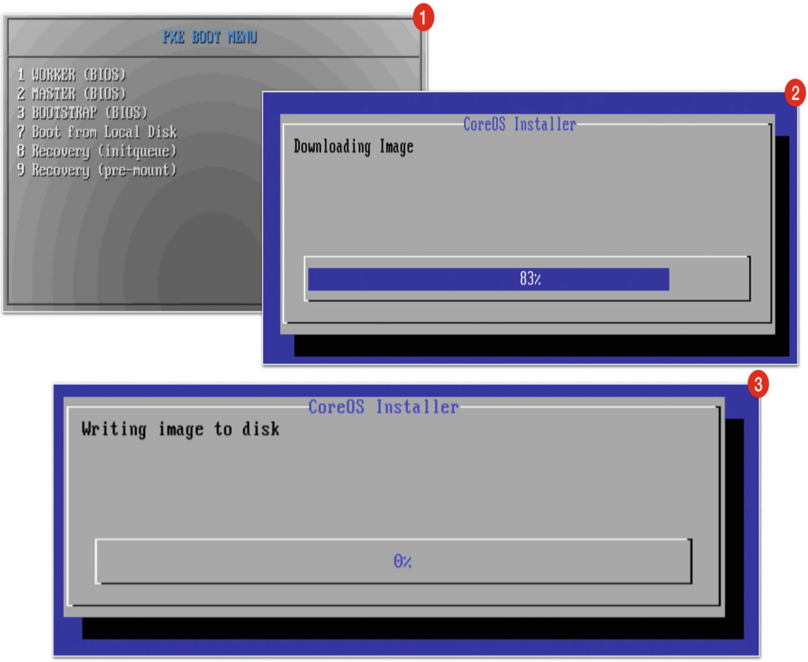

Figure 11-5 PXE Boot menu (example)

Select the Bootstrap from the menu and it will proceed with the installation of RHCOS.

Once the RHCOS installation of Bootstrap Node completes, it will reboot. After the Bootstrap is running, proceed to install RHCOS in the three Masters.

It is possible to use the `./openshift-install wait-for bootstrap-complete --dir=ocp4poc --log-level debug` command to have a high-level overview of the progress of the Bootstrap process of the Master Nodes. For more granular view of the progress, log in to the Bootstrap Node using the “core” user and the SSH key provided in the `install-config.yaml` (see #1 on Figure 11-6).

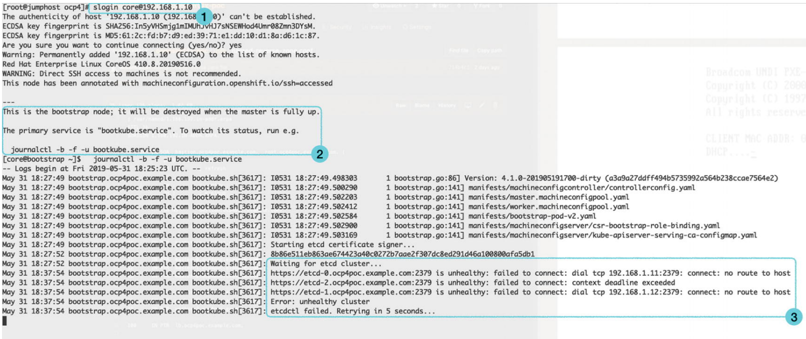

Figure 11-6 Log in to the Bootstrap Node

Once logged in, the Bootstrap Node executes the `journalctl -b -f -u bootkube.service` command to follow the detailed output messages about the progress of the process (see #2 and #3 in Figure 11-6).

After the installation of a Master Node completes, the Node will reboot in the RHCOS version used for the installation (see #1 on Figure 11-7). At this point, the Master requests the Machine Configuration rendered by the Cluster Version Operator running in the Bootstrap Node (see #2 on Figure 11-7). This will instruct the Node into downloading and applying the latest RHCOS (see #3 on Figure 11-7) and to start downloading and running the services corresponding to the Master Node.

Figure 11-7 Master Node Boot and Upgrade

Once the three Master Nodes are fully operational, the `openshift-install wait-for bootstrap` command will notify the Bootstrap Node has completed its job and it is time to shut down the Bootstrap Node (see #1 in Figure 11-8).

> **NOTE** At this point, it is safe to remove the Bootstrap Node from the Load Balancer configuration.

The log message from the Bootstrap Node will also indicate the Bootstrap process has been completed (see #2 on Figure 11-8).

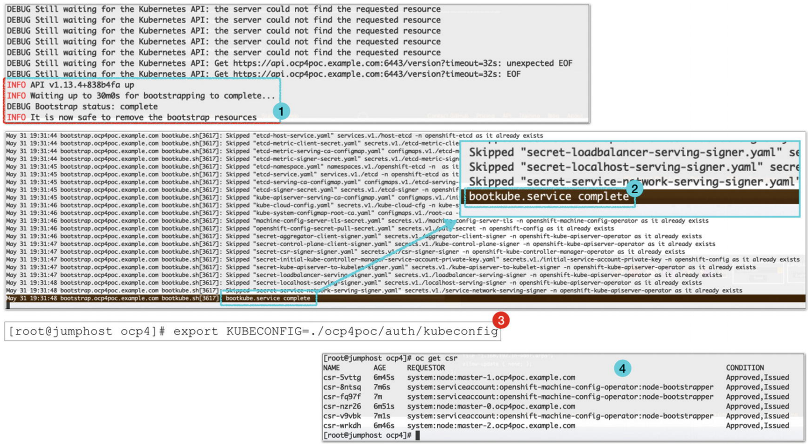

Figure 11-8 Bootstrap complete

During the bootstrap process, the *Bootstrap Node* takes care of signing the certificate requests from the *Masters* so they can become a single cluster (see #3 and #4 on Figure 11-8). After this point, adding workers or any other *Node* into the cluster requires for the cluster administrator to manually accept the *Certificate Signing Requests (CSR)* from the new *Nodes*.

After the *Bootstrap Node* has completed its purpose, the etcd and Kubernetes APIs are online, but the installation of the OpenShift Master Nodes is still in progress. To monitor this progress, use the command `./openshift-install wait-for install-complete --dir=ocp4poc --log-level debug`

WORKER NODES

Once the *Bootstrap Node* has been removed from the cluster, it is possible to install and onboard the *Worker Nodes*.

> **NOTE** Even when the installation of the *OpenShift Master Nodes* is still in progress, the successful completion of the OCP cluster installation requires at least two Worker Nodes to be online and be part of the cluster.

Boot and install RHCOS in the *Worker Nodes* using the same PXE Boot menu as before. This time, select the *Worker* option. The installation will be similar as with the Master Nodes. This time the Master Nodes are the ones providing the Machine Configuration to the Worker Nodes. For a Worker to start this process, it generates a Certificate Signing Request (CSR) for a node-bootstrapper Service Account which needs to be accepted by the cluster administrator (see #1 and #2 on Figure 11-9). Then it generates a system Node account CSR which needs to be approved for the Worker to join the cluster (see #3 and #4 on Figure 11-9).

Figure 11-9 OCP CSR signing

During this process, the Worker Nodes go over a RHCOS upgrade process and receive information on which containers to download and which services to bring online.

With all the Master and Worker Nodes online (see #1 on Figure 11-10), the installation will continue but will not complete to 100% until persistent storage is assigned to the Image Registry (see #3 on Figure 11-10).

Figure 11-10 Installation progress and Image Registry

> **NOTE** Persistent storage for the *Image Registry* should NOT be ephemeral in nature (like *emptyDir*) as images may be lost during a reboot of the Node hosting the registry. This type of ephemeral storage may only be used during testing or in nonproduction environments.

Once the installation is successfully completed, all the Cluster Operators should be shown as available (see Figure 11-11).

```
[root@jumphost ocp4]# oc get co
NAME                                  VERSION     AVAILABLE   PROGRESSING   DEGRADED   SINCE
authentication                        4.1.0-rc.7  True        False         False      40m
cloud-credential                      4.1.0-rc.7  True        False         False      63m
cluster-autoscaler                    4.1.0-rc.7  True        False         False      64m
console                               4.1.0-rc.7  True        False         False      43m
dns                                   4.1.0-rc.7  True        False         False      63m
image-registry                        4.1.0-rc.7  True        False         False      22s
ingress                               4.1.0-rc.7  True        False         False      47m
kube-apiserver                        4.1.0-rc.7  True        True          False      61m
kube-controller-manager               4.1.0-rc.7  True        False         False      61m
kube-scheduler                        4.1.0-rc.7  True        False         False      61m
machine-api                           4.1.0-rc.7  True        False         False      63m
machine-config                        4.1.0-rc.7  True        False         False      63m
marketplace                           4.1.0-rc.7  True        False         False      58m
monitoring                            4.1.0-rc.7  True        False         False      45m
network                               4.1.0-rc.7  True        False         False      64m
node-tuning                           4.1.0-rc.7  True        False         False      60m
openshift-apiserver                   4.1.0-rc.7  True        False         False      59m
openshift-controller-manager          4.1.0-rc.7  True        False         False      61m
openshift-samples                     4.1.0-rc.7  True        False         False      47m
operator-lifecycle-manager            4.1.0-rc.7  True        False         False      63m
operator-lifecycle-manager-catalog    4.1.0-rc.7  True        False         False      63m
service-ca                            4.1.0-rc.7  True        False         False      64m
service-catalog-apiserver             4.1.0-rc.7  True        False         False      60m
service-catalog-controller-manager    4.1.0-rc.7  True        False         False      60m
storage                               4.1.0-rc.7  True        False         False      58m
[root@jumphost ocp4]#
```

Figure 11-11 Cluster Operators running after successful installation

The OpenShift console (see Figure 11-12) for the new environment will be available at `https://console-openshift-console.apps.<cluster-name>.<base-domain>`.

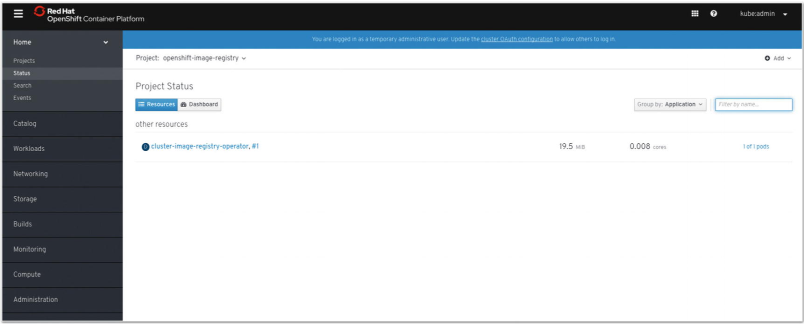

Figure 11-12 The OpenShift 4.1 Console

After the installation is completed, the system will have created the following Routes:

- `https://console-openshift-console.apps.ocp4poc.example.com` —default URL for the OpenShift console
- `https://oauth-openshift.apps.ocp4poc.example.com`
- `https://downloads-openshift-console.apps.ocp4poc.example.com`
- `https://alertmanager-main-openshift-monitoring.apps.ocp4poc.example.com`
- `https://grafana-openshift-monitoring.apps.ocp4poc.example.com`
- `https://prometheus-k8s-openshift-monitoring.apps.ocp4poc.example.com`

Summary

As seen in this chapter, the use of OpenShift User Provisioned Infrastructure (UPI) mode for Bare-Metal deployment may provide a way for organizations looking to retain control of the physical infrastructure while benefiting of a modern platform capable of auto-upgrade itself to the latest code.

The lecturer should be aware this is only one way to use the UPI mode. There are different ways in which UPI may be used to provision bare-metal or other types of infrastructures.

Footnotes

1 During the development of this book, Red Hat decided to keep OpenShift 4.0 as a Developer Preview release and instead did the release of OpenShift 4.1 as the first General Availability (GA) release of the 4.x major version.