**OpenShift 4: VMWare UPI installation (HA)**

Here I am going to follow a step by step guide to install Openshift 4.2 on VMWare Esxi. The same steps can be applied to provision high availability cluster on bare-metal. Actually, this is a high availability cluster deployment therefore we are using infra nodes to deploy router, logging and metrics. Otherwise by default these services runs on worker nodes.

**A short definition of:**  
UPI: User Provided Infrastructure (User manually prepare machine for bootstrap)  
IPI: Installer Provided Infrastructure (OCP Installer automatically terraforming machine for bootstrap)

**1. Pre-Requisites**

* A Valid Red Hat OpenShift [subscription](https://cloud.redhat.com/openshift)
* DNS Server (Bastion Node)
* HAProxy Server (Bastion Node)
* DHCP Server (Bastion Node)
* PXE/TFTP Server (Bastion Node)
* Note: if you are using bare-metal deployment then you need to replace the actual MAC addresses of individual machines (1 bootstrap, 3 masters, 3 infras, and 3 workers) in each configuration files (i.e in dhcpd.conf and in pxeboot.cfg folder)
* All are VMWare Hosts
  + 1 x Bastion Node  
    -> To host all the above-mentioned services  
    -> 4vCPU, 4GB RAM  
    -> 50 GB Disk  
    -> Centos 7 minimal  
    -> bastion.ocp.tmrnd.net
  + 1 x Bootstrap Node  
    -> 4 vCPU, 6GB RAM  
    -> 50 GB Disk  
    -> Empty Node  
    -> PXE Booted  
    -> bootstrap.ocp.tmrnd.net

-> MAC Address: ba:45:1f:2c:b1:4a

* + 3 x Master Node  
    -> 4 vCPU, 8GB RAM  
    -> 50 GB Disk  
    -> Empty Node  
    -> PXE Booted  
    -> master1.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:4b)

-> master2.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:4c)

-> master3.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:4d)

3 x Infra Node  
-> 4 vCPU, 8GB RAM  
-> 50 GB Disk  
-> Empty Node  
-> PXE Booted  
-> infra1.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:4e)

-> infra2.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:4f)

-> infra3.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:a4)

3 x Worker Node  
-> 4 vCPU, 8GB RAM  
-> 50 GB Disk  
-> Empty Node  
-> PXE Booted  
->worker1.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:b4)

->worker2.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:c4)

->worker3.ocp.tmrnd.net (mac: ba:45:1f:2c:b1:d4)

* SDN Subnets
  + Cluster Network: 10.128.0.0/14 (host prefix /23)
  + Service Network: 172.30.0.0/16
* FQDN
  + K8S API: api.ocp.tmrnd.net
  + Machine API: api-int.ocp.tmrnd.net
  + Wildcard subdomain: \*.apps.ocp.tmrnd.net

**NOTE**: Above sizing is purely for VMware deployment, for bare-metal deployment please consult recommended design from Red Hat OpenShift documentation.

**2. Configuration**

**2.1 Install and configure packages:**

[root@bastion ~]# dnf -y install bind bind-utils dhcp tftp-server syslinux httpd haproxy

[root@bastion ~]# firewall-cmd --add-service={dhcp,tftp,http,https,dns} --permanent

[root@bastion ~]# firewall-cmd --add-port={6443/tcp,22623/tcp,8080/tcp} --permanent

[root@bastion ~]# firewall-cmd --reload

**2.2 Configure DNS**

1. Now we need to create DNS hosted zones, create **/etc/named.conf**:

//

// named.conf

//

// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS

// server as a caching only nameserver (as a localhost DNS resolver only).

//

// See /usr/share/doc/bind\*/sample/ for example named configuration files.

//

// See the BIND Administrator's Reference Manual (ARM) for details about the

// configuration located in /usr/share/doc/bind-{version}/Bv9ARM.html

options {

listen-on port 53 { 192.168.50.9; };

// listen-on-v6 port 53 { ::1; };

directory "/var/named";

dump-file "/var/named/data/cache\_dump.db";

statistics-file "/var/named/data/named\_stats.txt";

memstatistics-file "/var/named/data/named\_mem\_stats.txt";

recursing-file "/var/named/data/named.recursing";

secroots-file "/var/named/data/named.secroots";

allow-query { any; };

/\*

- If you are building an AUTHORITATIVE DNS server, do NOT enable recursion.

- If you are building a RECURSIVE (caching) DNS server, you need to enable

recursion.

- If your recursive DNS server has a public IP address, you MUST enable access

control to limit queries to your legitimate users. Failing to do so will

cause your server to become part of large scale DNS amplification

attacks. Implementing BCP38 within your network would greatly

reduce such attack surface

\*/

recursion yes;

dnssec-enable yes;

dnssec-validation yes;

/\* Path to ISC DLV key \*/

bindkeys-file "/etc/named.root.key";

managed-keys-directory "/var/named/dynamic";

pid-file "/run/named/named.pid";

session-keyfile "/run/named/session.key";

};

logging {

channel default\_debug {

file "data/named.run";

severity dynamic;

};

};

zone "." IN {

type hint;

file "named.ca";

};

########################################

zone "ocp.tmrnd.net" IN {

type master;

file "zonefile.db";

allow-update { none; };

};

zone "50.168.192.in-addr.arpa" IN {

type master;

file "reverse.db";

allow-update { none; };

};

#######################################

include "/etc/named.rfc1912.zones";

include "/etc/named.root.key";

2. Create a zone database file in **/var/named/zonefile.db** with below content:

$TTL 1W

@ IN SOA masterdns.ocp.tmrnd.net. root.ocp.tmrnd.net (

2019080800 ; serial

3H ; refresh (3 hours)

30M ; retry (30 minutes)

2W ; expiry (2 weeks)

1W ) ; minimum (1 week)

@ IN NS masterdns.ocp.tmrnd.net.

@ IN PTR ocp.tmrnd.net.

masterdns IN A 192.168.50.9

; domain to use as reverse proxy in HA-proxy

; The api points to the IP of your load balancer

api IN A 192.168.50.9

api-int IN A 192.168.50.9

;

; The wildcard also points to the load balancer

\*.apps IN A 192.168.50.9

;

; Create entry for the bootstrap host

bootstrap IN A 192.168.50.2

;

; Create entries for the master hosts

master1 IN A 192.168.50.3

master2 IN A 192.168.50.4

master3 IN A 192.168.50.5

;

; Create entries for the worker hosts

worker1 IN A 192.168.50.11

worker2 IN A 192.168.50.12

worker3 IN A 192.168.50.13

;

; Create entries for the infra hosts

infra1 IN A 192.168.50.6

infra2 IN A 192.168.50.7

infra3 IN A 192.168.50.8

; The ETCd cluster lives on the masters...so point these to the IP of the masters

etcd-0 IN A 192.168.50.3

etcd-1 IN A 192.168.50.4

etcd-2 IN A 192.168.50.5

;

; The SRV records are IMPORTANT....make sure you get these right...note the trailing dot at the end...

\_etcd-server-ssl.\_tcp IN SRV 0 10 2380 etcd-0.ocp.tmrnd.net.

\_etcd-server-ssl.\_tcp IN SRV 0 10 2380 etcd-1.ocp.tmrnd.net.

\_etcd-server-ssl.\_tcp IN SRV 0 10 2380 etcd-2.ocp.tmrnd.net.

;

;EOF

3. Create a zone database file in **/var/named/reverse.db** with below content:

$TTL 1W

@ IN SOA masterdns.ocp.tmrnd.net. root (

2019080800 ; serial

3H ; refresh (3 hours)

30M ; retry (30 minutes)

2W ; expiry (2 weeks)

1W ) ; minimum (1 week)

IN NS masterdns.ocp.tmrnd.net.

@ IN NS masterdns.ocp.tmrnd.net.

@ IN PTR ocp.tmrnd.net

masterdns IN A 192.168.50.9

;

2 IN PTR bootstrap.ocp.tmrnd.net.

; syntax is "last octet" and the host must have fqdn with trailing dot

3 IN PTR master1.ocp.tmrnd.net.

4 IN PTR master2.ocp.tmrnd.net.

5 IN PTR master3.ocp.tmrnd.net.

;

6 IN PTR infra1.ocp.tmrnd.net.

7 IN PTR infra2.ocp.tmrnd.net.

8 IN PTR infra3.ocp.tmrnd.net.

;

11 IN PTR worker1.ocp.tmrnd.net.

12 IN PTR worker2.ocp.tmrnd.net.

13 IN PTR worker3.ocp.tmrnd.net.

;

;EOF

4. Restart named and test the DNS:

[root@bastion ~]# systemctl restart named

[root@bastion ~]# dig @localhost -t srv \_etcd-server-ssl.\_tcp.ocp.tmrnd.net

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @localhost -t srv \_etcd-server-ssl.\_tcp.ocp.tmrnd.net

; (2 servers found)

;; global options: +cmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58581

;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 1, ADDITIONAL: 5

;; OPT PSEUDOSECTION:

; EDNS: version: 0, flags:; udp: 4096

;; QUESTION SECTION:

;\_etcd-server-ssl.\_tcp.ocp.tmrnd.net. IN SRV

;; ANSWER SECTION:

\_etcd-server-ssl.\_tcp.ocp.tmrnd.net. 604800 IN SRV 0 10 2380 etcd-2.ocp.tmrnd.net.

\_etcd-server-ssl.\_tcp.ocp.tmrnd.net. 604800 IN SRV 0 10 2380 etcd-1.ocp.tmrnd.net.

\_etcd-server-ssl.\_tcp.ocp.tmrnd.net. 604800 IN SRV 0 10 2380 etcd-0.ocp.tmrnd.net.

;; AUTHORITY SECTION:

ocp.tmrnd.net. 604800 IN NS masterdns.ocp.tmrnd.net.

;; ADDITIONAL SECTION:

etcd-0.ocp.tmrnd.net. 604800 IN A 192.168.50.3

etcd-1.ocp.tmrnd.net. 604800 IN A 192.168.50.4

etcd-2.ocp.tmrnd.net. 604800 IN A 192.168.50.5

masterdns.ocp.tmrnd.net. 604800 IN A 192.168.50.9

;; Query time: 0 msec

;; SERVER: ::1#53(::1)

;; WHEN: Mon Oct 21 04:29:25 EDT 2019

;; MSG SIZE rcvd: 272

**2.3 Configuring DHCP and PXE**

1. Setup **/etc/dhcp/dhcpd.conf**with below content (update this as per your environment!):

#

# DHCP Server Configuration file.

# see /usr/share/doc/dhcp\*/dhcpd.conf.example

# see dhcpd.conf(5) man page

#

authoritative;

ddns-update-style interim;

default-lease-time 14400;

max-lease-time 14400;

option routers 192.168.50.1;

option broadcast-address 192.168.50.255;

option subnet-mask 255.255.255.0;

option domain-name-servers 192.168.50.9;

option domain-name "ocp.tmrnd.net";

default-lease-time -1;

max-lease-time -1;

subnet 192.168.50.0 netmask 255.255.255.0 {

pool {

range 192.168.50.2 192.168.50.8;

range 192.168.50.11 192.168.50.13;

# Static entries

host bootstrap { hardware ethernet ba:45:1f:2c:b1:4a; fixed-address 192.168.50.2;

option host-name "bootstrap.ocp.tmrnd.net";}

host master1 { hardware ethernet ba:45:1f:2c:b1:4b; fixed-address 192.168.50.3;

option host-name "master1.ocp.tmrnd.net";}

host master2 { hardware ethernet ba:45:1f:2c:b1:4c; fixed-address 192.168.50.4;

option host-name "master2.ocp.tmrnd.net";}

host master3 { hardware ethernet ba:45:1f:2c:b1:4d; fixed-address 192.168.50.5;

option host-name "master3.ocp.tmrnd.net";}

host infra1 {hardware ethernet ba:45:1f:2c:b1:4e; fixed-address 192.168.50.6;

option host-name "infra1.ocp.tmrnd.net"; }

host infra2 { hardware ethernet ba:45:1f:2c:b1:4f; fixed-address 192.168.50.7;

option host-name "infra2.ocp.tmrnd.net";}

host infra3 { hardware ethernet ba:45:1f:2c:b1:a4; fixed-address 192.168.50.8;

option host-name "infra3.ocp.tmrnd.net";}

host worker1 { hardware ethernet ba:45:1f:2c:b1:b4; fixed-address 192.168.50.11;

option host-name "worker1.ocp.tmrnd.net";}

host worker2 { hardware ethernet ba:45:1f:2c:b1:c4; fixed-address 192.168.50.12;

option host-name "worker2.ocp.tmrnd.net";}

host worker3 { hardware ethernet ba:45:1f:2c:b1:d4; fixed-address 192.168.50.13;

option host-name "worker3.ocp.tmrnd.net";}

# this will not give out addresses to hosts not listed above

deny unknown-clients;

# this is PXE specific

filename "pxelinux.0";

next-server 192.168.50.9;

}

}

1. Create a folder in /**var/lib/tftpboot/pxelinux.cfg**

The contents of the folder are files which are considered as PXE boot loaders for each host.

[root@bastion ~] mkdir /var/lib/tftpboot/pxelinux.cfg

[root@bastion ~] cd /var/lib/tftpboot/pxelinux.cfg/

[root@bastion pxelinux.cfg]# pwd

/var/lib/tftpboot/pxelinux.cfg

[root@bastion pxelinux.cfg]# ll

total 40

-rw-r--r--. 1 root root 563 Sep 26 09:19 01-ba-45-1f-2c-b1-4a // bootstrap

-rw-r--r--. 1 root root 554 Sep 26 09:20 01-ba-45-1f-2c-b1-4b //master1

-rw-r--r--. 1 root root 554 Sep 26 09:20 01-ba-45-1f-2c-b1-4c //master2

-rw-r--r--. 1 root root 554 Sep 26 09:21 01-ba-45-1f-2c-b1-4d //master3

-rw-r--r--. 1 root root 551 Sep 26 09:21 01-ba-45-1f-2c-b1-4e //infra1

-rw-r--r--. 1 root root 551 Sep 26 09:22 01-ba-45-1f-2c-b1-4f //infra2

-rw-r--r--. 1 root root 551 Sep 26 09:22 01-ba-45-1f-2c-b1-a4 //infra3

-rw-r--r--. 1 root root 550 Sep 27 13:47 01-ba-45-1f-2c-b1-b4 //worker1

-rw-r--r--. 1 root root 550 Sep 27 13:46 01-ba-45-1f-2c-b1-c4 //worker2

-rw-r--r--. 1 root root 550 Sep 27 14:42 01-ba-45-1f-2c-b1-d4 //worker3

3. Here is the content of bootstrap, one master, one infra and one worker file, the same contents are copied to each master and each infra in their respective mac address files.

[root@bastion pxelinux.cfg]# cat 01-ba-45-1f-2c-b1-4a

default menu.c32

prompt 0

timeout 50

menu title \*\*\*\* OpenShift 4 Bootstrap PXE Boot Menu \*\*\*\*

label Install CoreOS 4.2.0 Bootstrap Node

kernel /openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-kernel

append ip=dhcp rd.neednet=1 coreos.inst.install\_dev=sda console=tty0 console=ttyS0 coreos.inst=yes coreos.inst.image\_url=http://192.168.50.9:8080/openshift4/images/rhcos-4.2.0-x86\_64-metal-bios.raw.gz coreos.inst.ignition\_url=http://192.168.50.9:8080/openshift4/4.2.0/ignitions/bootstrap.ign initrd=/openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-initramfs.img

==================================================================================================

[root@bastion pxelinux.cfg]# cat 01-ba-45-1f-2c-b1-4b

default menu.c32

prompt 0

timeout 50

menu title \*\*\*\* OpenShift 4 Master PXE Boot Menu \*\*\*\*

label Install CoreOS 4.2.0 Master Node

kernel /openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-kernel

append ip=dhcp rd.neednet=1 coreos.inst.install\_dev=sda console=tty0 console=ttyS0 coreos.inst=yes coreos.inst.image\_url=http://192.168.50.9:8080/openshift4/images/rhcos-4.2.0-x86\_64-metal-bios.raw.gz coreos.inst.ignition\_url=http://192.168.50.9:8080/openshift4/4.2.0/ignitions/master.ign initrd=/openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-initramfs.img

=============================================================================================

[root@bastion pxelinux.cfg]# cat 01-ba-45-1f-2c-b1-4e

default menu.c32

prompt 0

timeout 50

menu title \*\*\*\* OpenShift 4 Infra PXE Boot Menu \*\*\*\*

label Install CoreOS 4.2.0 Infra Node

kernel /openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-kernel

append ip=dhcp rd.neednet=1 coreos.inst.install\_dev=sda console=tty0 console=ttyS0 coreos.inst=yes coreos.inst.image\_url=http://192.168.50.9:8080/openshift4/images/rhcos-4.2.0-x86\_64-metal-bios.raw.gz coreos.inst.ignition\_url=http://192.168.50.9:8080/openshift4/4.2.0/ignitions/infra.ign initrd=/openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-initramfs.img

===============================================================================================

[root@bastion pxelinux.cfg]# cat 01-ba-45-1f-2c-b1-b4

default menu.c32

prompt 0

timeout 50

menu title \*\* OpenShift 4 Worker PXE Boot Menu \*\*

label Install CoreOS 4.2.0 Worker Node

kernel /openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-kernel

append ip=dhcp rd.neednet=1 coreos.inst.install\_dev=sda console=tty0 console=ttyS0 coreos.inst=yes coreos.inst.image\_url=http://192.168.50.9:8080/openshift4/images/rhcos-4.2.0-x86\_64-metal-bios.raw.gz coreos.inst.ignition\_url=http://192.168.50.9:8080/openshift4/4.2.0/ignitions/worker.ign initrd=/openshift4/4.2.0/rhcos-4.2.0-x86\_64-installer-initramfs.img

3 . Copy syslinux for PXE boot

[root@bastion ~]# cp -rvf /usr/share/syslinux/\* /var/lib/tftpboot

4. Start TFTP

[root@bastion ~]# systemctl start tftp

**2.4 Configure webserver to host RHCOS image**

1. Switch httpd to listen from 80 to 8080 in **/etc/httpd/conf/httpd.conf** and restart httpd service:

Listen 8080

[root@bastion ~]# systemctl restart httpd

2. Create a new directory for hosting the kernel and initramfs for PXE boot:

[root@bastion 4.2.0]# mkdir -p /var/lib/tftpboot/openshift4/

[root@bastion 4.2.0]# mkdir -p /var/lib/tftpboot/openshift4/4.2.0

[root@bastion 4.2.0]# cd /var/lib/tftpboot/openshift4/4.2.0

[root@bastion 4.2.0]# wget https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.2/latest/rhcos-4.2.0-x86\_64-installer-kernel

[root@bastion 4..0]# wget https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.2/latest/rhcos-4.2.0-x86\_64-installer-initramfs.img

[root@bastion 4.2.0]# restorecon -RFv .

3. Now, we going to host the Red Hat CoreOS image:

[root@bastion openshift4]# mkdir -p /var/www/html/openshift4

[root@bastion openshift4]# pwd

/var/www/html/openshift4

[root@bastion openshift4]# mkdir -p /var/www/html/openshift4/images/

[root@bastion openshift4]# cd /var/www/html/openshift4/images/

[root@bastion images]# wget https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.2/latest/rhcos-4.2.0-x86\_64-metal-bios.raw.gz

[root@bastion images]# restorecon -RFv .

**2.5 Configure HAProxy as LB**

1. A load balance required for balancing masters and ingress router (in our case infra will play the role of ingress routers), create **/etc/haproxy/haproxy.conf**:

#---------------------------------------------------------------------

# Example configuration for a possible web application. See the

# full configuration options online.

#

# http://haproxy.1wt.eu/download/1.4/doc/configuration.txt

#

#---------------------------------------------------------------------

#---------------------------------------------------------------------

# Global settings

#---------------------------------------------------------------------

global

# to have these messages end up in /var/log/haproxy.log you will

# need to:

#

# 1) configure syslog to accept network log events. This is done

# by adding the '-r' option to the SYSLOGD\_OPTIONS in

# /etc/sysconfig/syslog

#

# 2) configure local2 events to go to the /var/log/haproxy.log

# file. A line like the following can be added to

# /etc/sysconfig/syslog

#

# local2.\* /var/log/haproxy.log

#

log 127.0.0.1 local2

chroot /var/lib/haproxy

pidfile /var/run/haproxy.pid

maxconn 4000

user haproxy

group haproxy

daemon

# turn on stats unix socket

stats socket /var/lib/haproxy/stats

#---------------------------------------------------------------------

# common defaults that all the 'listen' and 'backend' sections will

# use if not designated in their block

#---------------------------------------------------------------------

defaults

mode http

log global

option httplog

option dontlognull

option http-server-close

option forwardfor except 127.0.0.0/8

option redispatch

retries 3

timeout http-request 10s

timeout queue 1m

timeout connect 10s

timeout client 1m

timeout server 1m

timeout http-keep-alive 10s

timeout check 10s

maxconn 3000

#---------------------------------------------------------------------

# main frontend which proxys to the backends

#---------------------------------------------------------------------

frontend main \*:5000

acl url\_static path\_beg -i /static /images /javascript /stylesheets

acl url\_static path\_end -i .jpg .gif .png .css .js

use\_backend static if url\_static

default\_backend app

#---------------------------------------------------------------------

# static backend for serving up images, stylesheets and such

#---------------------------------------------------------------------

backend static

balance roundrobin

server static 127.0.0.1:4331 check

#---------------------------------------------------------------------

# round robin balancing between the various backends

#---------------------------------------------------------------------

backend app

balance roundrobin

server app1 127.0.0.1:5001 check

server app2 127.0.0.1:5002 check

server app3 127.0.0.1:5003 check

server app4 127.0.0.1:5004 check

############# FOR OCP 4 ####################

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 192.168.50.2:6443 check

server master1 192.168.50.3:6443 check

server master2 192.168.50.4:6443 check

server master3 192.168.50.5:6443 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 192.168.50.2:22623 check

server master1 192.168.50.3:22623 check

server master2 192.168.50.4:22623 check

server master3 192.168.50.5:22623 check

frontend ingress-http

bind \*:80

default\_backend ingress-http

mode tcp

option tcplog

backend ingress-http

balance source

mode tcp

server infra1 192.168.50.6:80 check

server infra2 192.168.50.7:80 check

server infra3 192.168.50.8:80 check

frontend ingress-https

bind \*:443

default\_backend ingress-https

mode tcp

option tcplog

backend ingress-https

balance source

mode tcp

server infra1 192.168.50.6:443 check

server infra2 192.168.50.7:443 check

server infra3 192.168.50.8:443 check

## Added for to browse HAProxy web console)

listen stats

bind :32700

stats enable

stats uri /

stats hide-version

stats auth admin:pass

2. Allow haproxy to bind to custom port with SELinux: (it is required if your SELinux is enabled)

[root@bastion ~]# semanage port -a 22623 -t http\_port\_t -p tcp

[root@bastion ~]# semanage port -a 6443 -t http\_port\_t -p tcp

[root@bastion ~]# semanage port -a 32700 -t http\_port\_t -p tcp

[root@bastion ~]# semanage port -l | grep -w http\_port\_t

http\_port\_t tcp 22623, 32700, 6443, 80, 81, 443, 488, 8008, 8009, 8443, 9000

**2.6 Configure OpenShift Installer and CLI binary**

1. To start creating ignition files and managing cluster, download installer and client:

[root@bastion ~]# mkdir ocp4/

[root@bastion ~]# cd ocp4/

[root@bastion ocp4]# wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/latest/openshift-install-linux-4.2.0.tar.gz

[root@bastion ~]# wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/latest/openshift-client-linux-4.2.0.tar.gz

[root@bastion ocp4]# tar -xvf openshift-client-linux-4.2.0.tar.gz

[root@bastion ocp4]# tar -xvf openshift-install-linux-4.2.0.tar.gz

[root@bastion ocp4]# cp -v {oc,kubectl,openshift-install} /usr/bin/

'oc' -> '/usr/bin/oc'

'kubectl' -> '/usr/bin/kubectl'

'openshift-install' -> '/usr/bin/openshift-install'

2. Create a sshkey pair to use to access CoreOS node later on:

[root@bastion ocp4]# ssh-keygen -t rsa

3. Create installation working directory and start to prepare the ignition file:

[root@bastion ocp4]# cat install-config-base.yaml

apiVersion: v1

baseDomain: tmrnd.net

compute:

- hyperthreading: Enabled

name: worker

replicas: 0

controlPlane:

hyperthreading: Enabled

name: master

replicas: 3

metadata:

name: ocp

networking:

clusterNetworks:

- cidr: 10.10.0.0/16

hostPrefix: 23

networkType: OpenShiftSDN

serviceNetwork:

- 172.16.0.0/16

platform:

none: {}

pullSecret: 'GET FROM try.redhat.com'

sshKey: 'SSH PUBLIC KEY from #cat /root/.ssh/id\_rsa.pub'

4. Now copy **install-config-base.yaml**to **install-config.yaml**. (install-config.yaml will be consumed by installer, hence copied from base file instead)

[root@bastion ocp4]# mkdir -p ign

[root@bastion ocp4]# cd ign

[root@bastion ign]# cp ../install-config-base.yaml install-config.yaml

[root@bastion ign]# openshift-install create ignition-configs

WARNING There are no compute nodes specified. The cluster will not fully initialize without compute nodes.

INFO Consuming "Install Config" from target directory

[root@bastion ign]# ll

total 288

drwxr-xr-x. 2 root root 50 Jul 1 16:46 auth

-rw-r--r--. 1 root root 276768 Jul 1 16:46 bootstrap.ign

-rw-r--r--. 1 root root 3545 Jul 1 16:42 install-config-base.yaml

-rw-r--r--. 1 root root 1824 Jul 1 16:46 master.ign

-rw-r--r--. 1 root root 96 Jul 1 16:46 metadata.json

-rw-r--r--. 1 root root 1824 Jul 1 16:46 worker.ign

[root@bastion ign]# cp worker.ign infra.ign

5. We need to copy ignition file to http hosted directory:

[root@bastion ign]# mkdir -p /var/www/html/openshift4/4.2.0

[root@bastion ign]# mkdir -p /var/www/html/openshift4/4.2.0/ignitions

[root@bastion ign]# cp -v \*.ign /var/www/html/openshift4/4.2.0/ignitions/

'bootstrap.ign' -> '/var/www/html/openshift4/4.2.0/ignitions/bootstrap.ign'

'master.ign' -> '/var/www/html/openshift4/4.2.0/ignitions/master.ign'

'infra.ign' -> '/var/www/html/openshift4/4.2.0/ignitions/infra.ign'

'worker.ign' -> '/var/www/html/openshift4/4.2.0/ignitions/worker.ign'

[root@bastion ign]# restorecon -RFv /var/www/html/

[root@bastion ocp4]#

6. At this moment we are done with the configuration, enable and start all services:

[root@bastion ~]# systemctl enable --now haproxy.service dhcpd httpd tftp named

[root@bastion ~]# systemctl restart --now haproxy.service dhcpd httpd tftp named

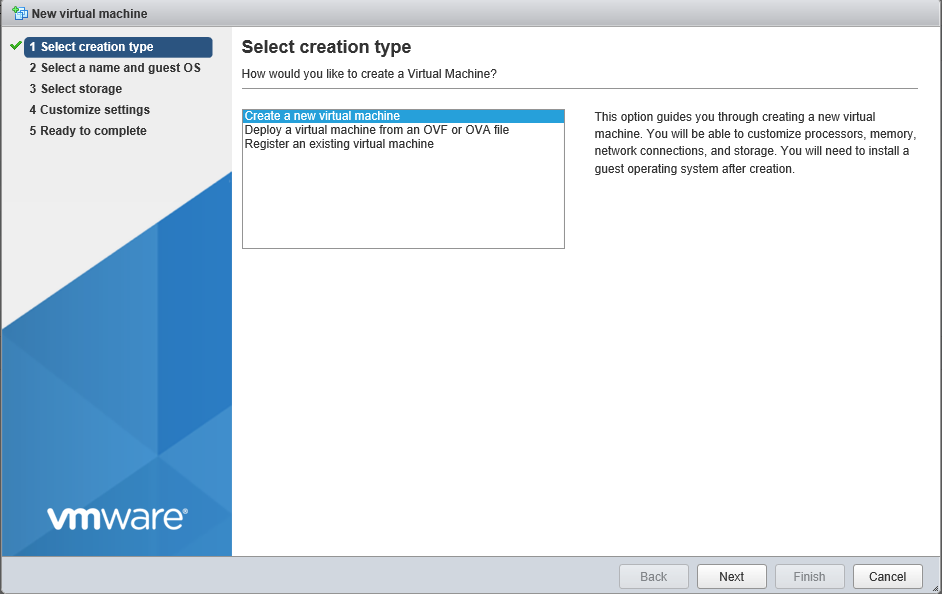
[root@bastion ~]# systemctl status --now haproxy.service dhcpd httpd tftp named

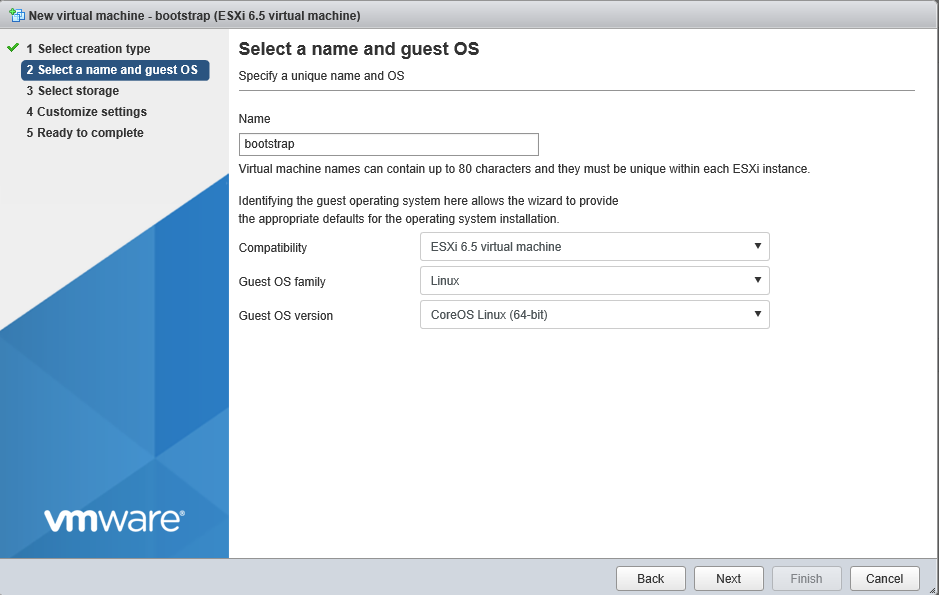
7. We are now should be able to boot up all the nodes and let bootstrap installing the initial cluster!

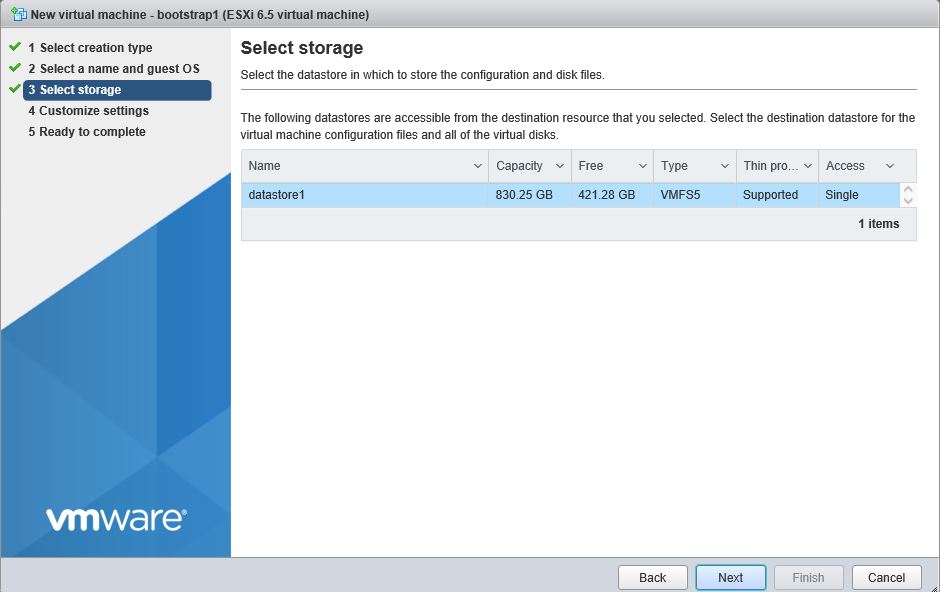
**2.7 Creating VMs**

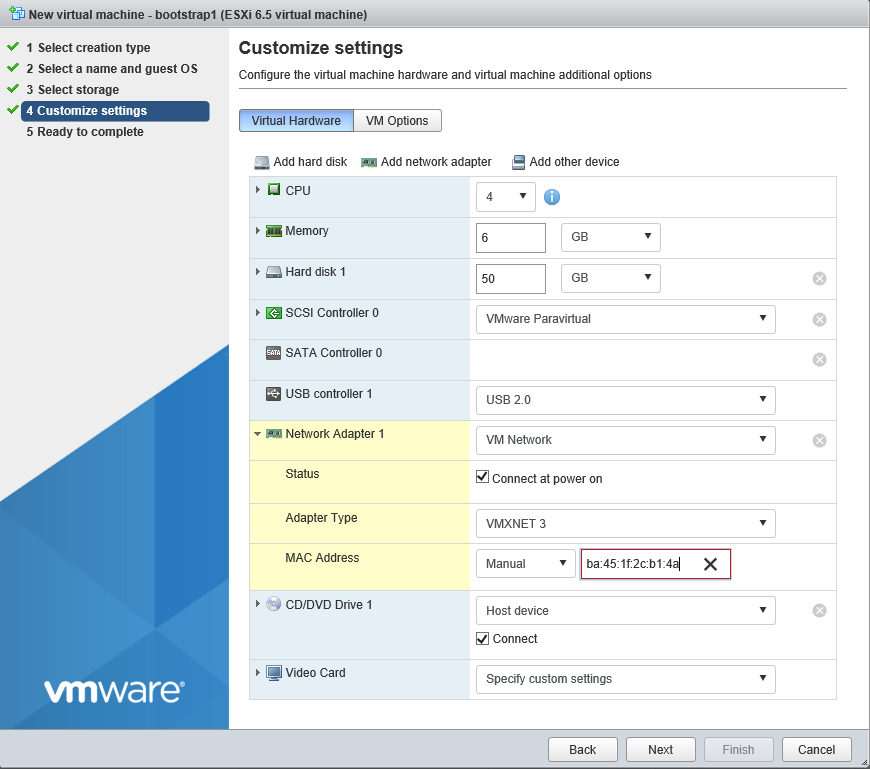
Here we are presenting to prepare VMWare Hosts (Example is given for Bootstrap and master1 the remaining VM configuration steps will be the same for each master, infra and worker). The important point is to give proper mac addresses on each device which is listed in the prerequisite step of the document and the name of VM.

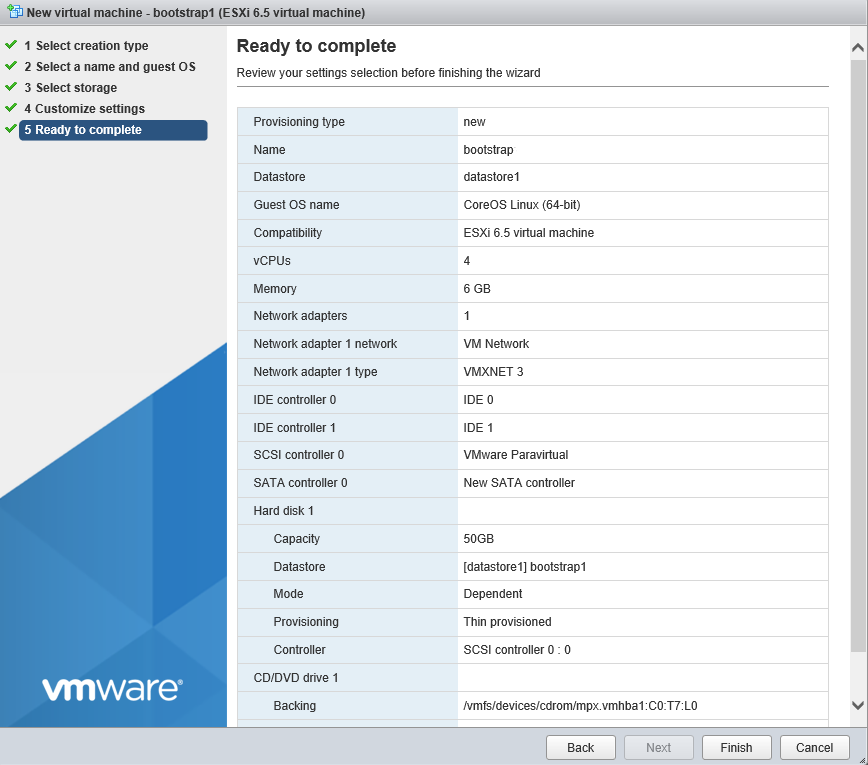
1. Bootstrap VM Creation



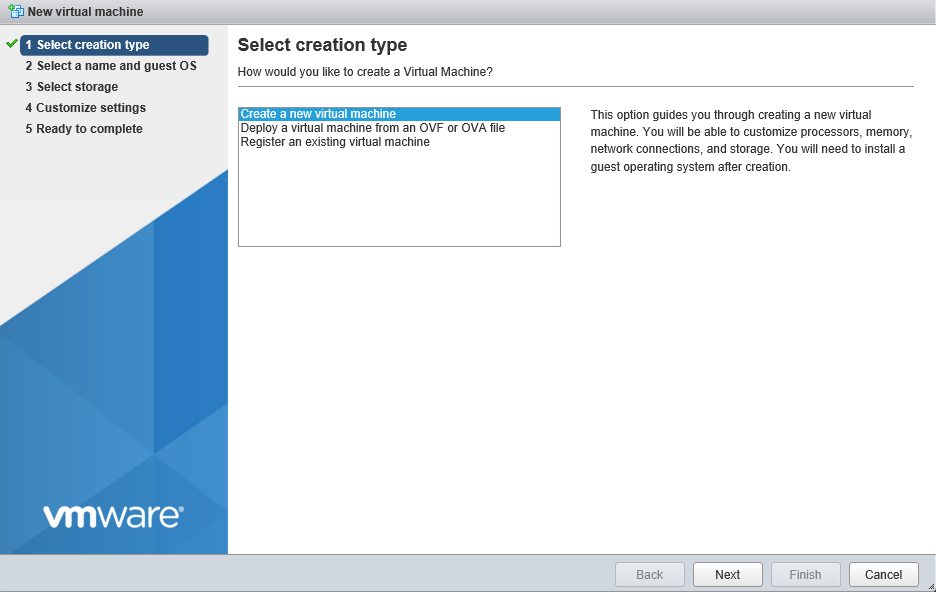


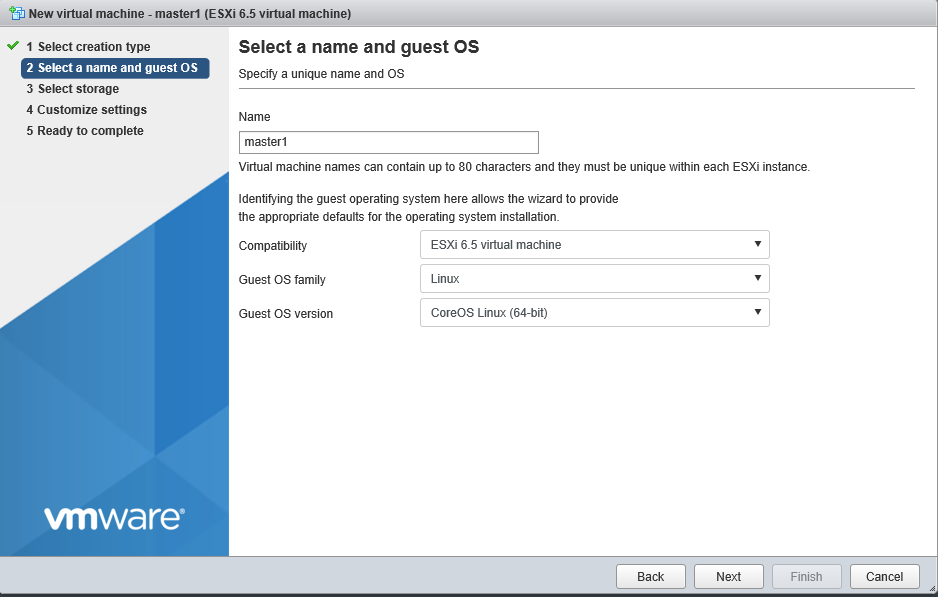


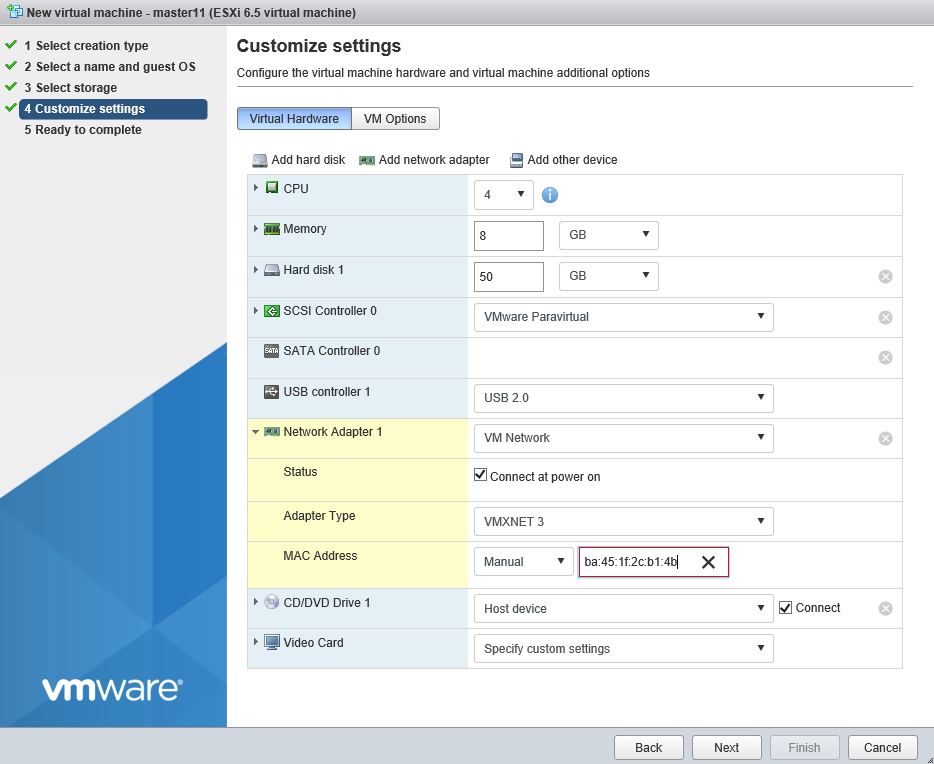


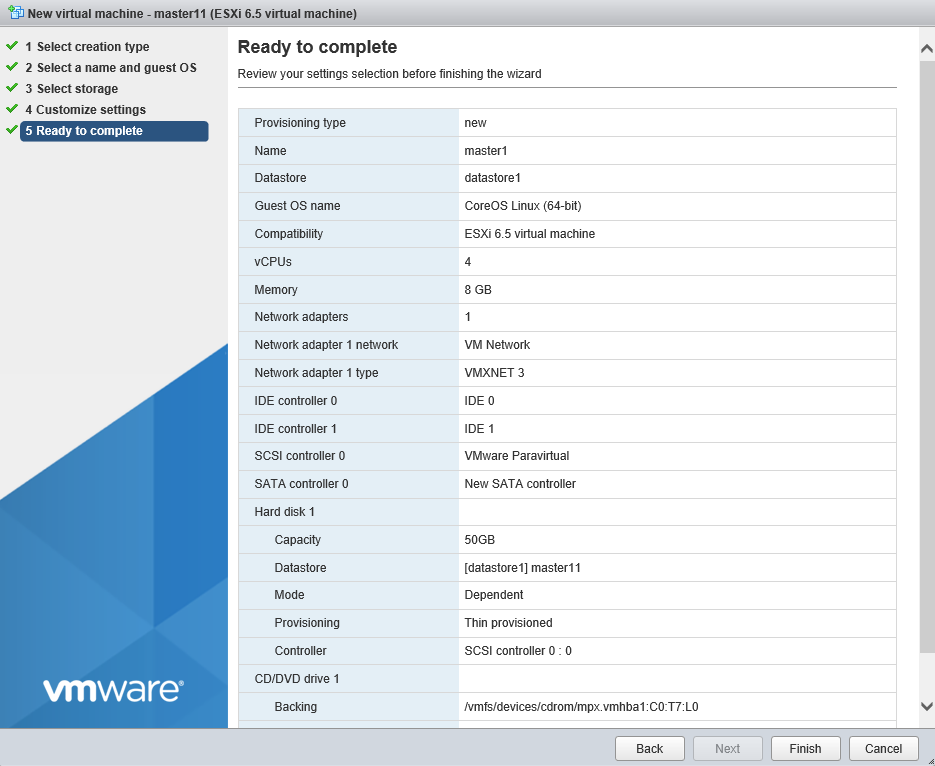


1. Creating Master1 VM, these steps will be the same for each master, each infra and each worker except the name of VM and MAC address.









1. After creating each VM, first boot bootstrap and do the following steps to check everything is going well.

[root@bastion ~]# ssh core@192.168.50.2

The authenticity of host '192.168.50.2 (192.168.50.2)' can't be established.

ECDSA key fingerprint is SHA256:DEFJSEfKStLUi/rvIPrwzMUSh0j6CAy0hWJRD0ZKx2A.

ECDSA key fingerprint is MD5:3b:2c:a1:0c:a8:9d:4b:b9:25:51:b5:64:91:c0:14:3d.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.50.2' (ECDSA) to the list of known hosts.

Red Hat Enterprise Linux CoreOS 410.8.20190520.0

WARNING: Direct SSH access to machines is not recommended.

This node has been annotated with machineconfiguration.openshift.io/ssh=accessed

---

This is the bootstrap node; it will be destroyed when the master is fully up.

The primary service is "bootkube.service". To watch its status, run e.g.

journalctl -b -f -u bootkube.service

Last login: Wed Oct 9 08:15:57 2019 from 192.168.50.9

[systemd]

Failed Units: 1

systemd-firstboot.service

[root@bastion ~]# sudo -i

[root@bastion ~]# journalctl -b -f -u bootkube.service

1. Once the “waiting for etcd” message appeared, boot each masters and infra and wait for etcd to come up.
2. After etcd successfully come up, boot all worker VMs.
3. It may take long time depending on your network speed.
4. After everything will be OK, browse the console with the following URL

<https://console-openshift-console.apps.ocp.tmrnd.net>

UserName: Kubeadmin

Password: cat /root/ocp4/ign/auth/Kubeadmin-password

1. To deploy applications, you need to setup persistent storage and allocate certain storage for openshift 4 registry. We are using rook-ceph but you can use any storage mechanism available in Red hat Openshift 4 storage documentation (<https://docs.openshift.com/container-platform/4.1/storage/understanding-persistent-storage.html>).

Enjoy OCP 4!!!!

References:

<https://www.techbeatly.com/2019/07/openshift-4-libvirt-upi-installation.html/>

<https://docs.openshift.com/container-platform/4.2/installing/installing_bare_metal/installing-bare-metal.html>

<https://blog.openshift.com/openshift-4-bare-metal-install-quickstart/>