OPENSHIFT INSTALLATION

ON A SINGLE VMWARE ESXI HOST

E-BOOK

w3Partnership

EXPERTISE TO INSPIRE CONFIDENCE

Contents

OpenShift Installation on a single VMware ESXi Host

The following instructions document the installation procedure to install a complete Red Hat OpenShift 4.5.x cluster on a single VMware ESXi host machine

N.B. The standard Red Hat installation instructions for VMWare concentrate on a central vCenter server that manages the VMware vSphere machines. This cannot be used.

The below instructions are a combination of a few installation procedures and utilises the Red Hat Bare metal installation instructions.

References:

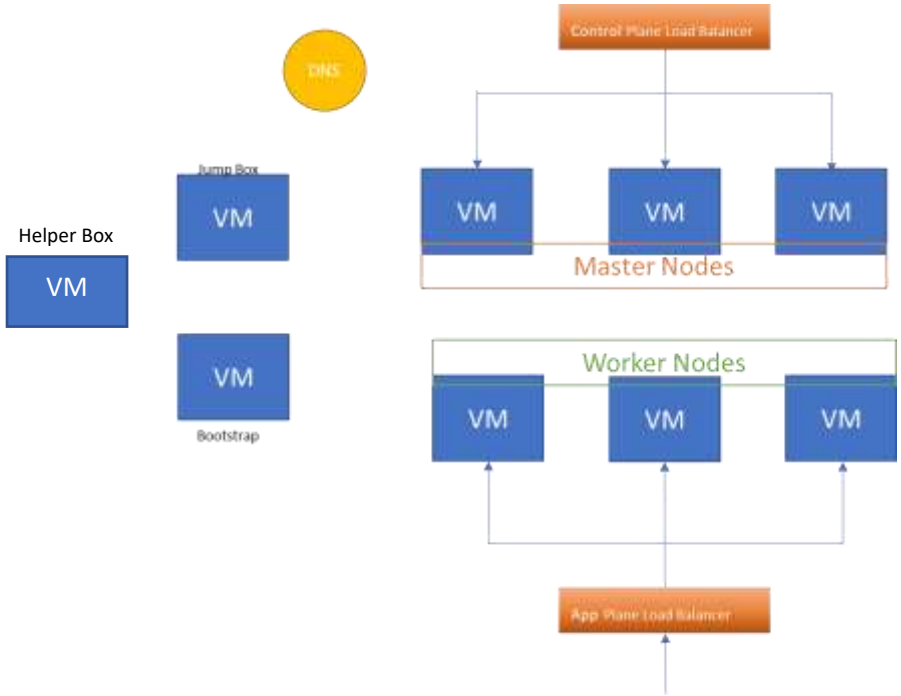
- https://www.youtube.com/watch?v=Be0dRq0wjWE
- https://www.openshift.com/blog/openshift-4-bare-metal-install-quickstart
- https://docs.openshift.com/container-platform/4.5/installing/installing_bare_metal/installing-bare-metal.html

Pre-requisites

- DNS Server
- Apache Web Server
- HAProxy
- VMWare ESXi

Architecture

The below image shows the overall architecture of an OpenShift environment



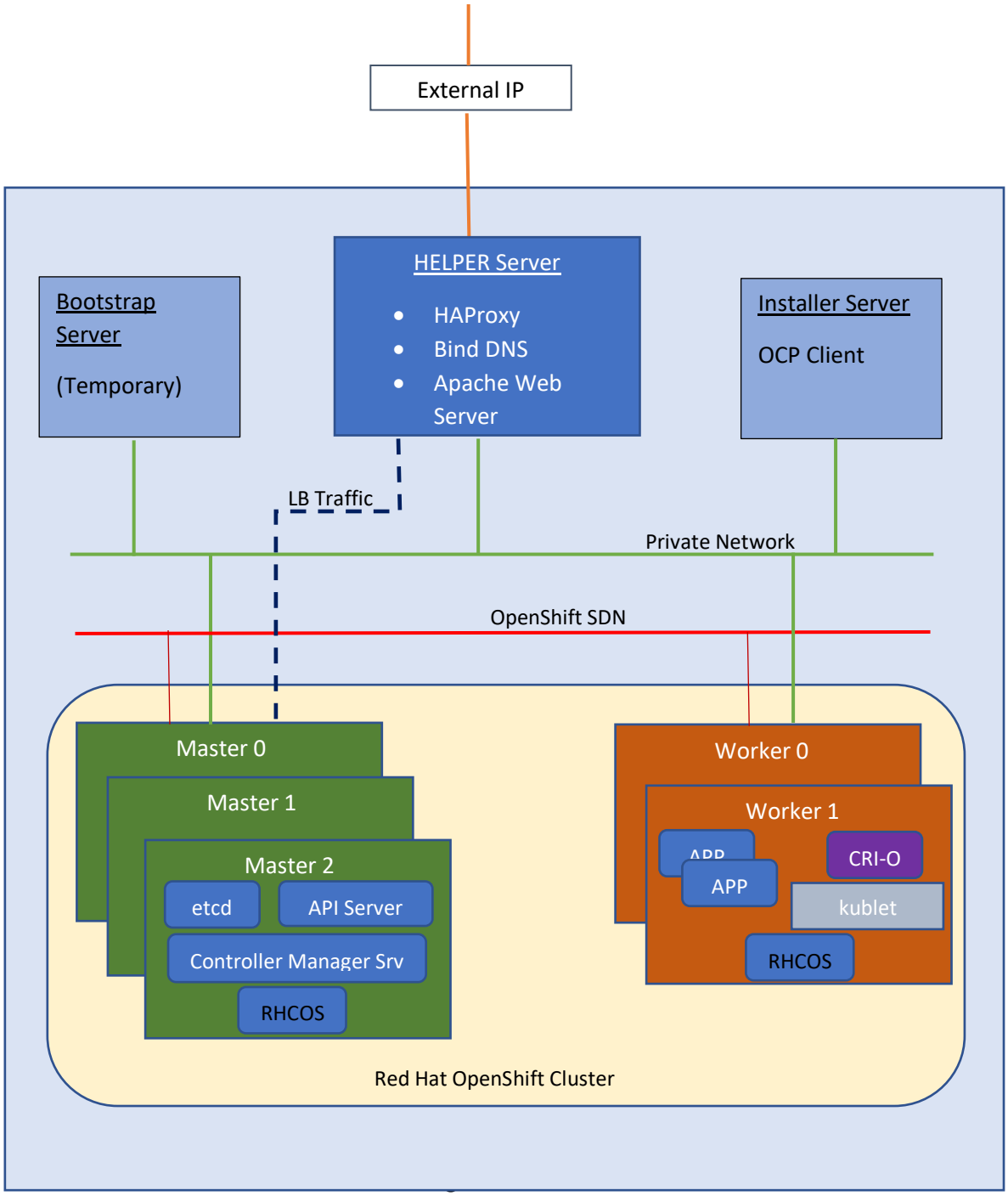
Solution Overview

The minimum requirement for an OpenShift cluster is 3 x master nodes and 2x worker nodes.

A temporary bootstrap server is required for the initial setup. This server contains the initial config for the master and worker nodes. Once the cluster is up and running, the bootstrap server can be removed.

In our solution, an installation server is used from which all the setup is run. A helper or bastion server is also required. This server houses the following components: - Bind DNS Server, Apache2 Webserver and HAProxy.

The below image shows our overall solution.



Domain Name = w3internal.com

Clustername = openshift4

VM Name	Hostname	Role	CPU	Ram	Storage	IP Address
bootstrap	bootstrap	OpenShift Bootstrap	4	16	120GB	192.168.1.210
openshiftInstaller	openshiftnstaller	Installer	1	4	16	192.168.1.211
Ubuntu_DNS_HAProxy	W3dnshaproxy	Bastion	1	4	16	192.168.1.200
Master0	Master0	OpenShift Control Plane	4	16	120	192.168.1.207
Master1	Master1	OpenShift Control Plane	4	16	120	192.168.1.202
Master2	Master2	OpenShift Control Plane	4	16	120	192.168.1.203
Worker0	Worker0	OpenShift Compute Node	2	8	120	192.168.1.204
Worker1	Worker1	OpenShift Compute Node	2	8	120	192.168.1.205

Gateway = 192.168.1.1

Required Software

- Ubuntu Server ISO
- openshift-client-linux.tar.gz
- openshift-install-linux.tar.gz
- rhcos-4.5.6-x86_64-metal.x86_64.raw.gz
- rhcos-installer.x86_64.iso

1. Setup DNS

Bind 9 is an open-source implementation of DNS. This DNS implementation will be installed on the "Helper Server" which is a vanilla install of Ubuntu 20.04.1 server.

With the Ubuntu VM running log in to the command prompt and do the following steps: -

1. update the apt package
   ```
   sudo apt-get update
   ```
2. Install BIND
   ```
   sudo apt-get install bind9 bind9utils bind9-doc
   ```
3. Configure options file
   ```
   sudo nano /etc/bind/named.conf.options
   ```

   Add the following lines below the "directory" directive

   ```
   Listen-on port 53 {localhost; 192.168.1.0/24;} ;
   allow-query {localhost; 192.168.1.0/24} ;
   forwarders {
   0.0.0.0;
   8.8.8.8;
   };
   recursions yes;
   ```

The resulting file should look something like:

4. Configure DNS Zones by editing named.conf.local fie

```
sudo nano /etc/bind/named.conf.local
```

In this file we'll specify our forward and reverse DNS zones. All our domains will be in the "w3internal.com" subdomain. We'll use this for our forward zone and since our IPs are within the 192.168.1.0/24 IP space, we will set up our reverse zone so that we can define reverse lookups within that range.

Add the forward zone with the following lines

```
zone "w3internal.com" {
    type master;
    file "/etc/bind/zones/forward.w3internal.com"; # zone file
    path
};
```

Assuming that our private subnet is 192.168.1.0/24, add the reverse zone by with the following lines (**note that our reverse zone name starts with "1.168.192" which is the octet reversal of "192.168.1"**)

```
zone "1.168.192.in-addr.arpa" {
  type master;
  file "/etc/bind/zones/reverse.w3internal.com";  #
  192.168.1.0/24 subnet
};
```

The resulting file should look like:

```
 GNU nano 4.8                          /etc/bind/named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
zone "w3internal.com" {
type master;
file "/etc/bind/zones/forward.w3internal.com";
};

zone "openshift4.w3internal.com" {
type master;
file "/etc/bind/zones/forward.w3internal.com";
};

zone "1.168.192.in-addr.arpa" {
type master;
file"/etc/bind/zones/reverse.w3internal.com";
};
```

We now need to specify the forward and reverse zone files

5. Create forward zone file

The forward zone file is where we define DNS records for forward DNS lookups. That is, when the DNS receives a name query, "host1.openshift4.w3internal.com" for example, it will look in the forward zone file to resolve **host1**'s corresponding private IP address.

```
sudo mkdir /etc/bind/zones
```

```
sudo cp /etc/bind/db.local /etc/bind/zones/forward.w3internal.com
```

Edit the new document so it looks like:

```
 GNU nano 4.8                          forward.w3internal.com
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     w3dnshaproxy.w3internal.com. root.w3internal.com. (
                              5         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
; name servers - NS records
        IN      NS      w3dnshaproxy.w3internal.com.

; name servers - A records
w3dnshaproxy.w3internal.com.    IN      A       192.168.1.200

; 192.168.1.0/24 - A records

$ORIGIN openshift4.w3internal.com.
api     IN      A       192.168.1.200
api-int IN      A       192.168.1.200
bootstrap       IN      A       192.168.1.210
master0 IN      A       192.168.1.207
etcd-0  IN      A       192.168.1.207
master1 IN      A       192.168.1.202
etcd-1  IN      A       192.168.1.202
master2 IN      A       192.168.1.203
etcd-2  IN      A       192.168.1.203
worker0 IN      A       192.168.1.204
worker1 IN      A       192.168.1.205
worker3 IN      A       192.168.1.206
```

```
_etcd-server-ssl._tcp   86400   IN      SRV     0       10      2380    etcd-0.openshift4.w3interna
_etcd-server-ssl._tcp   86400   IN      SRV     0       10      2380    etcd-1.openshift4.w3interna
_etcd-server-ssl._tcp   86400   IN      SRV     0       10      2380    etcd-2.openshift4.w3interna

$ORIGIN apps.openshift4.w3internal.com.
*       A       192.168.1.200
```

N.B Every time you edit a zone file, you need to increment the **serial** value before you restart the named process

6. Create reverse zone file

```
sudo cp /etc/bind/db.127 /etc/bind/zones/reverse.w3internal.com
```

Edit the file so that it looks like:

```
w3admin@w3dnshaproxy:/etc/bind/zones$ cat reverse.w3internal.com
;
; BIND reverse data file for local loopback interface
;
;$ORIGIN .
$TTL    604800
@       IN      SOA     w3dnshaproxy.w3internal.com. root.w3internal.com.
                              6         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

; name servers - NS records
        IN NS   w3dnshaproxy.w3internal.com.

; PTS Records
200     IN      PTR     w3dnshaproxy.w3internal.com.    ; 192.168.1.200

200     PTR     api.openshift4.w3internal.com.
        PTR     api-int.openshift4.w3internal.com.
210     PTR     bootstrap.openshift4.w3internal.com.
201     PTR     master0.openshift4.w3internal.com.
202     PTR     master1.openshift4.w3internal.com.
203     PTR     master2.openshift4.w3internal.com.
204     PTR     worker0.openshift4.w3internal.com.
205     PTR     worker1.openshift4.w3internal.com.
206     PTR     worker2.openshift4.w3internal.com.
```

7. Check the configuration

```
sudo named-checkconf
```

If there are no errors, the command prompt will return

```
sudo named-checkzone w3internal.com /etc/bind/forward.w3internal.com
sudo named-checkzone 1.168.192 /etc/bind/zones/reverse.w3internal.com
```

If there are no errors, then we can restart the BIND service

8. Start bind service

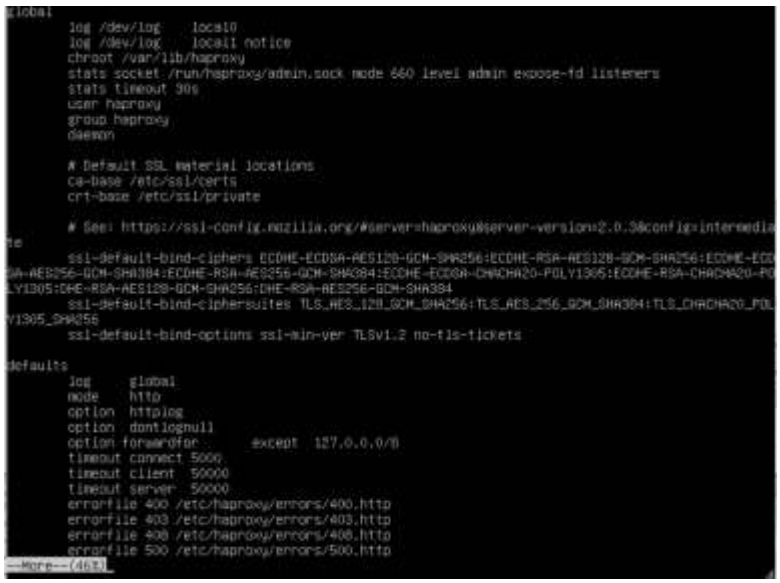
```
sudo systemctl restart bind9
sudo systemctl enable bind9
```

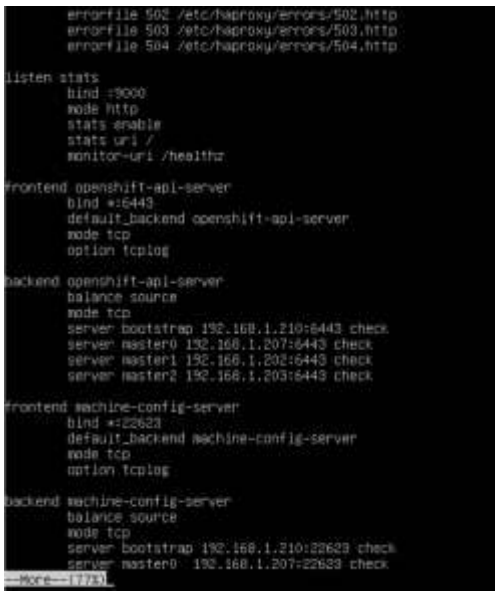
2. Setup HAProxy

As part of this solution, a load balancer is required for Kubernetes API server, both internal and external as well as for the OpenShift router.

In this deployment we will use HAProxy. This will be installed on the “helper ubuntu VM”.

1. Update sources list
   ```
   sudo apt update
   ```
2. Install HAProxy
   ```
   sudo apt install -y haproxy
   ```
3. Once installed, configure /etc/haproxy/haproxy.cfg. We need to add port 6443 and 22623 to point to the bootstrap and master nodes. We also need to add ports 80 and 443 to point to the worker nodes. The resulting config should look like the below:





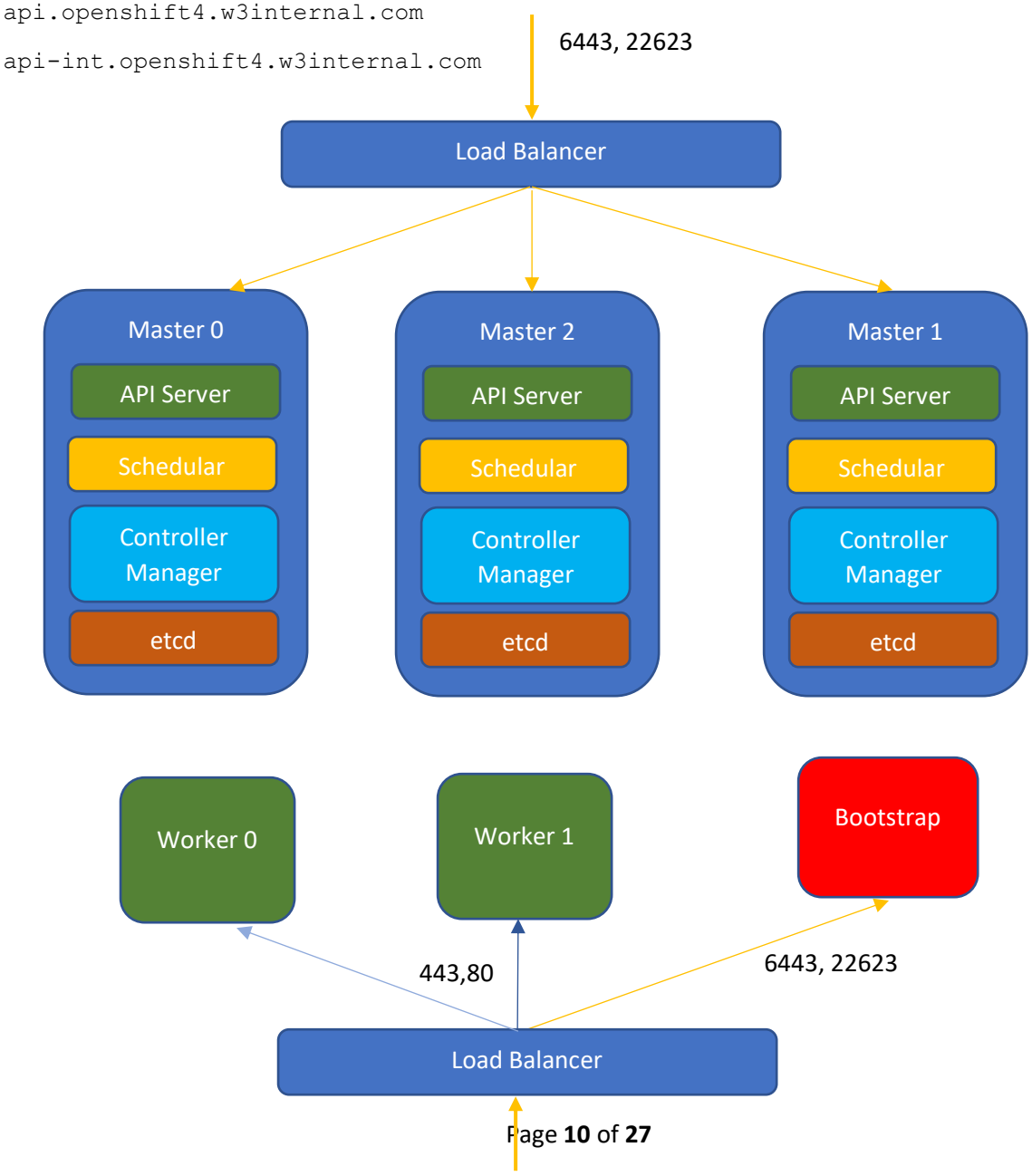
```
        server master1  192.168.1.202:22623 check
        server master2  192.168.1.203:22623 check

frontend ingress-http
        bind *:80
        default_backend ingress-http
        mode tcp
        option tcplog

backend ingress-http
        balance source
        mode tcp
        server worker0 192.168.1.204:80 check
        server worker1 192.168.1.205:80 check
        server worker2 192.168.1.206:80 check

frontend ingress-https
        bind *:443
        default_backend ingress-https
        mode tcp
        option tcplog

backend ingress-https
        balance source
        mode tcp
        server worker0 192.168.1.204:443 check
        server worker1 192.168.1.205:443 check
        server worker2 192.168.1.206:443 check
```



4. The HAProxy config can be tested by running the following:
```
haproxy -f /etc/haproxy/haproxy.cfg -c -V
```
5. Restart HAproxy
```
Systemctl restart haproxy
Systemctl status haproxy
```

6. We have also enabled stats in the HAProxy configure. The stats can be viewed using the following link (http://<load balancer public IP>/haproxy?stats)

3. Setup WebServer

A webserver is also required to be setup for placing ignition configurations and installation images for Red Hat CoreOS. Webserver must be reached by bootstrap, master, and worker nodes during the install.

In this install we will use Apache. Log into the helper box / server

```
sudo apt update
sudo apt install apache2
```

1. Check to ensure the web server is running
```
sudo systemctl status apache2
```
2. Create the following directories
   a. `mkdir -p /var/www/html/ignition`
   b. `mkdir -p /var/www/html/install`
3. Create SSH Key

On the installer server, generate an SSH Key

```
ssh-keygen -t rsa -b 4096 -N '' -f ~/.ssh/id_rsa
```

- start the ssh-agent
```
eval "$(ssh-agent -s)"
```
- Add SSH key to the ssh-agent
```
ssh-add ~/.ssh/id_rsa
```

4. Setup OpenShift

In our installation of OpenShift we will be using static Ips. The following instructions are based on this. If DHCP is used, the instructions will be slightly different. Please refer to the red hat documentation.

1. Log onto the installer server and create a directory called openshift_install_dir

```
mkdir openshift_install_dir
```

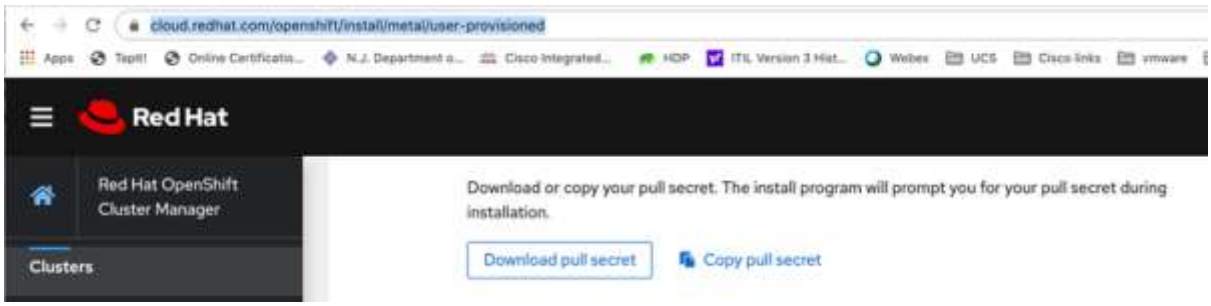
2. From within this directory unzip the contents of *openshift-install-linux.tar.gz* . this will result in the executable file openshift-install.sh to be present.
3. Unzip the command line binary *openshift-client-linux.tar.gz,* and place the *oc* binary into a directory that is in your path e.g /usr/local/bin

```
w3admin@openshiftinstaller:~$ ls -l /usr/local/bin
total 76760
-rwxr-xr-x 1 root root 78599240 Oct  9 16:10 oc
```

Installation program requires pull secret. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

Without pull secret, installation will not continue. It will be specified in install config file.

4. Download the pull secret as a .txt file from the OpenShift Cluster Manager site. E.g.



5. Create the install-config.yaml from the given template. It should end up looking like: -

```
apiVersion: v1
baseDomain: w3internal.com [1]
compute:
 - hyperthreading: Enabled   [2] [3]
   name: worker
   replicas: 0 [4]
controlPlane:
  hyperthreading: Enabled   [2] [3]
  name: master [3]
  replicas: 3  [5]
metadata:
  name: openshift4 [6]
networking:
```

```
  clusterNetworks:
  - cidr: 10.128.0.0/14 7
    hostPrefix: 23 8
  networkType: OpenShiftSDN
  serviceNetwork:  9
  - 172.30.0.0/16
platform:
  none: {} 10
fips: false 11
pullSecret: <PULL SECRET FROM FILE> 12
sshKey: <GENERATED SSH KEY> 13
```

where:

1 The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2 The `controlPlane` section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used

3 Whether to enable or disable simultaneous multithreading, or `hyperthreading`. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to `Disabled`. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

4 You must set the value of the `replicas` parameter to `0`. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

5 The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

6 The cluster name that you specified in your DNS records.

7 A block of IP addresses from which Pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the Pod network. If you need to access the Pods from an external network, you must configure load balancers and routers to manage the traffic.

8 The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to `23`, then each node is assigned a `/23` subnet out of the given `cidr`, which allows for 510 (2^(32 - 23) - 2)

pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

9 The IP address pool to use for service IP addresses. You can enter only one IP address pool. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

10 You must set the platform to `none`. You cannot provide additional platform configuration variables for bare metal infrastructure.

11 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

12 The pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

13 The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

N. B the install-config.yaml we are using is based on the bare metal install. The install config file will change slightly based on where the installation will take place. Refer to the red hat OpenShift install documentation for further examples.

6. Copy the install-config.yaml file to the openshift_install_dir on the installer server
N.B make a backup of this file, since the following commands will delete this file once completed.

7. Generate the Kubernetes manifests for the cluster

```
./openshift-install create manifests -- dir=/home/w3admin/openshift_install_dir
```

8. Modify the <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes manifest file to prevent Pods from being scheduled on the control plane machines:

- Open the <installation_directory>/manifests/cluster-scheduler-02-config.yml file.
- Locate the mastersSchedulable parameter and set its value to False.
- Save and exit the file.

9. Create the ignition files

```
./openshift-install create ignition-configs --dir=/home/w3admin/openshift_install_dir
```

The following files will be created

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

```
w3admin@openshiftnstaller:~/openshift_install_dir$ ls -l
total 359944
drwxr-x--- 2 w3admin w3admin      4096 Oct 14 10:29 auth
-rw-r----- 1 w3admin w3admin    298651 Oct 14 10:29 bootstrap.ign
-rw-r----- 1 w3admin w3admin      1830 Oct 14 10:29 master.ign
-rw-r----- 1 w3admin w3admin       108 Oct 14 10:29 metadata.json
-rwxr-xr-x 1 w3admin w3admin 368267264 Oct 14 10:21 openshift-install
-rw-r----- 1 w3admin w3admin      1830 Oct 14 10:29 worker.ign
w3admin@openshiftnstaller:~/openshift_install_dir$
```

10. Copy the *.ign files to the webserver in the folder /var/www/html/ignition e.g.

```
scp *.ign w3admin@192.168.1.200:/var/www/html/ignition
```

11. Copy the rhcos-4.5.6-x86_64-metal.x86_64.raw.gz to the webserver into folder /var/www/html/install

12. Copy the rhcos-installer.x86_64.iso to a directory on the installer server e.g., /home/w3admin/installer

The traditional way to create each required image (i.e., master 0, master 1, worker 0, worker 1 etc) is to boot the machine using the iso file and then add the required parameters to the kernel command line. However, we will customise the iso for each image that we want.

13. On the installer server:

a. `mkdir rhcos-installer-modified`

b. `mkdir iso`

```
w3admin@openshiftnstaller:~/installer$ ls -l
total 1328288
drwxrwxr-x 2 w3admin w3admin      4096 Oct  9 11:24 iso
-rwxr-xr-x 2 w3admin w3admin  78599240 Sep 16 16:27 kubectl
-rwxr-xr-x 2 w3admin w3admin  78599240 Sep 16 16:27 oc
-rw-rw-r-- 1 w3admin w3admin  25908982 Oct  9 16:08 openshift-client-linux.tar.gz
-rw-r--r-- 1 w3admin w3admin  92438128 Oct  9 16:01 openshift-install-linux.tar.gz
-rw-r--r-- 1 w3admin w3admin       954 Sep 16 16:27 README.md
-rw-r--r-- 1 root    root     92796928 Oct 13 12:02 rhcos-4.5.6-modified.iso
-rw-rw-r-- 1 w3admin w3admin 898463618 Oct 13 17:14 rhcos-4.5.6-x86_64-metal.x86_64.raw.gz
drwxrwxr-x 5 w3admin w3admin      4096 Oct  9 11:29 rhcos-installer-modified
-rw-rw-r-- 1 w3admin w3admin  93323264 Oct  9 11:04 rhcos-installer.x86_64.iso
drwxrwxr-x 2 w3admin w3admin      4096 Oct  9 12:53 test
```

c. `sudo mount -o loop rhcos-installer.x86_64.iso /home/w3admin/installer/iso`

d. `ls -l iso` ------ to check that it has been mounted

e. Now copy the file from the original iso to the modified directory

```
rsync -av iso/* rhcos-installer-modified/
```

f. change into the directory rhcos-installer-modified

```
cd isolinux
```

g. edit isolinux.cfg

Add the following lines to the file:

```
label master0
  menu label ^Install master0
  kernel /images/vmlinuz
  append initrd=/images/initramfs.img nomodeset rd.neednet=1
coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.1.200:81/install/rhcos-4.5.6-
x86_64-metal.x86_64.raw.gz
coreos.inst.ignition_url=http://192.168.1.200:81/ignition/master.ign
ip=192.168.1.201::192.168.1.1:255.255.255.0:master0.openshift4.w3inte
rnal.com:ens192:none nameserver=192.168.1.200

label master1
  menu label ^Install master1
  kernel /images/vmlinuz
  append initrd=/images/initramfs.img nomodeset rd.neednet=1
coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.1.200:81/install/rhcos-4.5.6-
x86_64-metal.x86_64.raw.gz
coreos.inst.ignition_url=http://192.168.1.200:81/ignition/master.ign
ip=192.168.1.202::192.168.1.1:255.255.255.0:master1.openshift4.w3inte
rnal.com:ens192:none nameserver=192.168.1.200

label master2
  menu label ^Install master2
  kernel /images/vmlinuz
  append initrd=/images/initramfs.img nomodeset rd.neednet=1
coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.1.200:81/install/rhcos-4.5.6-
```

```
x86_64-metal.x86_64.raw.gz
coreos.inst.ignition_url=http://192.168.1.200:81/ignition/master.ign
ip=192.168.1.203::192.168.1.1:255.255.255.0:master2.openshift4.w3inte
rnal.com:ens192:none nameserver=192.168.1.200

      label worker0
        menu label ^Install worker0
        kernel /images/vmlinuz
        append initrd=/images/initramfs.img nomodeset rd.neednet=1
coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.1.200:81/install/rhcos-4.5.6-
x86_64-metal.x86_64.raw.gz
coreos.inst.ignition_url=http://192.168.1.200:81/ignition/worker.ign
ip=192.168.1.204::192.168.1.1:255.255.255.0:worker0.openshift4.w3inte
rnal.com:ens192:none nameserver=192.168.1.200

      label worker1
        menu label ^Install worker1
        kernel /images/vmlinuz
        append initrd=/images/initramfs.img nomodeset rd.neednet=1
coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.1.200:81/install/rhcos-4.5.6-
x86_64-metal.x86_64.raw.gz
coreos.inst.ignition_url=http://192.168.1.200:81/ignition/worker.ign
ip=192.168.1.205::192.168.1.1:255.255.255.0:worker1.openshift4.w3inte
rnal.com:ens192:none nameserver=192.168.1.200
```

14. Create the new ISO

```
sudo mkisofs -U -A rhcos-4.5.6-modified -V rhcos-4.5.6-modified -
volset rhcos-4.5.6-modified -J -joliet-long -r -v -T -x ./lost+found -o
~/installer/rhcos-4.5.6-modified.iso -b isolinux/isolinux.bin -c
isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -
eltorito-alt-boot -e images/efiboot.img -no-emul-boot
```

N.B If not present, then install mkisofs onto the installer server

15. Test the new iso by running

```
sudo mount -o loop rhcos-4.5.6-modified.iso test
```

16. Enter the test directory and then "`cat isolinux.cfg`" file. You should see our new entries.

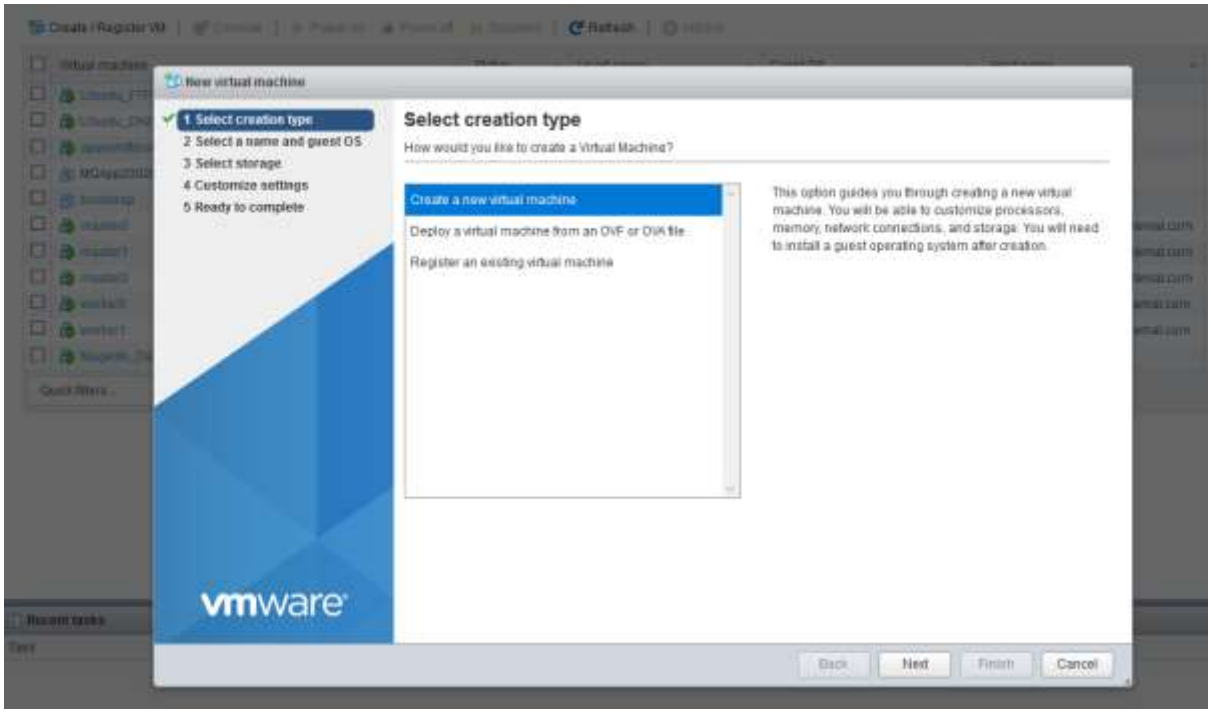
17. Unmount the iso

```
sudo unmount test
```

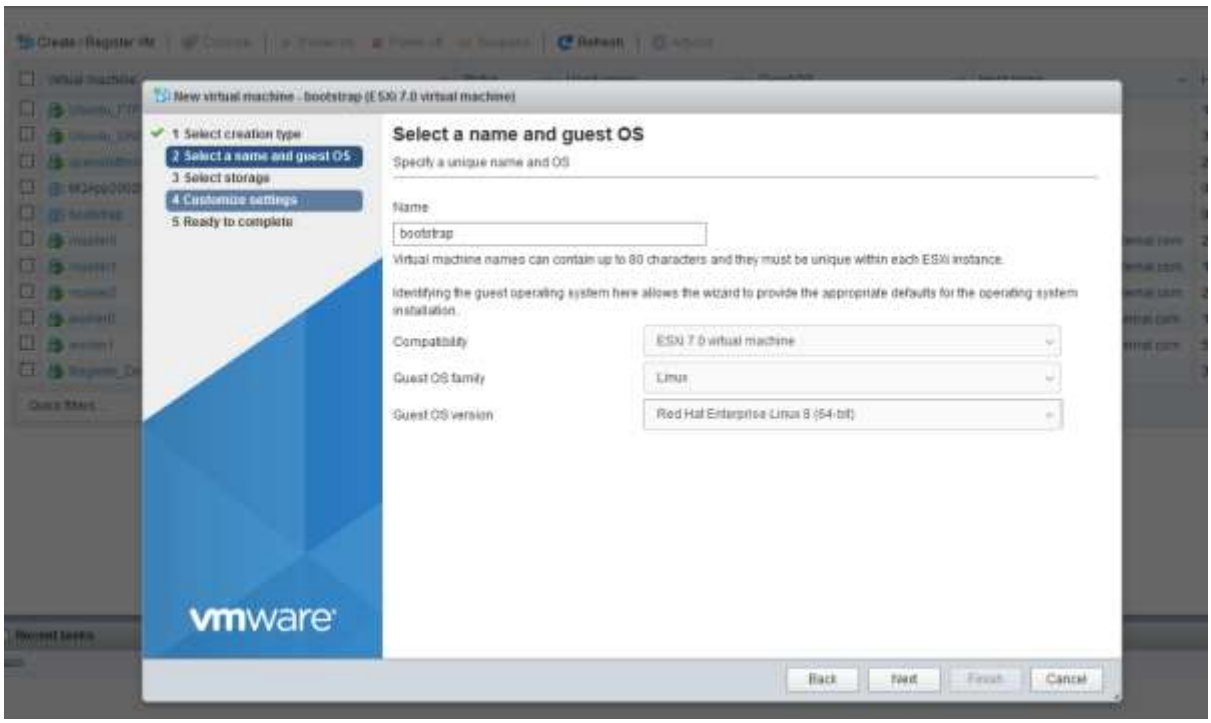
18. Now upload the newly created iso file to the ESXi datastore.

Creating VMs

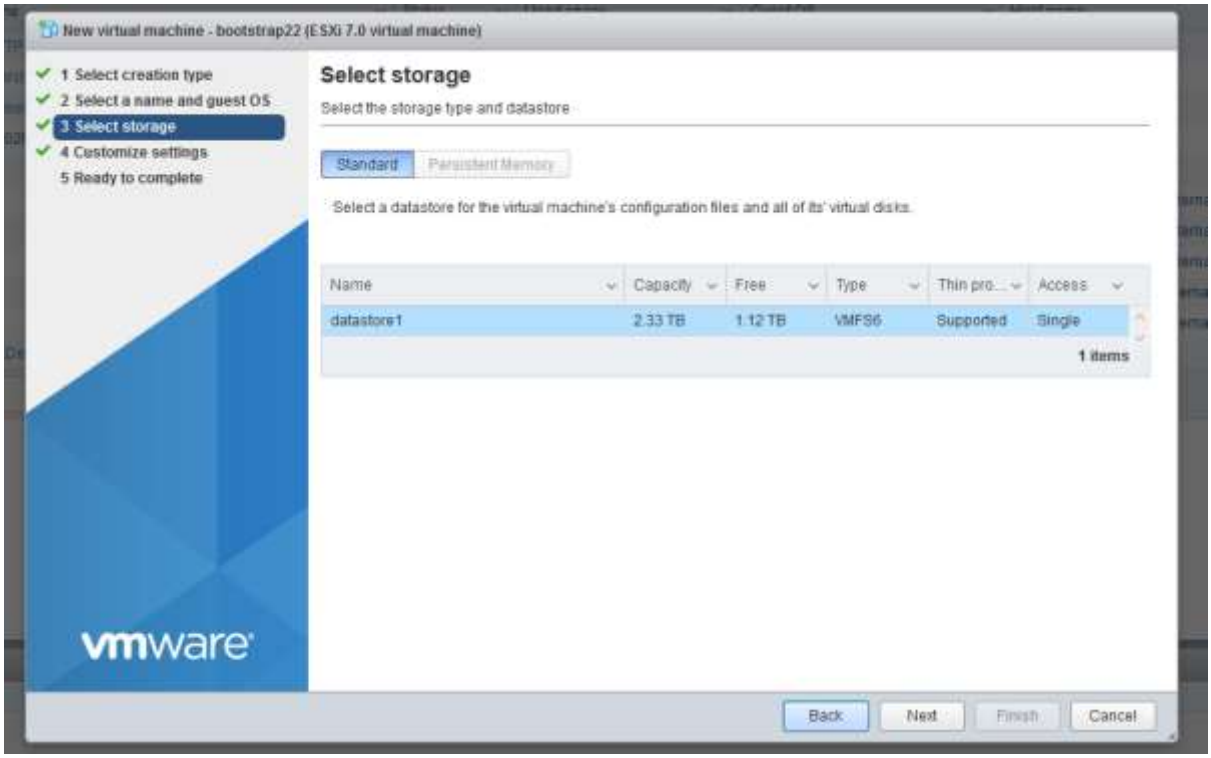
1. Create the required VMs (bootstrap, master0, master1, master 2, worker0, worker 1) by following the below instructions:
   - Select create new virtual machine



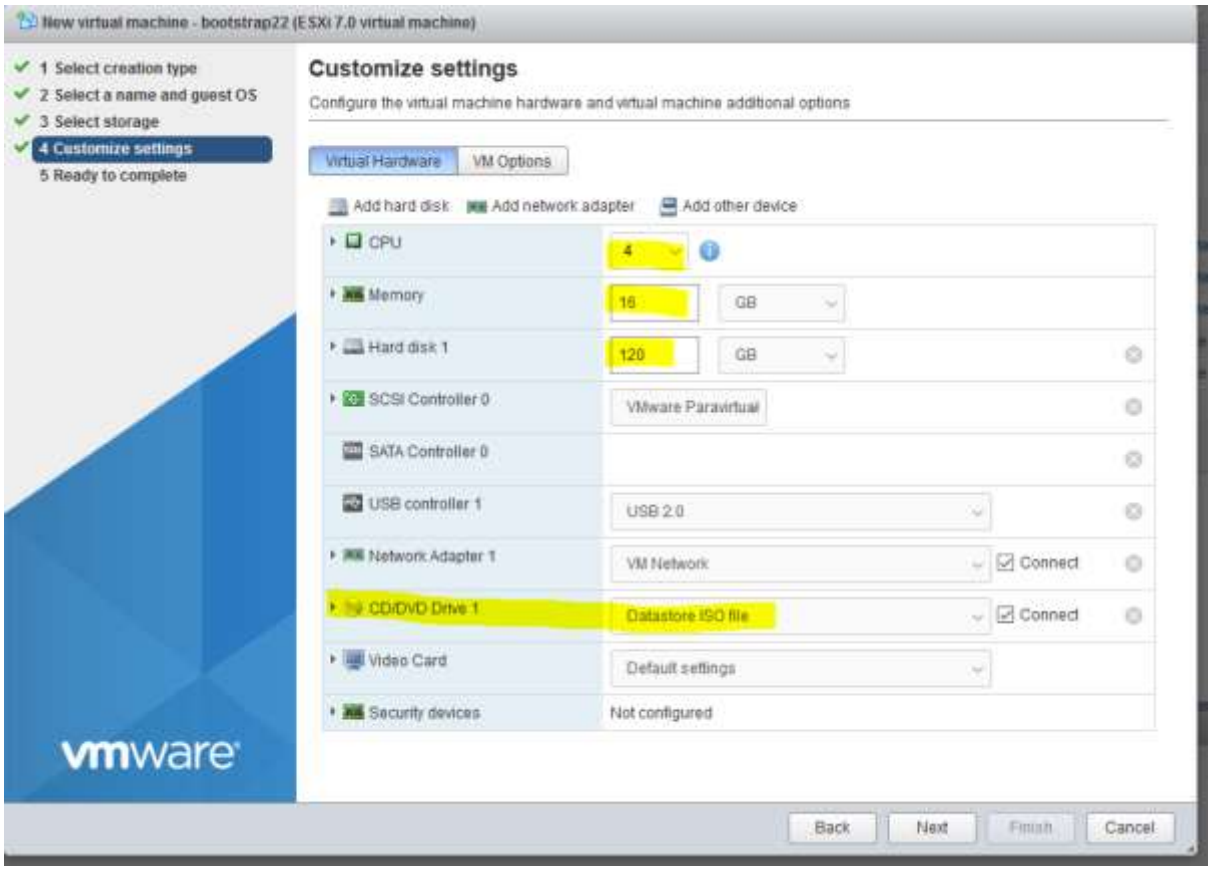
- Enter the VM name to be created and select the appropriate values



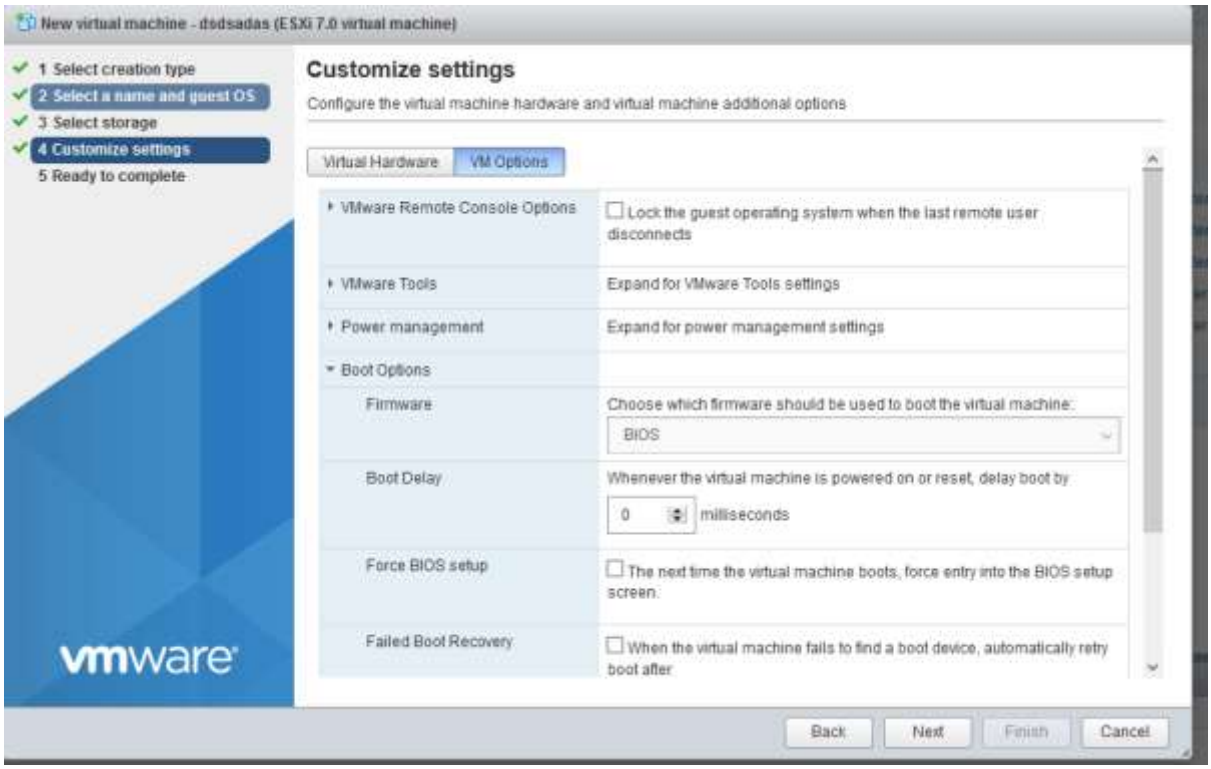
- Click next to Select Storage



- On the customize settings screen
    a. Enter the correct CPU, mem and hard disk size for the image you are creating (see table at beginning of doc)
    b. Ensure the cd/dvd drive 1 option is pointing to the datastore iso file (the modified iso that we created)

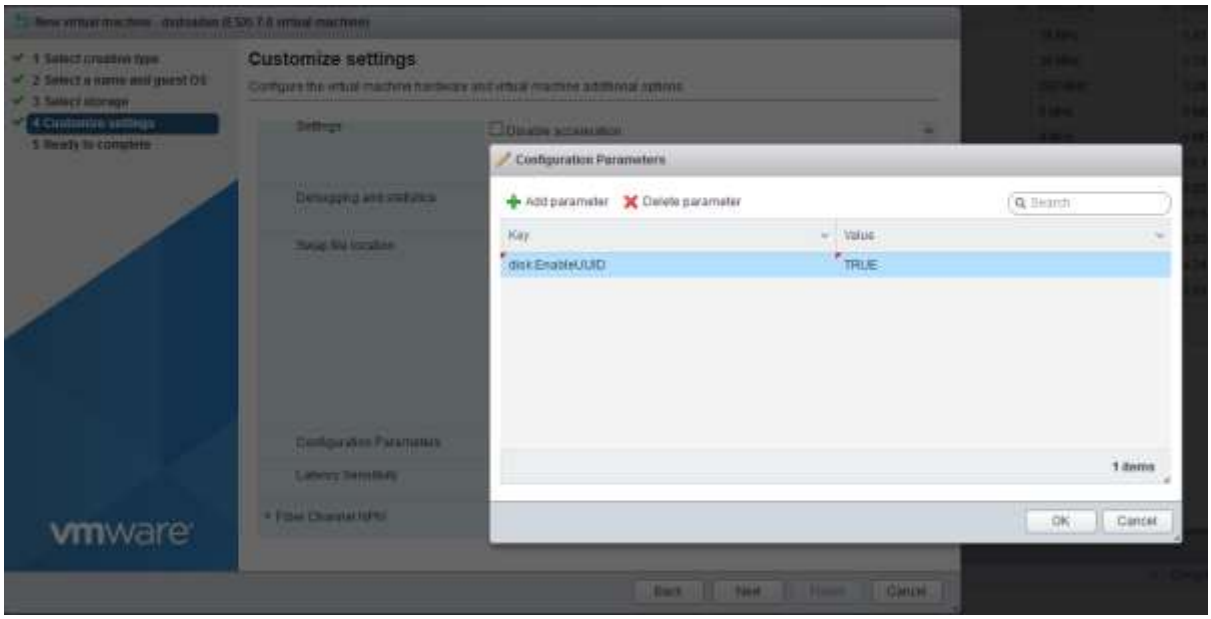


- In the VM Options tab, ensure the following are set:
    - Boot options to BIOS



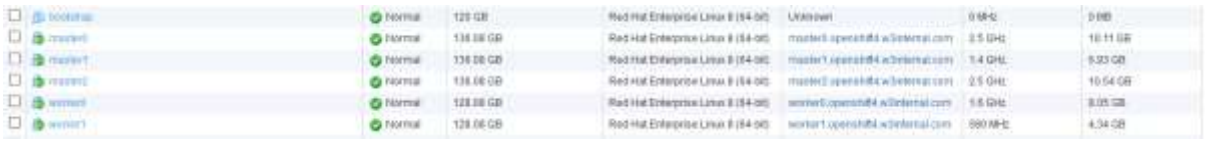
    - Under Advanced -> config parameters

      disk.EnableUUID = TRUE



- Complete the wizard and finish

Repeat the steps for the remaining VMs. Once done you should end up with the following:



2. Start the bootstrap VM. On start-up, you should be presented with a boot menu. Select the bootstrap option and press enter. The install of the bootstrap VM should start.
3. Start the bootstrap VM and repeat for the other VMs.

N.B. a few get errors will be seen initially. If the certificate is valid (see issues section below), the HAProxy will mark the server as up and then install will eventually proceed.

Configuration

1. Log onto the installer server and run

   ```
   ./openshift-install wait-for bootstrap-complete --log-level=info
   ```

   N.B. this process can take some time

2. Once bootstrap process is finished, the bootstrap VM can be turned off
3. On installer node do

   ```
   export KUBECONFIG=/home/w3admin/openshift_install_dir/auth/kubeconfig
   ```

4. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami

   system:admin
   ```

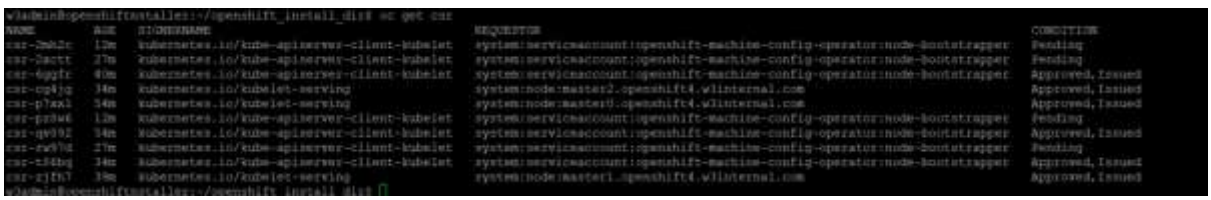
5. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   ```

```
w3admin@openshiftnstaller:~/openshift_install_dir$ oc get nodes
NAME                                STATUS   ROLES    AGE   VERSION
master0.openshift4.w3internal.com   Ready    master   52m   v1.18.3+47c0e71
master1.openshift4.w3internal.com   Ready    master   37m   v1.18.3+47c0e71
master2.openshift4.w3internal.com   Ready    master   32m   v1.18.3+47c0e71
w3admin@openshiftnstaller:~/openshift_install_dir$
```

Review the pending certificate signing requests (CSRs) and ensure that the you see a client and server request with `Pending` or `Approved` status for each machine that you added to the cluster:

```
oc get csr
```



6. To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>

To approve all pending CSRs, run the following command:


oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.na
me}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve


N.B. Because the CSRs rotate automatically, approve your CSRs within an hou
r of adding the machines to the cluster. If you do not approve them within
an hour, the certificates will rotate, and more than two certificates will
be present for each node. You must approve all of these certificates. After
you approve the initial CSRs, the subsequent node client CSRs are automatic
ally approved by the cluster kube-controller-manager. You must implement a
method of automatically approving the kubelet serving certificate requests.
```

7. On the installer server run

```
./openshift-install wait-for install-complete --log-level=debug
```

8. Wait for the install to finish.
   N.B this can take some time to complete.

9. View the cluster operators by running

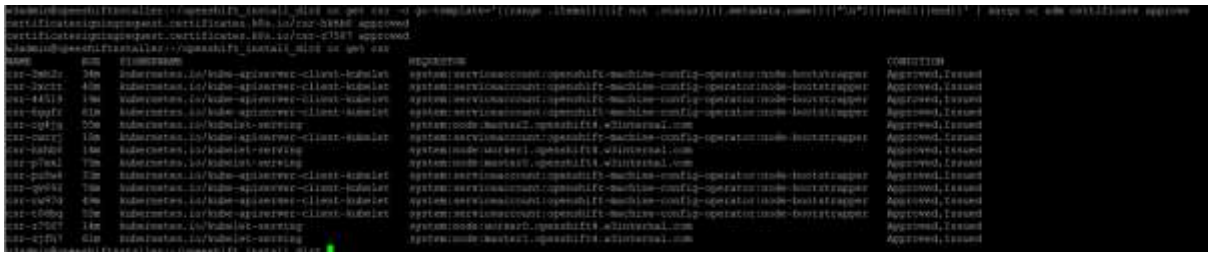
```
oc get co
```

```
w3admin@openshiftnstaller:~/openshift_install_dir$ oc get co
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE
authentication                             4.5.13    True        False         False      2m14s
cloud-credential                           4.5.13    True        False         False      84m
cluster-autoscaler                         4.5.13    True        False         False      44m
config-operator                            4.5.13    True        False         False      44m
console                                    4.5.13    True        False         False      6m4s
csi-snapshot-controller                    4.5.13    True        False         False      7m36s
dns                                        4.5.13    True        False         False      68m
etcd                                       4.5.13    True        False         False      50m
image-registry                             4.5.13    True        False         False      45m
ingress                                    4.5.13    True        False         False      8m21s
insights                                   4.5.13    True        False         False      45m
kube-apiserver                             4.5.13    True        False         False      49m
kube-controller-manager                    4.5.13    True        False         False      68m
kube-scheduler                             4.5.13    True        False         False      68m
kube-storage-version-migrator              4.5.13    True        False         False      8m16s
machine-api                                4.5.13    True        False         False      45m
machine-approver                           4.5.13    True        False         False      48m
machine-config                             4.5.13    True        False         False      46m
marketplace                                4.5.13    True        False         False      45m
monitoring                                 4.5.13    True        False         False      7m10s
network                                    4.5.13    True        False         False      70m
node-tuning                                4.5.13    True        False         False      70m
openshift-apiserver                        4.5.13    True        False         False      45m
openshift-controller-manager               4.5.13    True        False         False      45m
openshift-samples                          4.5.13    True        False         False      44m
operator-lifecycle-manager                 4.5.13    True        False         False      69m
operator-lifecycle-manager-catalog         4.5.13    True        False         False      69m
operator-lifecycle-manager-packageserver   4.5.13    True        False         False      45m
service-ca                                 4.5.13    True        False         False      70m
storage                                    4.5.13    True        False         False      45m
w3admin@openshiftnstaller:~/openshift_install_dir$
```

10. Check for any pending certificates again

```
oc get csr
```

11. Approve any pending csrs



12. Once all certs have been approved, check the stats page. It should look like

N.B. In the above diagram worker 2 is red since it has not been configured in this install and bootstrap is red, since the VM has been disabled once the setup was complete. If 3 worker nodes are provisioned, then only 2 worker nodes at any one time will be up.

13. Run the "`oc get node`" command. This shows that workers have joined and are uploaded.

```
w3admin@openshiftnstaller:~/openshift_install_dir$ oc get node
NAME                                STATUS   ROLES    AGE   VERSION
master0.openshift4.w3internal.com   Ready    master   88m   v1.18.3+47c0e71
master1.openshift4.w3internal.com   Ready    master   73m   v1.18.3+47c0e71
master2.openshift4.w3internal.com   Ready    master   67m   v1.18.3+47c0e71
worker0.openshift4.w3internal.com   Ready    worker   26m   v1.18.3+47c0e71
worker1.openshift4.w3internal.com   Ready    worker   26m   v1.18.3+47c0e71
w3admin@openshiftnstaller:~/openshift_install_dir$
```

14. For a non-production environment, you might need to configure an image registry.

    To verify that we have an image registry setup, run the following:

    `oc get pod -n openshift-image-registry`

```
w3admin@openshiftnstaller:~/openshift_install_dir$ oc get pod -n openshift-image-registry
NAME                                               READY   STATUS    RESTARTS   AGE
cluster-image-registry-operator-75d6b9f9bf-nf7nc   2/2     Running   0          72m
node-ca-81jz6                                      1/1     Running   0          70m
node-ca-m21xw                                      1/1     Running   0          36m
node-ca-scdpj                                      1/1     Running   0          70m
node-ca-v2j89                                      1/1     Running   0          36m
node-ca-xwjtl                                      1/1     Running   0          70m
w3admin@openshiftnstaller:~/openshift_install_dir$ oc get clusteroperator image-registry
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE
image-registry   4.5.13    True        False         False      72m
```

After running this command, if the above is shown, then this step can be skipped

15. To complete and verify the installation,
    a. check the cluster operator status

```
oc get clusteroperator
```

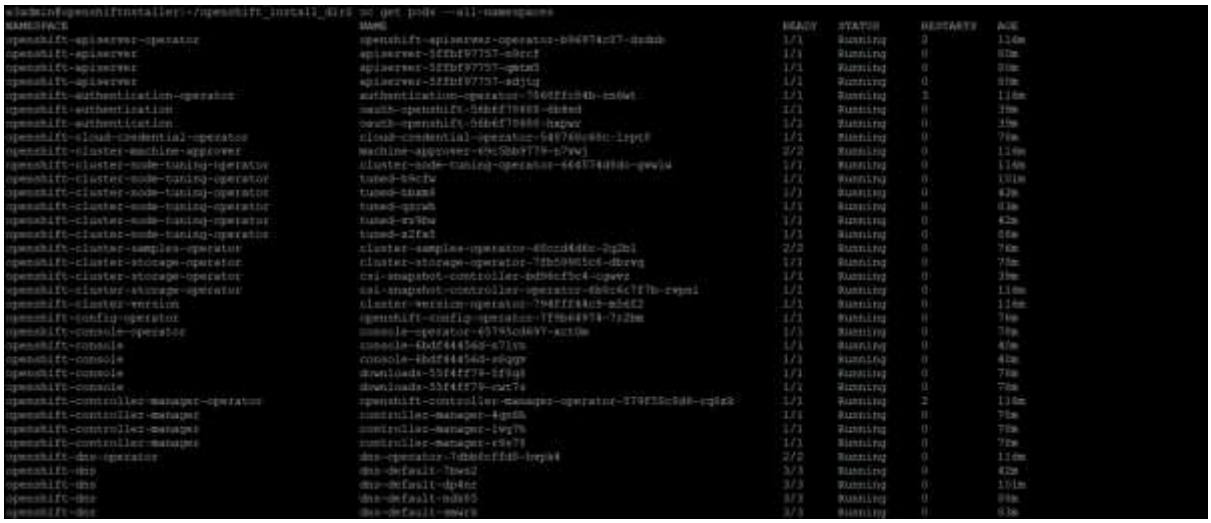
```
Every 5.0s: oc get clusteroperators

NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE
authentication                             4.5.13    True        False         False      32m
cloud-credential                           4.5.13    True        False         False      115m
cluster-autoscaler                         4.5.13    True        False         False      74m
config-operator                            4.5.13    True        False         False      74m
console                                    4.5.13    True        False         False      36m
csi-snapshot-controller                    4.5.13    True        False         False      37m
dns                                        4.5.13    True        False         False      99m
etcd                                       4.5.13    True        False         False      81m
image-registry                             4.5.13    True        False         False      76m
ingress                                    4.5.13    True        False         False      38m
insights                                   4.5.13    True        False         False      75m
kube-apiserver                             4.5.13    True        False         False      79m
kube-controller-manager                    4.5.13    True        False         False      98m
kube-scheduler                             4.5.13    True        False         False      98m
kube-storage-version-migrator              4.5.13    True        False         False      38m
machine-api                                4.5.13    True        False         False      76m
machine-approver                           4.5.13    True        False         False      78m
machine-config                             4.5.13    True        False         False      76m
marketplace                                4.5.13    True        False         False      75m
monitoring                                 4.5.13    True        False         False      37m
network                                    4.5.13    True        False         False      101m
node-tuning                                4.5.13    True        False         False      100m
openshift-apiserver                        4.5.13    True        False         False      75m
openshift-controller-manager               4.5.13    True        False         False      76m
openshift-samples                          4.5.13    True        False         False      74m
operator-lifecycle-manager                 4.5.13    True        False         False      99m
operator-lifecycle-manager-catalog         4.5.13    True        False         False      99m
operator-lifecycle-manager-packageserver   4.5.13    True        False         False      75m
service-ca                                 4.5.13    True        False         False      100m
storage                                    4.5.13    True        False         False      76m
```

Everything should be available

    b. View a list of all Pods

```
oc get pods --all-namespaces
```



16. kubeadmin password can be obtained from /auth/kubeadmin-password file on the installer server. Make a note of it. It will be required to log into the openshift GUI

17. Log into the OpenShift GUI:

    https://console-openshift-console.apps.openshift4.w3internal.com

    user: kubeadmin

    pwd: see above step

Useful Information

- **INFO** To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/w3admin/openshift_install_dir/auth/kubeconfig'

- When the ignition files are created, the OpenShift installer automatically generates certificates, which are used by the nodes. These certificates are only valid for 24hrs. If the install is not completed in this time, the master and worker nodes will through certificate errors in the logs and will not join the cluster. To fix this, recreate the ignition files and repeat the install instructions.
  Certificate expiry can be seen by running

  ```
  echo | openssl s_client -connect
  api.openshift4.w3internal.com:6443 | openssl x509 -noout -text
  ```

- If re-installing then ensure that the complete directory structure is deleted, since the previous config is stored in a hidden file ".openshift_install_state.json".

Appendices

Appendix A – Configuration YAML files

install-config.yaml

```
apiVersion: v1
baseDomain: w3internal.com
compute:
 - hyperthreading: Enabled
   name: worker
   replicas: 0
controlPlane:
  hyperthreading: Enabled
```

```
  name: master
  replicas: 3
metadata:
  name: openshift4
networking:
  clusterNetworks:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: <PULL SECRET>
sshKey: <SSH KEY>
```