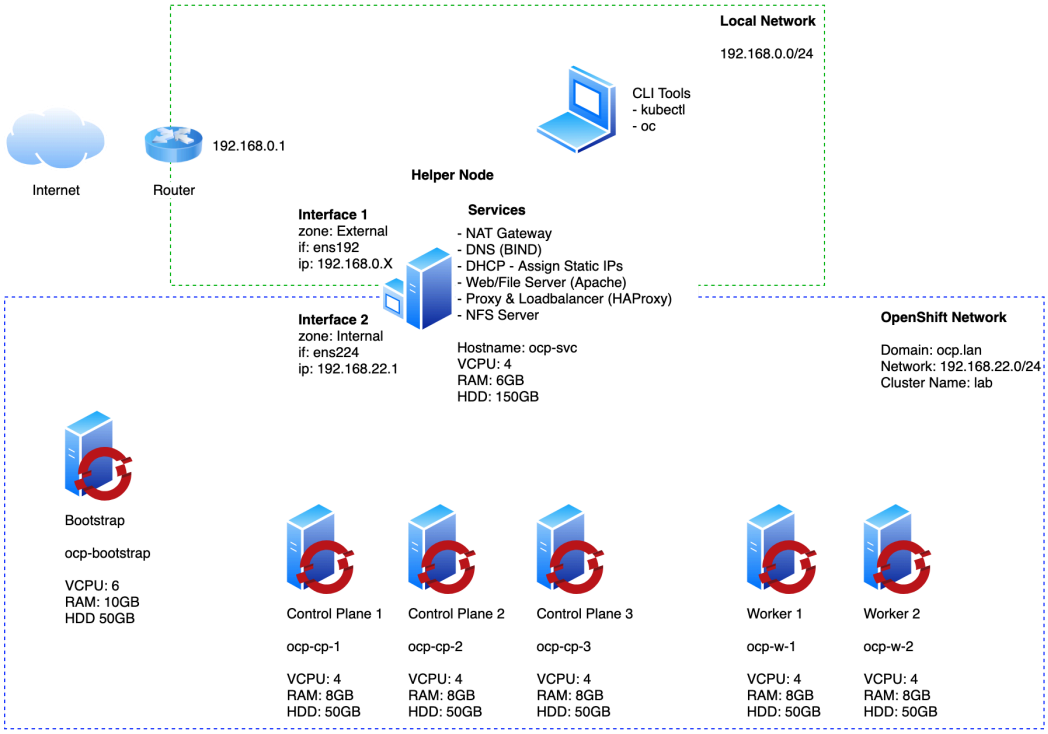
OpenShift 4 Bare Metal Install - User Provisioned Infrastructure (UPI)

Architecture Diagram



Download Software

1. Download CentOS 8 x86_64 image
2. Login to RedHat OpenShift Cluster Manager
3. Select 'Create Cluster' from the 'Clusters' navigation menu
4. Select 'RedHat OpenShift Container Platform'
5. Select 'Run on Bare Metal'
6. Download the following files:
    - Openshift Installer for Linux
    - Pull secret
    - Command Line Interface for Linux and your workstations OS
    - Red Hat Enterprise Linux CoreOS (RHCOS)

- rhcos-X.X.X-x86_64-metal.x86_64.raw.gz
- rhcos-X.X.X-x86_64-installer.x86_64.iso (or rhcos-X.X.X-x86_64-live.x86_64.iso for newer versions)

Prepare the 'Bare Metal' environment

VMware ESXi used in this guide

1. Copy the CentOS 8 iso to an ESXi datastore
2. Create a new Port Group called 'OCP' under Networking
    - (In case of VirtualBox choose "Internal Network" when creating each VM and give it the same name. ocp for instance)
    - (In case of ProxMox you may use the same network bridge and choose a specific VLAN tag. 50 for instance)
3. Create 3 Control Plane virtual machines with minimum settings:
    - Name: ocp-cp-# (Example ocp-cp-1)
    - 4vcpu
    - 8GB RAM
    - 50GB HDD
    - NIC connected to the OCP network
    - Load the rhcos-X.X.X-x86_64-installer.x86_64.iso image into the CD/DVD drive
4. Create 2 Worker virtual machines (or more if you want) with minimum settings:
    - Name: ocp-w-# (Example ocp-w-1)
    - 4vcpu
    - 8GB RAM
    - 50GB HDD
    - NIC connected to the OCP network
    - Load the rhcos-X.X.X-x86_64-installer.x86_64.iso image into the CD/DVD drive
5. Create a Bootstrap virtual machine (this vm will be deleted once installation completes) with minimum settings:
    - Name: ocp-boostrap
    - 4vcpu
    - 8GB RAM
    - 50GB HDD
    - NIC connected to the OCP network

    - Load the rhcos-X.X.X-x86_64-installer.x86_64.iso image into the CD/DVD drive
6. Create a Services virtual machine with minimum settings:
    - Name: ocp-svc
    - 4vcpu
    - 4GB RAM
    - 120GB HDD
    - NIC1 connected to the VM Network (LAN)
    - NIC2 connected to the OCP network
    - Load the CentOS_8.iso image into the CD/DVD drive
7. Boot all virtual machines so they each are assigned a MAC address
8. Shut down all virtual machines except for 'ocp-svc'
9. Use the VMware ESXi dashboard to record the MAC address of each vm, these will be used later to set static IPs

Configure Environmental Services

1. Install CentOS8 on the ocp-svc host
    - Remove the home dir partition and assign all free storage to '/'
    - Optionally you can install the 'Guest Tools' package to have monitoring and reporting in the VMware ESXi dashboard
    - Enable the LAN NIC only to obtain a DHCP address from the LAN network and make note of the IP address (ocp-svc_IP_address) assigned to the vm
2. Boot the ocp-svc VM
3. Move the files downloaded from the RedHat Cluster Manager site to the ocp-svc node
4. `scp ~/Downloads/openshift-install-linux.tar.gz ~/Downloads/openshift-client-linux.tar.gz ~/Downloads/rhcos-metal.x86_64.raw.gz root@{ocp-svc_IP_address}:/root/`
5. SSH to the ocp-svc vm
6. `ssh root@{ocp-svc_IP_address}`
7. Extract Client tools and copy them to `/usr/local/bin`

```
tar xvf openshift-client-linux.tar.gz
```

8. `mv oc kubectl /usr/local/bin`
9. Confirm Client Tools are working

```
kubectl version
```

10. `oc version`
11. Extract the OpenShift Installer
12. `tar xvf openshift-install-linux.tar.gz`
13. Update CentOS so we get the latest packages for each of the services we are about to install
14. `dnf update`
15. Install Git
16. `dnf install git -y`
17. Download config files for each of the services
18. `git clone https://github.com/ryanhay/ocp4-metal-install`
19. OPTIONAL: Create a file '~/.vimrc' and paste the following (this helps with editing in vim, particularly yaml files):

```
cat <<EOT >> ~/.vimrc
syntax on
set nu et ai sts=0 ts=2 sw=2 list hls
```

20. `EOT`
21. Update the preferred editor

```
export OC_EDITOR="vim"
```

22. `export KUBE_EDITOR="vim"`
23. Set a Static IP for OCP network interface `nmtui-edit ens224` or edit `/etc/sysconfig/network-scripts/ifcfg-ens224`
    - Address: 192.168.22.1
    - DNS Server: 127.0.0.1
    - Search domain: ocp.lan
    - Never use this network for default route
    - Automatically connect
24. If changes arent applied automatically you can bounce the NIC with `nmcli connection down ens224` and `nmcli connection up ens224`
25. Setup firewalld
    Create internal and external zones

```
nmcli connection modify ens224 connection.zone internal
```

26. `nmcli connection modify ens192 connection.zone external`
27. View zones:
28. `firewall-cmd --get-active-zones`
29. Set masquerading (source-nat) on the both zones.
    So to give a quick example of source-nat - for packets leaving the external

interface, which in this case is ens192 - after they have been routed they will have their source address altered to the interface address of ens192 so that return packets can find their way back to this interface where the reverse will happen.

```
firewall-cmd --zone=external --add-masquerade --permanent
```

30. `firewall-cmd --zone=internal --add-masquerade --permanent`
31. Reload firewall config
32. `firewall-cmd --reload`
33. Check the current settings of each zone

```
firewall-cmd --list-all --zone=internal
```

34. `firewall-cmd --list-all --zone=external`
35. When masquerading is enabled so is ip forwarding which basically makes this host a router. Check:
36. `cat /proc/sys/net/ipv4/ip_forward`
37. Install and configure BIND DNS
    Install
38. `dnf install bind bind-utils -y`
39. Apply configuration

```
\cp ~/ocp4-metal-install/dns/named.conf /etc/named.conf
```

40. `cp -R ~/ocp4-metal-install/dns/zones /etc/named/`
41. Configure the firewall for DNS

```
firewall-cmd --add-port=53/udp --zone=internal --permanent
# for OCP 4.9 and later 53/tcp is required
firewall-cmd --add-port=53/tcp --zone=internal --permanent
```

42. `firewall-cmd --reload`
43. Enable and start the service

```
systemctl enable named
systemctl start named
```

44. `systemctl status named`
45. At the moment DNS will still be pointing to the LAN DNS server. You can see this by testing with `dig ocp.lan`.
    Change the LAN nic (ens192) to use 127.0.0.1 for DNS AND ensure `Ignore automatically Obtained DNS parameters` is ticked
46. `nmtui-edit ens192`
47. Restart Network Manager

48. `systemctl restart NetworkManager`
49. Confirm dig now sees the correct DNS results by using the DNS Server running locally

```
dig ocp.lan
# The following should return the answer ocp-bootstrap.lab.ocp.lan from the
local server
```

50. `dig -x 192.168.22.200`
51. Install & configure DHCP
    Install the DHCP Server
52. `dnf install dhcp-server -y`
53. Edit dhcpd.conf from the cloned git repo to have the correct mac address for each host and copy the conf file to the correct location for the DHCP service to use
54. `\cp ~/ocp4-metal-install/dhcpd.conf /etc/dhcp/dhcpd.conf`
55. Configure the Firewall

```
firewall-cmd --add-service=dhcp --zone=internal --permanent
```

56. `firewall-cmd --reload`
57. Enable and start the service

```
systemctl enable dhcpd
systemctl start dhcpd
```

58. `systemctl status dhcpd`
59. Install & configure Apache Web Server
    Install Apache
60. `dnf install httpd -y`
61. Change default listen port to 8080 in httpd.conf
62. `sed -i 's/Listen 80/Listen 0.0.0.0:8080/' /etc/httpd/conf/httpd.conf`
63. Configure the firewall for Web Server traffic

```
firewall-cmd --add-port=8080/tcp --zone=internal --permanent
```

64. `firewall-cmd --reload`
65. Enable and start the service

```
systemctl enable httpd
systemctl start httpd
```

66. `systemctl status httpd`
67. Making a GET request to localhost on port 8080 should now return the default Apache webpage

68. `curl localhost:8080`
69. Install & configure HAProxy
    Install HAProxy
70. `dnf install haproxy -y`
71. Copy HAProxy config
72. `\cp ~/ocp4-metal-install/haproxy.cfg /etc/haproxy/haproxy.cfg`
73. Configure the Firewall
    Note: Opening port 9000 in the external zone allows access to HAProxy stats that are useful for monitoring and troubleshooting. The UI can be accessed at: `http://{ocp-svc_IP_address}:9000/stats`

```
firewall-cmd --add-port=6443/tcp --zone=internal --permanent # kube-api-server
on control plane nodes
firewall-cmd --add-port=6443/tcp --zone=external --permanent # kube-api-server
on control plane nodes
firewall-cmd --add-port=22623/tcp --zone=internal --permanent # machine-config
server
firewall-cmd --add-service=http --zone=internal --permanent # web services
hosted on worker nodes
firewall-cmd --add-service=http --zone=external --permanent # web services
hosted on worker nodes
firewall-cmd --add-service=https --zone=internal --permanent # web services
hosted on worker nodes
firewall-cmd --add-service=https --zone=external --permanent # web services
hosted on worker nodes
firewall-cmd --add-port=9000/tcp --zone=external --permanent # HAProxy Stats
```

74. `firewall-cmd --reload`
75. Enable and start the service

```
setsebool -P haproxy_connect_any 1 # SELinux name_bind access
systemctl enable haproxy
systemctl start haproxy
```

76. `systemctl status haproxy`
77. Install and configure NFS for the OpenShift Registry. It is a requirement to provide storage for the Registry, emptyDir can be specified if necessary.
    Install NFS Server
78. `dnf install nfs-utils -y`
79. Create the Share
    Check available disk space and its location `df -h`

```
mkdir -p /shares/registry
chown -R nobody:nobody /shares/registry
```

80. `chmod -R 777 /shares/registry`
81. Export the Share

```
echo "/shares/registry
192.168.22.0/24(rw,sync,root_squash,no_subtree_check,no_wdelay)" > /etc/exports
```

82. `exportfs -rv`
83. Set Firewall rules:

```
firewall-cmd --zone=internal --add-service mountd --permanent
firewall-cmd --zone=internal --add-service rpc-bind --permanent
firewall-cmd --zone=internal --add-service nfs --permanent
```

84. `firewall-cmd --reload`
85. Enable and start the NFS related services

```
systemctl enable nfs-server rpcbind
```

86. `systemctl start nfs-server rpcbind nfs-mountd`

Generate and host install files

1. Generate an SSH key pair keeping all default options
2. `ssh-keygen`
3. Create an install directory
4. `mkdir ~/ocp-install`
5. Copy the install-config.yaml included in the clones repository to the install directory
6. `cp ~/ocp4-metal-install/install-config.yaml ~/ocp-install`
7. Update the install-config.yaml with your own pull-secret and ssh key.
    - Line 23 should contain the contents of your pull-secret.txt
    - Line 24 should contain the contents of your '~/.ssh/id_rsa.pub'
8. `vim ~/ocp-install/install-config.yaml`
9. Generate Kubernetes manifest files
10. `~/openshift-install create manifests --dir ~/ocp-install`
11. A warning is shown about making the control plane nodes schedulable. It is up to you if you want to run workloads on the Control Plane nodes. If you dont want to you can disable this with: `sed -i 's/mastersSchedulable: true/mastersSchedulable: false/' ~/ocp-install/manifests/cluster-scheduler-02-config.yml`. Make any other

custom changes you like to the core Kubernetes manifest files.

Generate the Ignition config and Kubernetes auth files

12. `~/openshift-install create ignition-configs --dir ~/ocp-install/`
13. Create a hosting directory to serve the configuration files for the OpenShift booting process
14. `mkdir /var/www/html/ocp4`
15. Copy all generated install files to the new web server directory
16. `cp -R ~/ocp-install/* /var/www/html/ocp4`
17. Move the Core OS image to the web server directory (later you need to type this path multiple times so it is a good idea to shorten the name)
18. `mv ~/rhcos-X.X.X-x86_64-metal.x86_64.raw.gz /var/www/html/ocp4/rhcos`
19. Change ownership and permissions of the web server directory

```
chcon -R -t httpd_sys_content_t /var/www/html/ocp4/
chown -R apache: /var/www/html/ocp4/
```

20. `chmod 755 /var/www/html/ocp4/`
21. Confirm you can see all files added to the `/var/www/html/ocp4/` dir through Apache
22. `curl localhost:8080/ocp4/`

Deploy OpenShift

1. Power on the ocp-bootstrap host and ocp-cp-# hosts and select 'Tab' to enter boot configuration. Enter the following configuration:

```
# Bootstrap Node - ocp-bootstrap
coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.22.1:8080/ocp4/rhcos
coreos.inst.insecure=yes
coreos.inst.ignition_url=http://192.168.22.1:8080/ocp4/bootstrap.ign

# Or if you waited for it boot, use the following command then just reboot
after it finishes and make sure you remove the attached .iso
```

2. `sudo coreos-installer install /dev/sda -u`

   `http://192.168.22.1:8080/ocp4/rhcos -I`

   `http://192.168.22.1:8080/ocp4/bootstrap.ign --insecure`

   `--insecure-ignition`

```
# Each of the Control Plane Nodes - ocp-cp-\#
```

```
coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.22.1:8080/ocp4/rhcos
coreos.inst.insecure=yes
coreos.inst.ignition_url=http://192.168.22.1:8080/ocp4/master.ign

# Or if you waited for it boot, use the following command then just reboot
after it finishes and make sure you remove the attached .iso
```

3. `sudo coreos-installer install /dev/sda -u http://192.168.22.1:8080/ocp4/rhcos -I http://192.168.22.1:8080/ocp4/master.ign --insecure --insecure-ignition`
4. Power on the ocp-w-# hosts and select 'Tab' to enter boot configuration. Enter the following configuration:

```
# Each of the Worker Nodes - ocp-w-\#
coreos.inst.install_dev=sda
coreos.inst.image_url=http://192.168.22.1:8080/ocp4/rhcos
coreos.inst.insecure=yes
coreos.inst.ignition_url=http://192.168.22.1:8080/ocp4/worker.ign

# Or if you waited for it boot, use the following command then just reboot
after it finishes and make sure you remove the attached .iso
```

5. `sudo coreos-installer install /dev/sda -u http://192.168.22.1:8080/ocp4/rhcos -I http://192.168.22.1:8080/ocp4/worker.ign --insecure --insecure-ignition`

Monitor the Bootstrap Process

1. You can monitor the bootstrap process from the ocp-svc host at different log levels (debug, error, info)
2. `~/openshift-install --dir ~/ocp-install wait-for bootstrap-complete --log-level=debug`
3. Once bootstrapping is complete the ocp-boostrap node can be removed

Remove the Bootstrap Node

1. Remove all references to the `ocp-bootstrap` host from the `/etc/haproxy/haproxy.cfg` file

```
# Two entries
vim /etc/haproxy/haproxy.cfg
# Restart HAProxy - If you are still watching HAProxy stats console you will
see that the ocp-boostrap host has been removed from the backends.
```

2. `systemctl reload haproxy`
3. The ocp-bootstrap host can now be safely shutdown and deleted from the VMware ESXi Console, the host is no longer required

Wait for installation to complete

IMPORTANT: if you set mastersSchedulable to false the worker nodes will need to be joined to the cluster to complete the installation. This is because the OpenShift Router will need to be scheduled on the worker nodes and it is a dependency for cluster operators such as ingress, console and authentication.

1. Collect the OpenShift Console address and kubeadmin credentials from the output of the install-complete event
2. `~/openshift-install --dir ~/ocp-install wait-for install-complete`
3. Continue to join the worker nodes to the cluster in a new tab whilst waiting for the above command to complete

Join Worker Nodes

1. Setup 'oc' and 'kubectl' clients on the ocp-svc machine

```
export KUBECONFIG=~/ocp-install/auth/kubeconfig
# Test auth by viewing cluster nodes
```

2. `oc get nodes`
3. View and approve pending CSRs
   Note: Once you approve the first set of CSRs additional 'kubelet-serving' CSRs will be created. These must be approved too. If you do not see pending requests wait until you do.

```
# View CSRs
oc get csr
# Approve all pending CSRs
oc get csr -o go-template='{{range .items}}{{if not
.status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate
approve
# Wait for kubelet-serving CSRs and approve them too with the same command
```

4. `oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve`
5. Watch and wait for the Worker Nodes to join the cluster and enter a 'Ready' status
   This can take 5-10 minutes
6. `watch -n5 oc get nodes`

Configure storage for the Image Registry

A Bare Metal cluster does not by default provide storage so the Image Registry Operator bootstraps itself as 'Removed' so the installer can complete. As the installation has now completed storage can be added for the Registry and the operator updated to a 'Managed' state.

1. Create the 'image-registry-storage' PVC by updating the Image Registry operator config by updating the management state to 'Managed' and adding 'pvc' and 'claim' keys in the storage key:
2. `oc edit configs.imageregistry.operator.openshift.io`
3. `managementState: Managed`

```
storage:
  pvc:
```

4. `    claim: # leave the claim blank`
5. Confirm the 'image-registry-storage' pvc has been created and is currently in a 'Pending' state
6. `oc get pvc -n openshift-image-registry`
7. Create the persistent volume for the 'image-registry-storage' pvc to bind to
8. `oc create -f ~/ocp4-metal-install/manifest/registry-pv.yaml`
9. After a short wait the 'image-registry-storage' pvc should now be bound
10. `oc get pvc -n openshift-image-registry`

Create the first Admin user

1. Apply the `oauth-htpasswd.yaml` file to the cluster
   This will create a user 'admin' with the password 'password'. To set a different

username and password substitue the htpasswd key in the '~/ocp4-metal-install/manifest/oauth-htpasswd.yaml' file with the output of `htpasswd -n -B -b <username> <password>`

2. `oc apply -f ~/ocp4-metal-install/manifest/oauth-htpasswd.yaml`
3. Assign the new user (admin) admin permissions
4. `oc adm policy add-cluster-role-to-user cluster-admin admin`

Access the OpenShift Console

1. Wait for the 'console' Cluster Operator to become available
2. `oc get co`
3. Append the following to your local workstations `/etc/hosts` file:
   From your local workstation If you do not want to add an entry for each new service made available on OpenShift you can configure the ocp-svc DNS server to serve externally and create a wildcard entry for *.apps.lab.ocp.lan

```
# Open the hosts file
sudo vi /etc/hosts

# Append the following entries:
```

4. ```
   192.168.0.96 ocp-svc api.lab.ocp.lan
   console-openshift-console.apps.lab.ocp.lan
   oauth-openshift.apps.lab.ocp.lan
   downloads-openshift-console.apps.lab.ocp.lan
   alertmanager-main-openshift-monitoring.apps.lab.ocp.lan
   grafana-openshift-monitoring.apps.lab.ocp.lan
   prometheus-k8s-openshift-monitoring.apps.lab.ocp.lan
   thanos-querier-openshift-monitoring.apps.lab.ocp.lan
   ```
5. Navigate to the OpenShift Console URL and log in as the 'admin' user
   You will get self signed certificate warnings that you can ignore If you need to login as kubeadmin and need to the password again you can retrieve it with: `cat ~/ocp-install/auth/kubeadmin-password`

Troubleshooting

1. You can collect logs from all cluster hosts by running the following command from the 'ocp-svc' host:
   ```

2.
```
./openshift-install gather bootstrap --dir ocp-install
--bootstrap=192.168.22.200 --master=192.168.22.201
--master=192.168.22.202 --master=192.168.22.203
```
3. Modify the role of the Control Plane Nodes
   If you would like to schedule workloads on the Control Plane nodes apply the 'worker' role by changing the value of 'mastersSchedulable' to true.
   If you do not want to schedule workloads on the Control Plane nodes remove the 'worker' role by changing the value of 'mastersSchedulable' to false.
   Remember depending on where you host your workloads you will have to update HAProxy to include or exclude the control plane nodes from the ingress backends.
4.
```
oc edit schedulers.config.openshift.io cluster
```