

IBM Power Virtual User Group

Stuart Cunliffe - IBM Systems Lab Services

European Power & Cognitive CTO

Twitter: @Stu Cunliffe

email: s_cunliffe@uk.ibm.com

Openshift Container Platform v4.3 on IBM Power

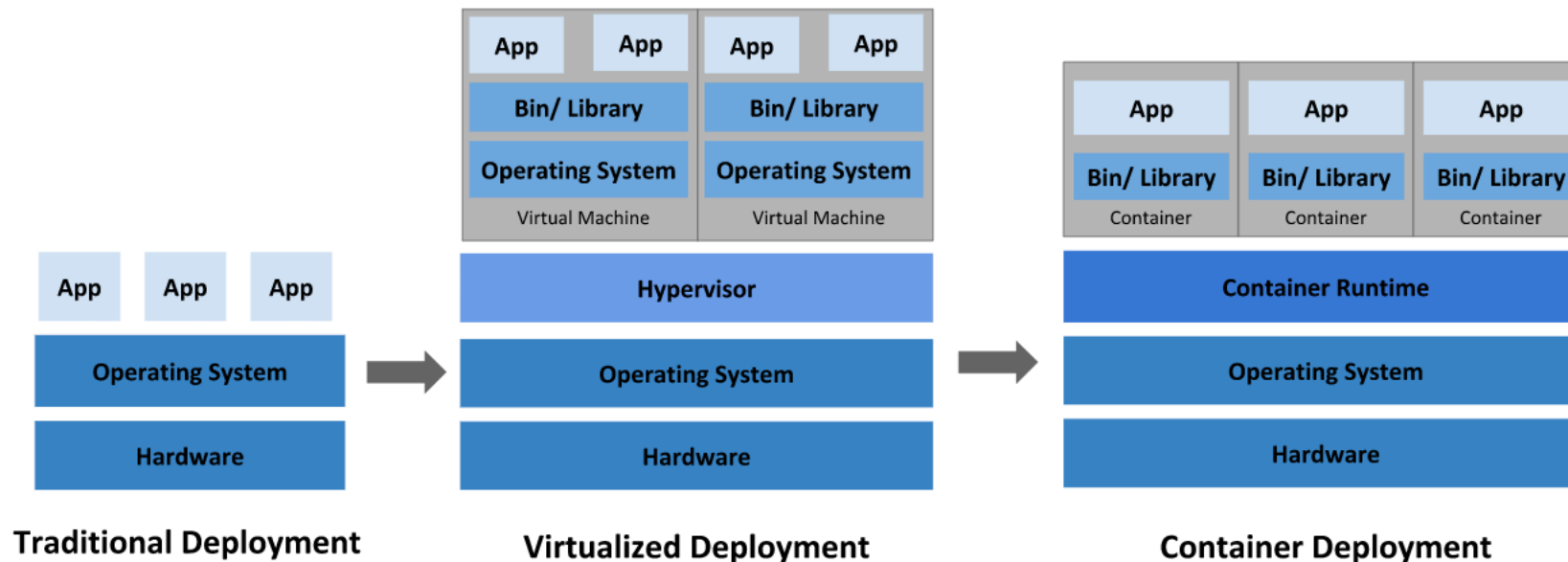
Agenda



- Overview of Openshift Container Platform (OCP)
- Differences between OCP v3.11 and v4.3
- Installing OCP on User Provisioned Infrastructure (UPI)
- Using OCP via the GUI and CLI
- Running workloads on OCP

Overview of OCP - containers

- Container runtime environments performs operating system level virtualization also known as containerization.
- Containers allow us to package an application along with all its dependencies such as libraries.
- Container runtimes such as Docker and CRI-O, allow us to create, deploy and run applications within containers.
- A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.



Containers

Pros

- Lightweight, fast, isolated
- Contains all the dependencies, libraries, binaries and config needed, easy to migrate
- Small - most containers are <100MB
- Consolidation - more workloads than VMs
- Upgrades - single container can be upgraded as opposed to a whole VM
- Allows developers to work on their microservice apps, build then and share them.
- The decoupling of applications from the environment they would normally run.

Cons

- Linux only
- Shares the OS kernel - potential vulnerabilities
- More items to monitor and manage

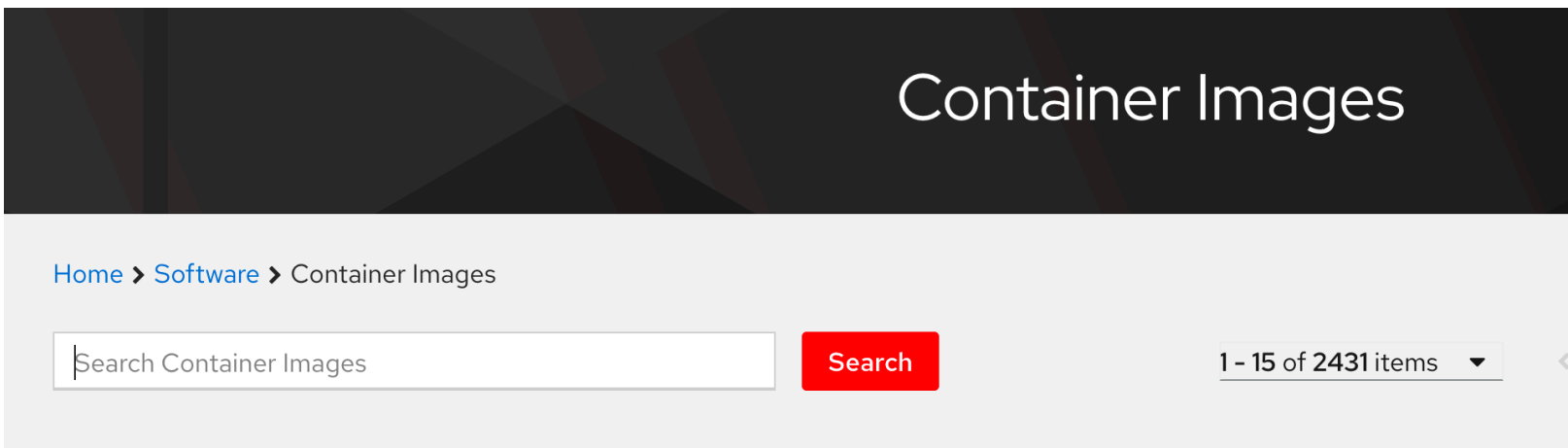
Container images

Images are stored in a number of locations depending on what they contain, licensing etc.:

- DockerHub (<https://hub.docker.com/>) - contains thousands of images for all architectures.



- Red Hat (<https://catalog.redhat.com/software/containers/search>) - supported and certified images.



Starting a new container



podman

We can simply start a new container by asking our container runtime:

```
host1# podman run -it --entrypoint /bin/bash centos -s
```

```
[root@eabe20b44a88 /]# cat /etc/*ease | grep PRETTY_NAME
```

```
PRETTY_NAME="CentOS Linux 8 (Core)"
```

```
[root@eabe20b44a88 /]# hostname
```

```
eabe20b44a88
```

Creating new container images

Process to manually update/create a new image:

1. Deploy a 'base' image to create a new container e.g. registry.redhat.io/centos
2. Login to that container and configure it as required:
 - a. Install new rpms
 - b. Add users
 - c. Copy files into the container
 - d. Create start script to fire up when container boots, etc...
3. Capture that container as a new image

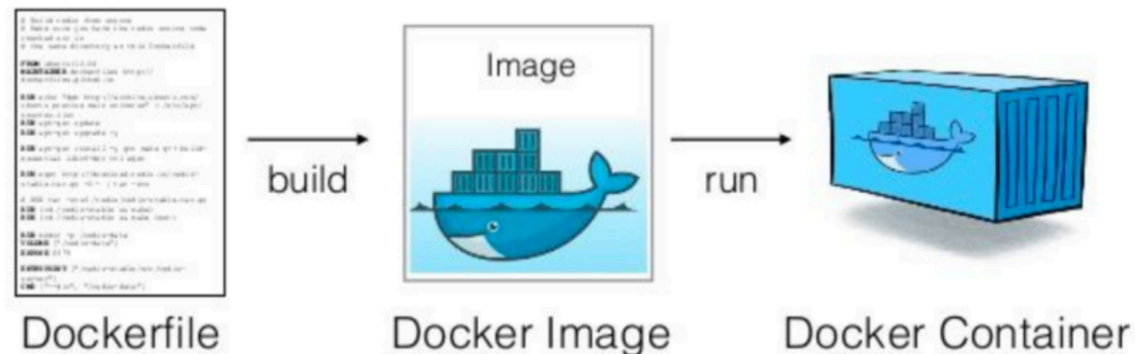
Issues with this method:

- Each time you want to make a change to the image, you have to repeat all these steps
- Prone to error if you get one of the steps wrong
- Not very agile

Creating new container images

Process to update/create a new image using Dockerfile:

1. Create a Dockerfile that uses the 'base' image.
2. Add the desired configuration to that Dockerfile:
 - a. Install new rpms
 - b. Add users
 - c. Copy files into the container
 - d. State startup command to run when container is built, etc..
3. Build a new image using that Dockerfile.



Creating new container images

Example Dockerfile:

```
FROM ubuntu
CMD /bin/bash
MAINTAINER Stu Cunliffe,UK s_cunliffe@uk.ibm.com
RUN apt-get update
RUN apt-get install -y npm
RUN mkdir -p /usr/src/node-red
WORKDIR /usr/src/node-red
RUN groupadd --force node-red
RUN useradd --home /usr/src/node-red --gid node-red node-red
RUN chown -R node-red:node-red /usr/src/node-red
USER node-red
RUN npm install node-red
EXPOSE 1880/tcp
COPY package.json /usr/src/node-red/package.json
COPY flow-file.json /usr/src/node-red/.node-red/node-red
CMD npm start node-red
```

We can then build the image by simply calling:

```
# docker build . -t docker.io/cunlifs/node-red-iss:v0.8
```

Combining Dockerfile and GitHub

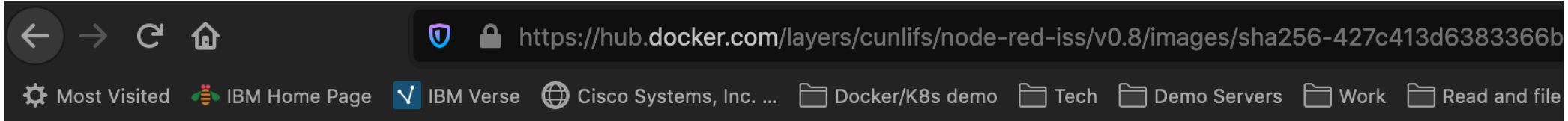
Example Dockerfile in GitHub:

```
FROM ubuntu
CMD /bin/bash
MAINTAINER Stu Cunliffe,UK s_cunliffe@uk.ibm.com
RUN apt-get update
RUN apt-get install -y npm
RUN mkdir -p /usr/src/node-red
WORKDIR /usr/src/node-red
RUN groupadd --force node-red
RUN useradd --home /usr/src/node-red --gid node-red node-red
RUN chown -R node-red:node-red /usr/src/node-red
USER node-red
RUN npm install node-red
EXPOSE 1880/tcp
COPY package.json /usr/src/node-red/package.json
COPY flow-file.json /usr/src/node-red/.node-red/node-red
CMD npm start node-red
```

We can then build the image by simply calling the github source:

```
# docker build github.com/cunlifs/node-red-iss -t docker.io/cunlifs/node-red-iss:v0.8
```

Combining Dockerfile and GitHub



cunlifs/node-red-iss:v0.8

DIGEST: sha256:427c413d6383366b4332c98d7b3804f2274d7340e31dbc0c9c77f793feb2b0c0

OS/ARCH

linux/ppc64le

SIZE

167.78 MB

LAST PUSHED

6 months ago by [cunlifs](#)

IMAGE HISTORY ?

1	ADD file ... in /	28.99 MB
2	/bin/sh -c [-z "\$(apt-get	34.38 KB
3	/bin/sh -c set -xe &&	849 B
4	/bin/sh -c mkdir -p /run/systemd	188 B
5	CMD ["/bin/bash"]	0 B
6	CMD ["/bin/sh" "-c" "/bin/bash"]	0 B
7	MAINTAINER Stuart Cunliffe,UK s_cunliffe@uk.ib...	0 B
8	/bin/sh -c apt-get update	17.91 MB
9	/bin/sh -c apt-get install -y	100.59 MB

Command

```
/bin/sh -c apt-get install -y npm
```

Containers are great, but.....

..what about running them in a large development or production environment?

We have to consider a number of issues:

- We might end up with hundreds or thousands of containers
- How do we manage them all
- How do we allow them to communicate with each other
- What if it fails, how would monitor it and restart it?
- What if we want to scale and add more instances?
- What if we want to add storage?
- How do we upgrade it with little or no impact to the users?

Overview of Kubernetes (k8s)

- Kubernetes is the orchestration layer that manages containers across a group of physical servers or VMs.
- Kubernetes is specifically designed to manage the ephemeral nature of thousands of containers by:
 - deploying containerised applications
 - scaling up due to demand
 - scaling down or terminating
 - version control
 - internal and external container communications
 - storage creation and attachment
 - monitoring - what to do when a host server/VM fails
- Kubernetes runs just as well on traditional on-premises infrastructure stacks as it does for third-party service providers and public cloud environments.
- Offerings include: IBM Kubernetes Service (IKS), Azure Kubernetes Service (AKS), Amazon Elastic Kubernetes Service (EKS)¹³ and Red Hat OpenShift.



Kubernetes terminology

Nodes - VM/Server that runs a workload

Pods - Collection of containers, normally just a single container per pod

Projects - Namespaces/Reserved space

Services - Collection of pods, exposed as an endpoint

Routes - exposes services for external comms

Deployments - creates the pods and replica sets in single configuration

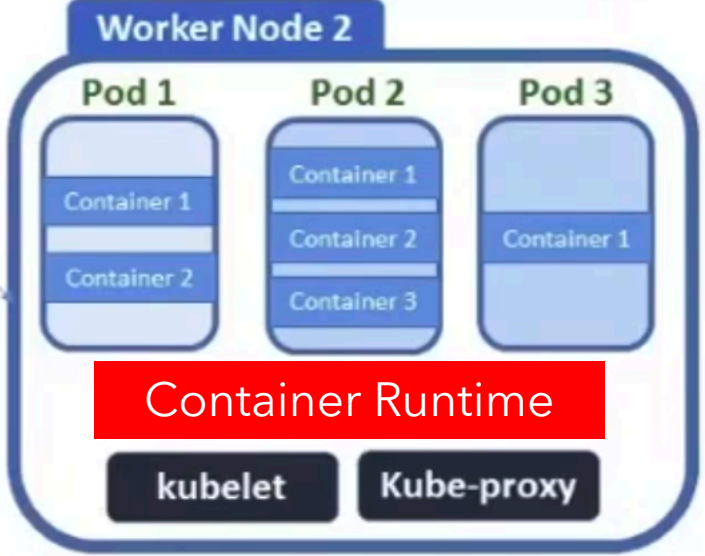
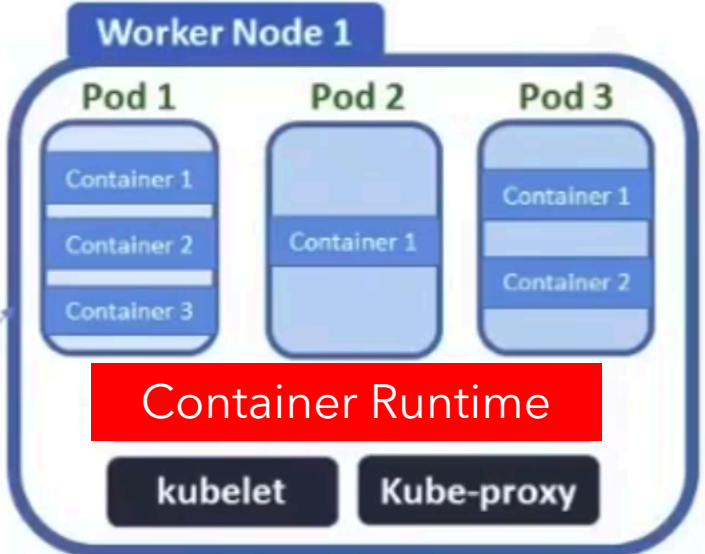
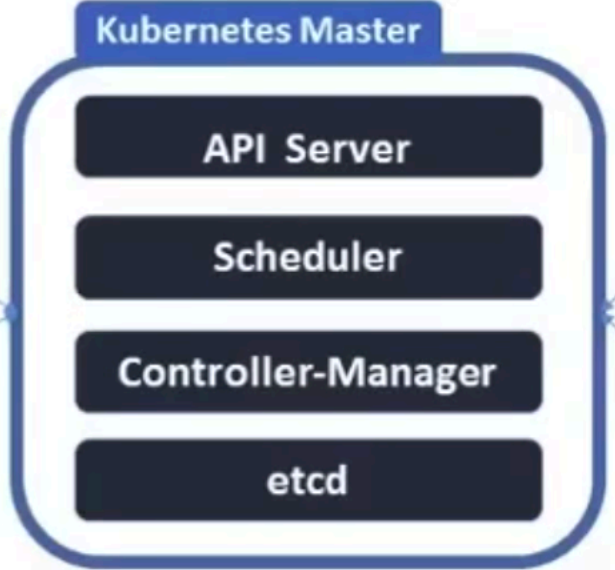
Overview of Kubernetes (k8s)

KUBERNETES ARCHITECTURE

User Interface



kubectl



Red Hat OpenShift includes everything needed for hybrid cloud, enterprise containers and Kubernetes development and deployment

Tested and integrated components include:

- Enterprise Linux operating system
- Container runtime
- Networking
- Monitoring
- Container registry
- Authentication

Red Hat OpenShift is available in 3 flavours:

Red Hat OpenShift Container Platform

Deploy and operate OpenShift in on premise or in public cloud on physical or virtual servers managed by you.

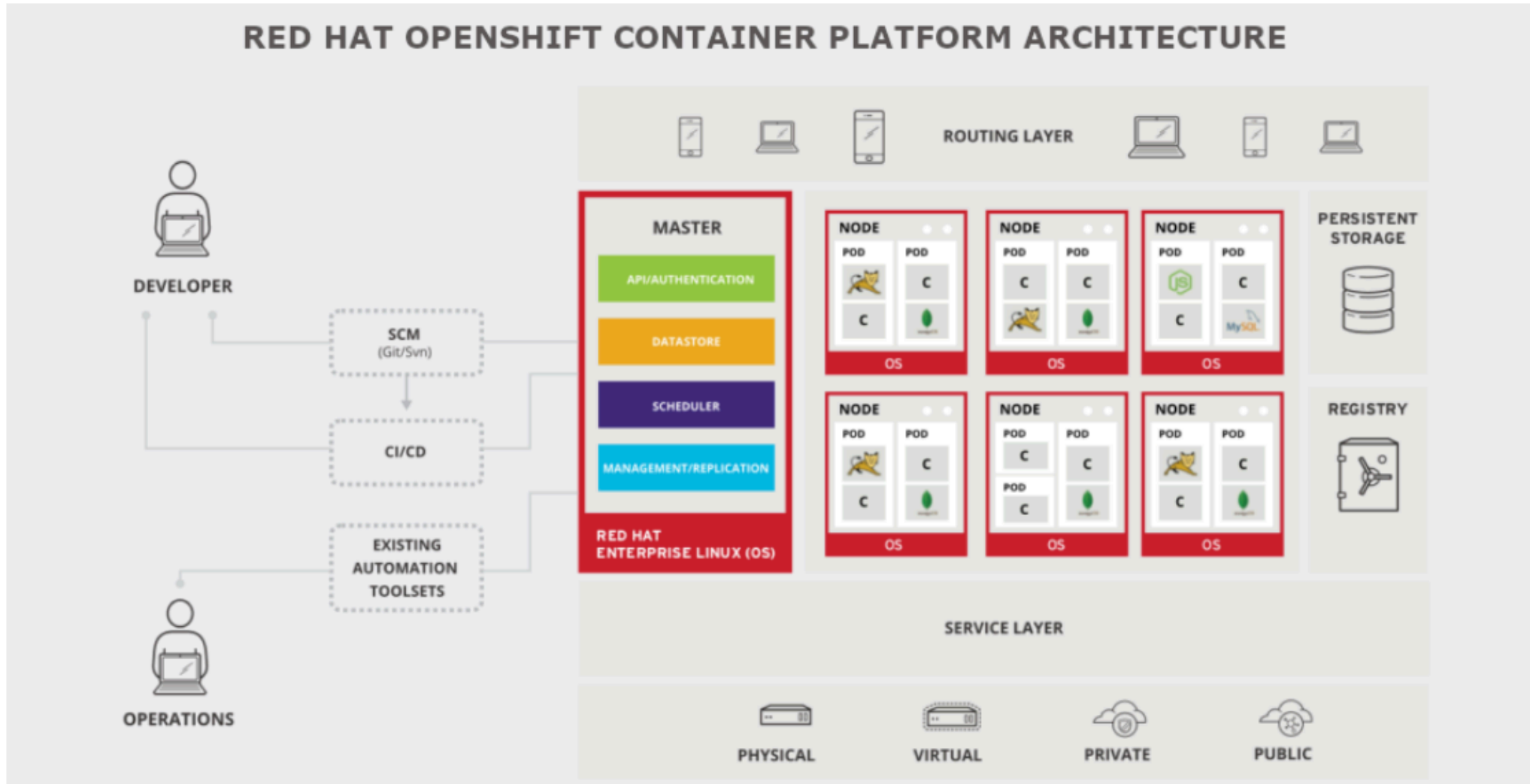
Red Hat OpenShift Dedicated

Develop and manage containerized applications with your own OpenShift cluster, managed and operated by Red Hat and deployed in Public Cloud.

Red Hat OpenShift Online

Quickly build, launch, and host applications in the public cloud, operated and supported by Red Hat.

OpenShift Container Platform



NOTE: OCP v4 User Provisioned Infrastructure on IBM Power - all nodes inc. Masters are Red Hat CoreOS

What's new in OCP v4 (on IBM Power)

Red Hat Enterprise Linux CoreOS:

Master and worker nodes all now run RHCOS

Kubernetes Operators:

Method to deploy, package and manage K8s applications.

Cluster Upgrades:

Simple interface to keep track of updates and applying

Cluster Installation:

Installer-Provisioned Infrastructure and User-Provisioned Infrastructure

Red Hat Enterprise Linux CoreOS (RHCOS)

Represents the next generation of single-purpose container operating system technology, and created by the same teams that brought you RHEL atomic and CoreOS Container Linux

Based on RHEL - s/w is in RPM packages and services managed by the system init system

Controlled immutability - managed more tightly than a RHEL installation, you can only modify a few settings.

Uses the CRI-O container engine instead of Docker container engine.

Container tools such as podman and skopeo are part of RHCOS

Updates are delivered as container images as part of the OCP upgrade process. The node uses a rolling update method to ensure minimal cluster impact.

These updates are controlled in OCP via the Machine Config Operator.

RHEL and RHEL CoreOS

RED HAT® ENTERPRISE LINUX®

General Purpose OS

BENEFITS

- 10+ year enterprise life cycle
- Industry standard security
- High performance on any infrastructure
- Customizable and compatible with wide ecosystem of partner solutions

WHEN TO USE

When customization and integration with additional solutions is required

RED HAT® ENTERPRISE LINUX CoreOS

Immutable container host

- Self-managing, over-the-air updates
- Immutable and tightly integrated with OpenShift
- Host isolation is enforced via Containers
- Optimized performance on popular infrastructure

When cloud-native, hands-free operations are a top priority

RHEL CoreOS - Immutable OS

Red Hat Enterprise Linux CoreOS is versioned with OpenShift

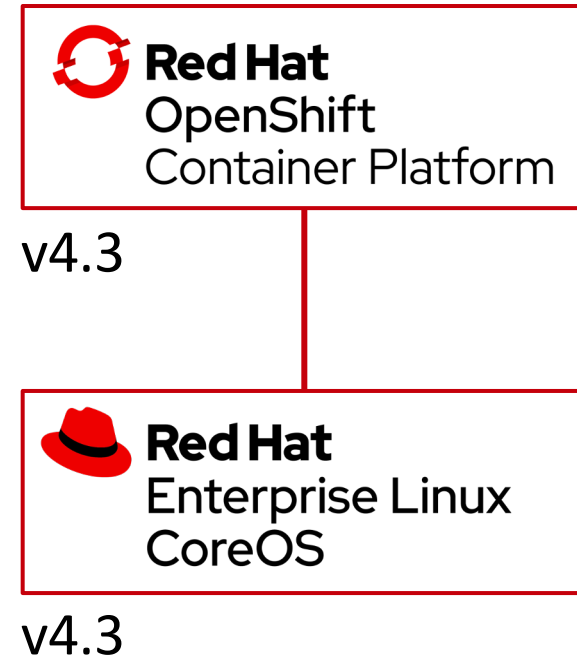
CoreOS is tested and shipped in conjunction with the platform.

Red Hat runs thousands of tests against these configurations.

Red Hat Enterprise Linux CoreOS is managed by the cluster

The Operating system is operated as part of the cluster, with the config for components managed by Machine Config Operator:

- CRI-O config
- Kubelet config
- Authorized registries
- SSH config

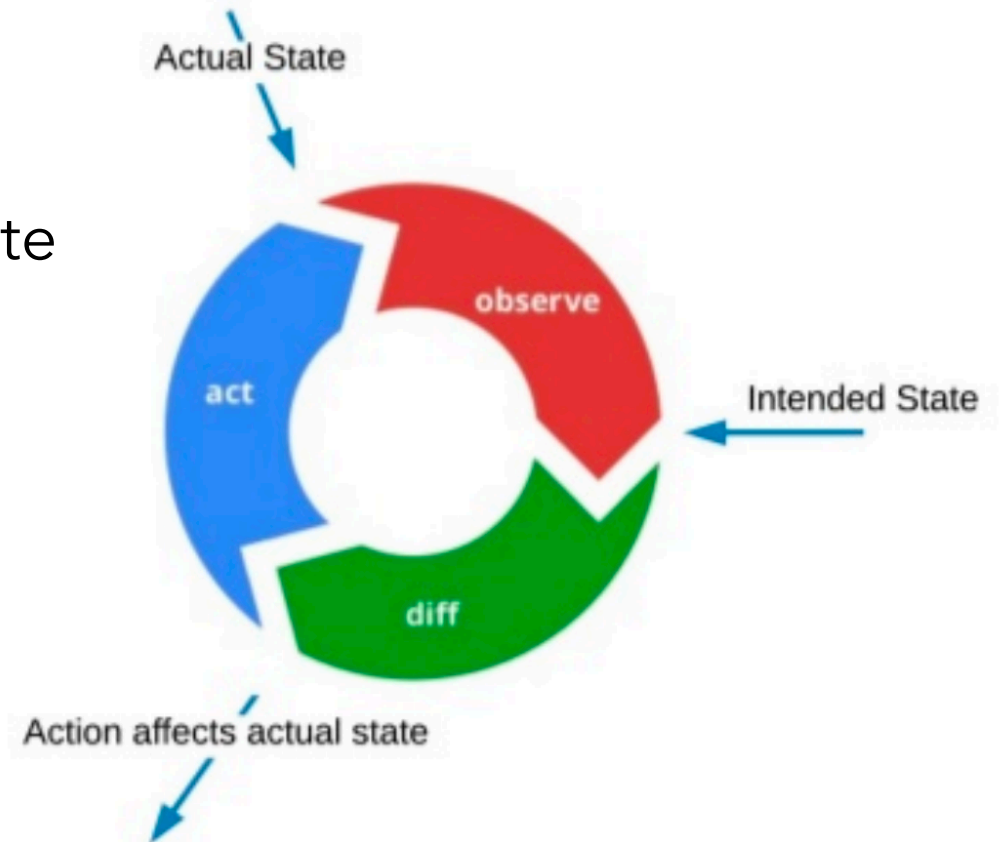


Controllers & Operators

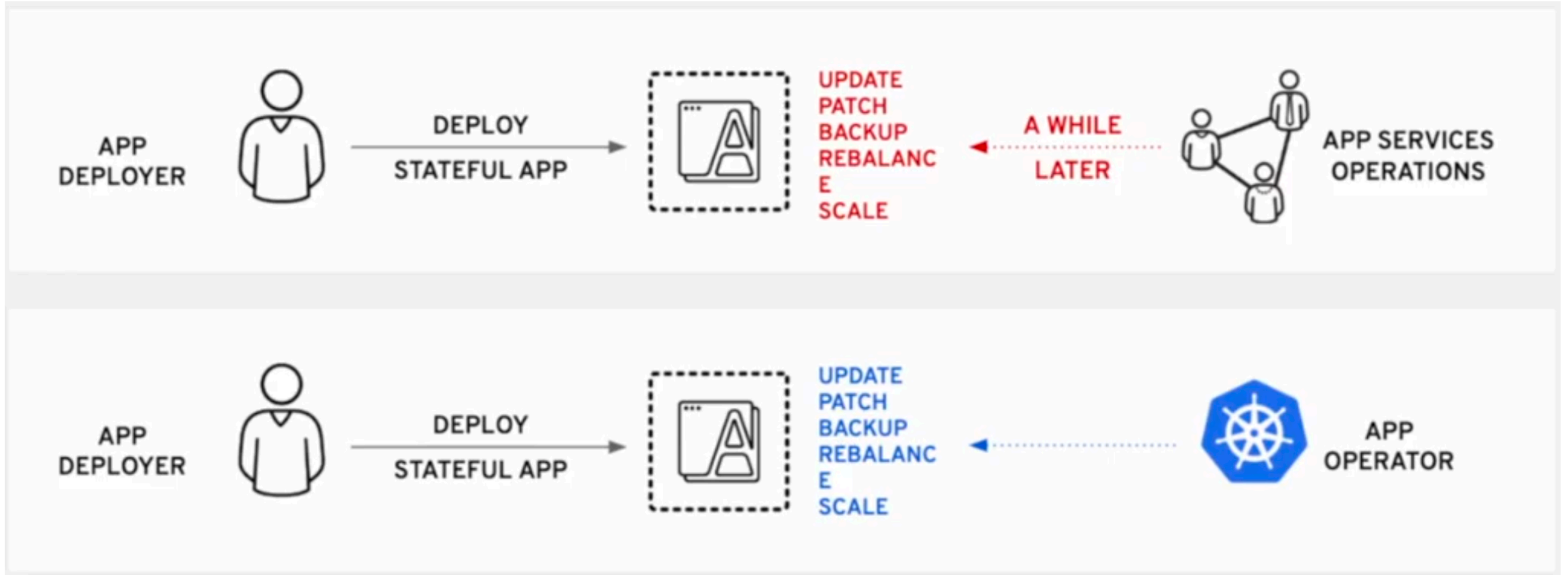
Controllers are a core part of Kubernetes.

- Run continuously on the Master Nodes
- Compare actual state with intended state
- Take action (if necessary) to reconcile the state

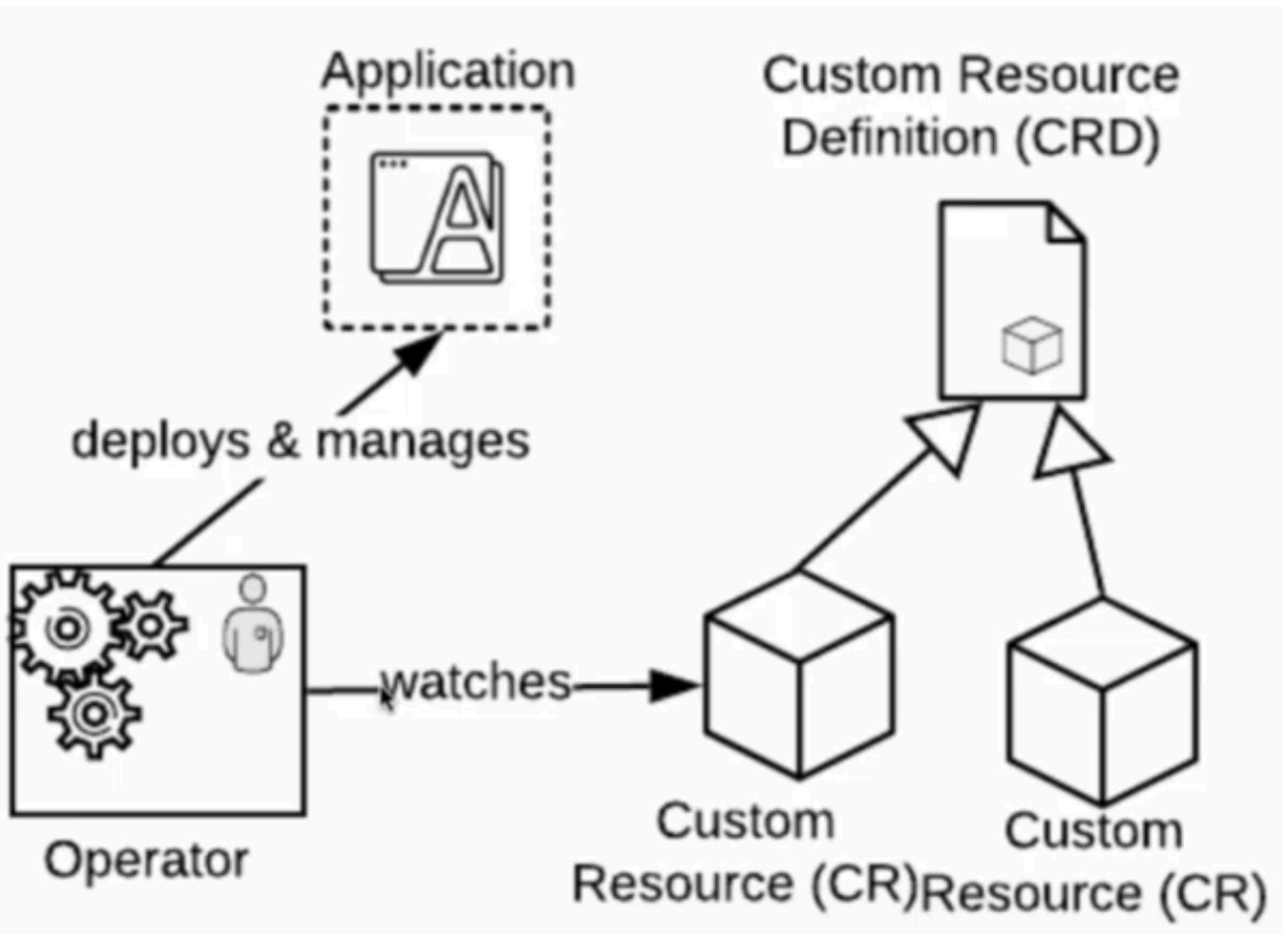
- Works on objects such as:
 - Pods
 - Services
 - ConfigMaps
 - PersistentVolumes



Use case for Operators



Operators and Custom Resources



- Custom Resource Definition (CRD) allows us to define our own object/resource that Kubernetes now knows about.
- Custom Resources (CR) can now be created and assigned the 'type' of resource defined in the CRD above.
- A custom controller (Operator) is needed to know how to handle the CR

Operators and Helm

Package and Install

Automated Day-2 Operations

Helm

Operator

Phase I

Phase II

Phase III

Phase IV

Phase V

Basic Install

Seamless Upgrades

Full Lifecycle

Deep Insights

Auto Pilot

Automated application provisioning and configuration management

Patch and minor version upgrades supported

App lifecycle, storage lifecycle (backup, failure recovery)

Metrics, alerts, log processing and workload analysis

Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning

Operator Hub

OperatorHub.io

Search OperatorHub...

Contribute ▾


Welcome to OperatorHub.io

OperatorHub.io is a new home for the Kubernetes community to share Operators. Find an existing Operator or list your own today.

CATEGORIES

- AI/Machine Learning
- Application Runtime
- Big Data
- Cloud Provider
- Database
- Developer Tools
- Integration & Delivery
- Logging & Tracing
- Monitoring
- Networking

139 ITEMS

VIEW  ▾ SORT A-Z ▾



Akka Cluster Operator
provided by Lightbend, Inc.

Run Akka Cluster applications
on Kubernetes.



**Altinity ClickHouse
Operator**
provided by Altinity

ClickHouse Operator manages
full lifecycle of ClickHouse



Anchore Engine Operator
provided by Anchore Inc.

Anchore Engine - container
image scanning service for
policy-based security, best-



Apache Spark Operator
provided by radanalytics.io

An operator for managing the
Apache Spark clusters and
intelligent applications that

Cluster/Platform Operators

OCP v4.3 contains a number of operators used to control the cluster itself (cluster operators):

```
[root@ocp43-0a5e-bastion ~]# oc get clusteroperators
```

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.3.23	True	False	False	30d
cloud-credential	4.3.23	True	False	False	30d
cluster-autoscaler	4.3.23	True	False	False	30d
console	4.3.23	True	False	False	23d
dns	4.3.23	True	False	False	30d
image-registry	4.3.23	True	False	False	6d18h
ingress	4.3.23	True	False	False	6d18h
insights	4.3.23	True	False	False	30d
kube-apiserver	4.3.23	True	False	False	30d
kube-controller-manager	4.3.23	True	False	False	30d
kube-scheduler	4.3.23	True	False	False	30d
machine-api	4.3.23	True	False	False	30d
machine-config	4.3.23	True	False	False	23d
marketplace	4.3.23	True	False	False	23d
monitoring	4.3.23	True	False	False	6d15h
network	4.3.23	True	False	False	30d
node-tuning	4.3.23	True	False	False	14d
openshift-apiserver	4.3.23	True	False	False	6d16h
openshift-controller-manager	4.3.23	True	False	False	30d
openshift-samples	4.3.23	True	False	False	23d
operator-lifecycle-manager	4.3.23	True	False	False	30d
operator-lifecycle-manager-catalog	4.3.23	True	False	False	30d
operator-lifecycle-manager-packageserver	4.3.23	True	False	False	13d
service-ca	4.3.23	True	False	False	30d
service-catalog-apiserver	4.3.23	True	False	False	20d
service-catalog-controller-manager	4.3.23	True	False	False	20d
storage	4.3.23	True	False	False	23d

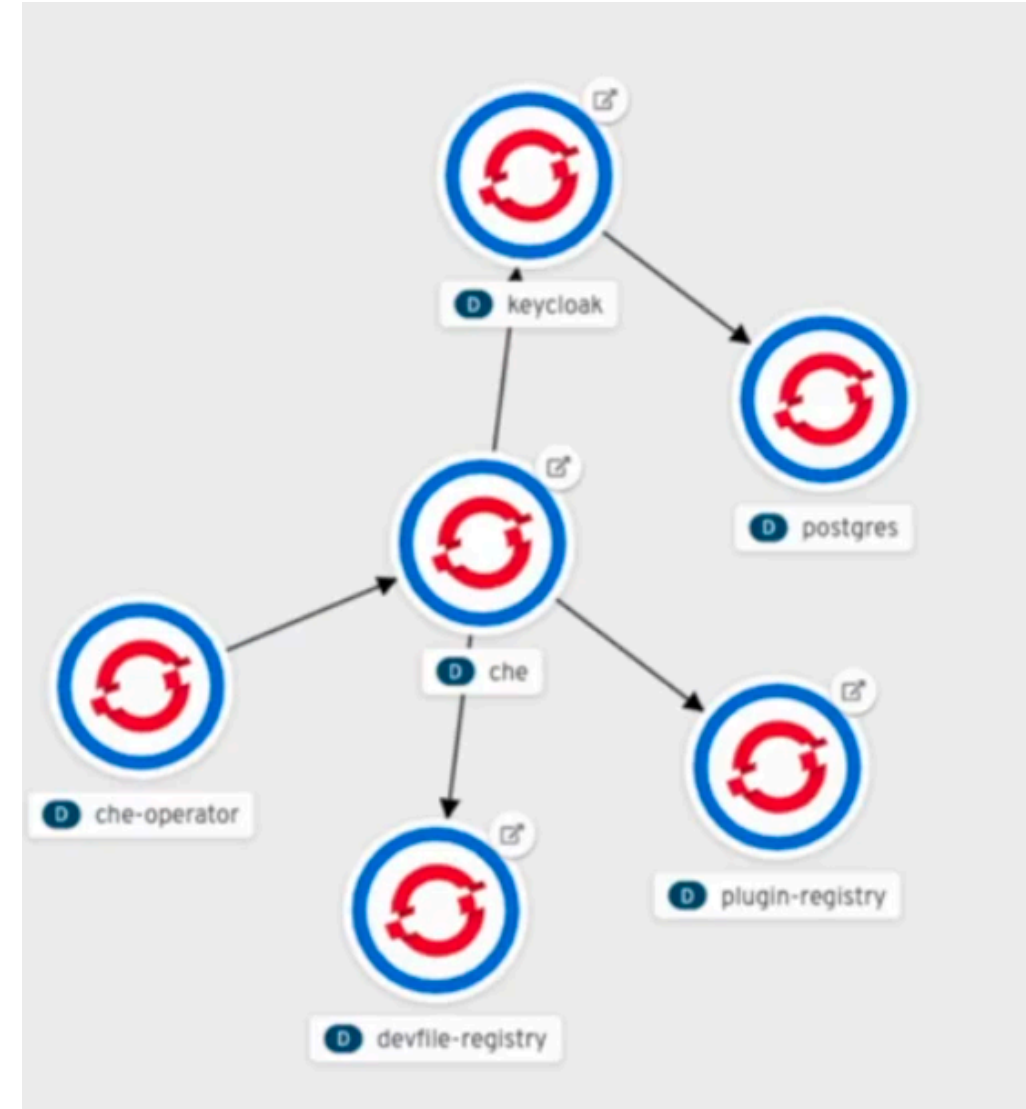
Operator - example

Project: che

Create Pod

6 Running 0 Pending 0 Terminating 0 CrashLoopBackOff 0 Completed 0 Failed 0 Unknown [Select All Filters](#)

Name ↑	Namespace ↓	Pod Labels ↓	Node ↓	Status ↓
che-9fb5d4f9f-ntvt2	NS che	app=che component=che pod-templa...=9fb5...	N ip-10-0-173-108.us-west-2.compute.internal	Running
che-operator-8d6b7ddcb-5ccnz	NS che	app=che-operator pod-tempi...=8d6b...	N ip-10-0-151-232.us-west-2.compute.internal	Running
devfile-registry-77cdd5dc64-89wsn	NS che	app=che compo...=devfile-re... pod-tempi...=77cdd...	N ip-10-0-151-232.us-west-2.compute.internal	Running
keycloak-79b97fcf47-tbcd	NS che	app=che component=keycloak pod-tempi...=79b9...	N ip-10-0-151-232.us-west-2.compute.internal	Running
plugin-registry-d7f77cb6-vxhw8	NS che	app=che compo...=plugin-re... pod-tempia...=d7f7...	N ip-10-0-151-232.us-west-2.compute.internal	Running
postgres-6d5dbdfb9f-wzbqt	NS che	app=che component=postgres pod-tempi...=6d5d...	N ip-10-0-151-232.us-west-2.compute.internal	Running



Installing OCP v4

Clusters > Create > OpenShift Container Platform

Install OpenShift Container Platform 4

Select an infrastructure provider



Run on Amazon Web Services



Run on Microsoft Azure



Run on Google Cloud Platform



Run on VMware vSphere



Run on Red Hat OpenStack



Run on Red Hat Virtualization



Run on Bare Metal

IBM Z
IBM LinuxONE™

Run on IBM Z



Run on Power

<https://cloud.redhat.com/openshift/install>

Installing OCP v4 : IPI vs UPI

(Installer-Provisioned Infrastructure vs User-Provisioned Infrastructure)

Clusters > Create > OpenShift Container Platform > Red Hat OpenStack Platform

Install OpenShift Container Platform 4

OpenStack: Select an installation type

★ Recommended



Installer-provisioned infrastructure

Deploy an OpenShift cluster on infrastructure that the installation program provisions and the cluster maintains.



User-provisioned infrastructure

Deploy an OpenShift cluster on infrastructure that you prepare and maintain.

<https://cloud.redhat.com/openshift/install/openstack>

Installing OCP v4 - UPI on Power - Step 1

Clusters > Create > OpenShift Container Platform > Power

Install OpenShift on Power with user-provisioned infrastructure

1

What you need to get started

OpenShift installer

Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

MacOS ▾

[Download installer](#)

[Developer Preview](#)

[Download pre-release builds](#)

Pull secret

Download or copy your pull secret. The install program will prompt you for your pull secret during installation.

[Download pull secret](#)

[Copy pull secret](#)

Command line interface

Download the OpenShift command-line tools and add them to your `PATH`.

MacOS ▾

[Download command-line tools](#)

When the installer is complete you will see the console URL and credentials for accessing your new cluster. A `kubeconfig` file will also be generated for you to use with the `oc` CLI tools you downloaded.

Red Hat Enterprise Linux CoreOS (RHCOS)

[Download RHCOS](#)

1a) Download the installer


1b) Pull the secret needed from Red Hat

1c) Download the client command line interface

1d) Download the Red Hat CoreOS

<https://cloud.redhat.com/openshift/install/power/user-provisioned>

Download install binary and client (1a & 1c)

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 openshift-client-linux-4.3.28.tar.gz	29-Jun-2020 01:59	24M	
 openshift-client-linux.tar.gz	29-Jun-2020 01:59	24M	
 openshift-client-mac-4.3.28.tar.gz	29-Jun-2020 02:36	27M	
 openshift-client-mac.tar.gz	29-Jun-2020 02:36	27M	
 openshift-client-windows-4.3.28.zip	29-Jun-2020 02:38	24M	
 openshift-client-windows.zip	29-Jun-2020 02:38	24M	
 openshift-install-linux-4.3.28.tar.gz	29-Jun-2020 02:22	69M	
 openshift-install-linux.tar.gz	29-Jun-2020 02:22	69M	
 openshift-install-mac-4.3.28.tar.gz	29-Jun-2020 03:11	84M	
 openshift-install-mac.tar.gz	29-Jun-2020 03:11	84M	

<https://mirror.openshift.com/pub/openshift-v4/ppc64le/clients/ocp/latest-4.3/>

Download the pull secret (1b)

Pull secret







Download or copy your pull secret. The install program will prompt you for your pull secret during installation.

Download pull secret

Copy pull secret

```
pull-secret-2
{"auths":{"cloud.openshift.com":
{"auth":"
"quay.io":
, "registry.connect.redhat.com":
{"auth":"
, "registry.redhat.io":
uk.ibm.com}}}]
```





Download Red Hat CoreOS image (1d)

	<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
	Parent Directory		-	
	rhcos-4.3.18-installer-initramfs.ppc64le.img	29-Apr-2020 15:46	68M	
	rhcos-4.3.18-installer-kernel-ppc64le	29-Apr-2020 15:46	23M	
	rhcos-4.3.18-installer.ppc64le.iso	29-Apr-2020 15:46	97M	
	rhcos-4.3.18-metal.ppc64le.raw.gz	29-Apr-2020 15:45	740M	
	rhcos-4.3.18-openstack.ppc64le.qcow2.gz	20-Apr-2020 23:45	740M	
	rhcos-4.3.18-ostree.ppc64le.tar	29-Apr-2020 15:45	661M	
	rhcos-4.3.18-ppc64le-installer-initramfs.ppc64le.img	20-Apr-2020 23:45	68M	
	rhcos-4.3.18-ppc64le-installer-kernel-ppc64le	20-Apr-2020 23:45	23M	
	rhcos-4.3.18-ppc64le-installer.ppc64le.iso	20-Apr-2020 23:45	97M	
	rhcos-4.3.18-ppc64le-metal.ppc64le.raw.gz	20-Apr-2020 23:45	740M	
	rhcos-4.3.18-ppc64le-openstack.ppc64le.qcow2.gz	29-Apr-2020 15:46	740M	

<https://mirror.openshift.com/pub/openshift-v4/ppc64le/dependencies/rhcos/4.3/latest/>

Create the CoreOS image in PowerVC

- Download and extract the CoreOS image with suffix `-openstack.ppc64le.qcow2.gz`.
- Convert the CoreOS qcow2 image to raw image
 - `qemu-img convert -f qcow2 -O raw <image>.qcow2 <image>.raw`
- Attach a new volume with required size (e.g. 20G) to an existing PowerVC VM
- Copy the CoreOS raw image to this VM and dump the raw image to the newly added disk
 - `dd if=<image>.raw of=/dev/<device> bs=4M` where `<device>` is the newly added device
- Detach the newly added Volume, from the VM
- Go to PowerVC UI ->images and select create for creating a new image
- Specify image name and choose PowerVM for Hypervisor type, RHEL for Operating system and littleEndian for Endianness
- Select Add Volume (from the dd copy above) and set Boot set to yes.
- Create the image by clicking on create

Name	State	Operating System 	Description
 RHCOS 4.3.18	 Active	RHEL	Only for OpenShift 4.3

Installing OCP v4 - UPI on Power - Step 2

2

Follow the instructions to configure your environment and install your cluster

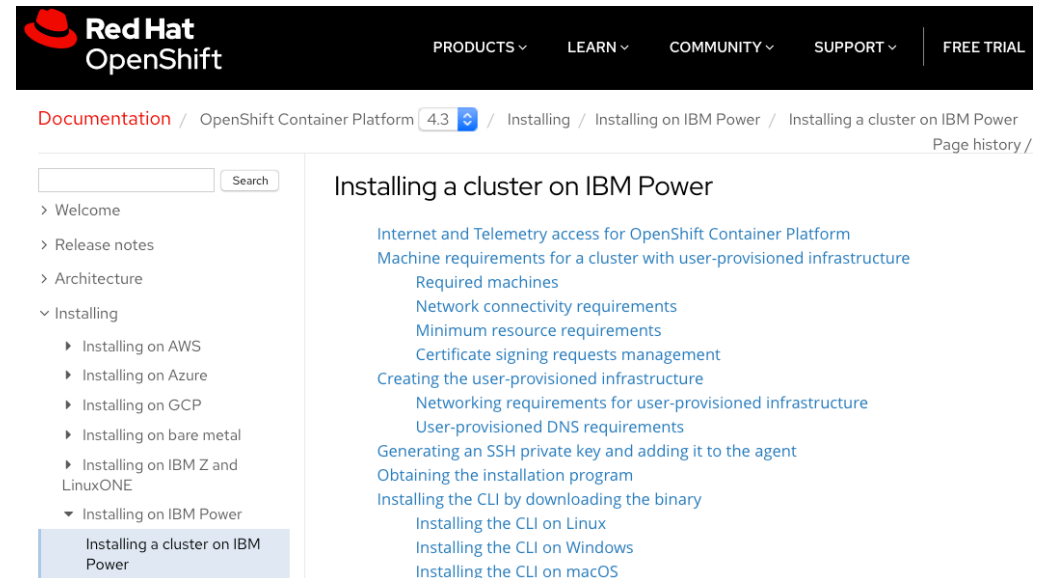
The installer will take about 45 minutes to run.

[Get started](#)

Red Hat collects a limited amount of telemetry data. By installing OpenShift Container Platform 4, you accept our data collection policy. [Learn more](#) about the data we collect.

The 'Get started' links to OpenShift docs:

https://docs.openshift.com/container-platform/4.3/installing/installing_ibm_power/installing-ibm-power.html



The screenshot shows the Red Hat OpenShift documentation website. The navigation bar includes the Red Hat logo, 'OpenShift', and links for 'PRODUCTS', 'LEARN', 'COMMUNITY', 'SUPPORT', and 'FREE TRIAL'. The breadcrumb trail reads: 'Documentation / OpenShift Container Platform 4.3 / Installing / Installing on IBM Power / Installing a cluster on IBM Power'. A search bar is present. The left sidebar lists a navigation menu with 'Installing a cluster on IBM Power' highlighted. The main content area is titled 'Installing a cluster on IBM Power' and lists several sub-topics: 'Internet and Telemetry access for OpenShift Container Platform', 'Machine requirements for a cluster with user-provisioned infrastructure' (including 'Required machines', 'Network connectivity requirements', 'Minimum resource requirements', and 'Certificate signing requests management'), 'Creating the user-provisioned infrastructure' (including 'Networking requirements for user-provisioned infrastructure' and 'User-provisioned DNS requirements'), 'Generating an SSH private key and adding it to the agent', 'Obtaining the installation program', 'Installing the CLI by downloading the binary' (including 'Installing the CLI on Linux', 'Installing the CLI on Windows', and 'Installing the CLI on macOS').

Requirements for UPI cluster

Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

Machine	Operating System	vCPU	Virtual RAM	Storage
Bootstrap	RHCOS	4	16 GB	120 GB
Control plane	RHCOS	2	16 GB	120 GB
Compute	RHCOS	2	8 GB	120 GB

The size and number of control plane (master nodes) will depend on the number of compute (worker) nodes

The size and number of compute (worker) nodes will depend on the number of pods expected to be running

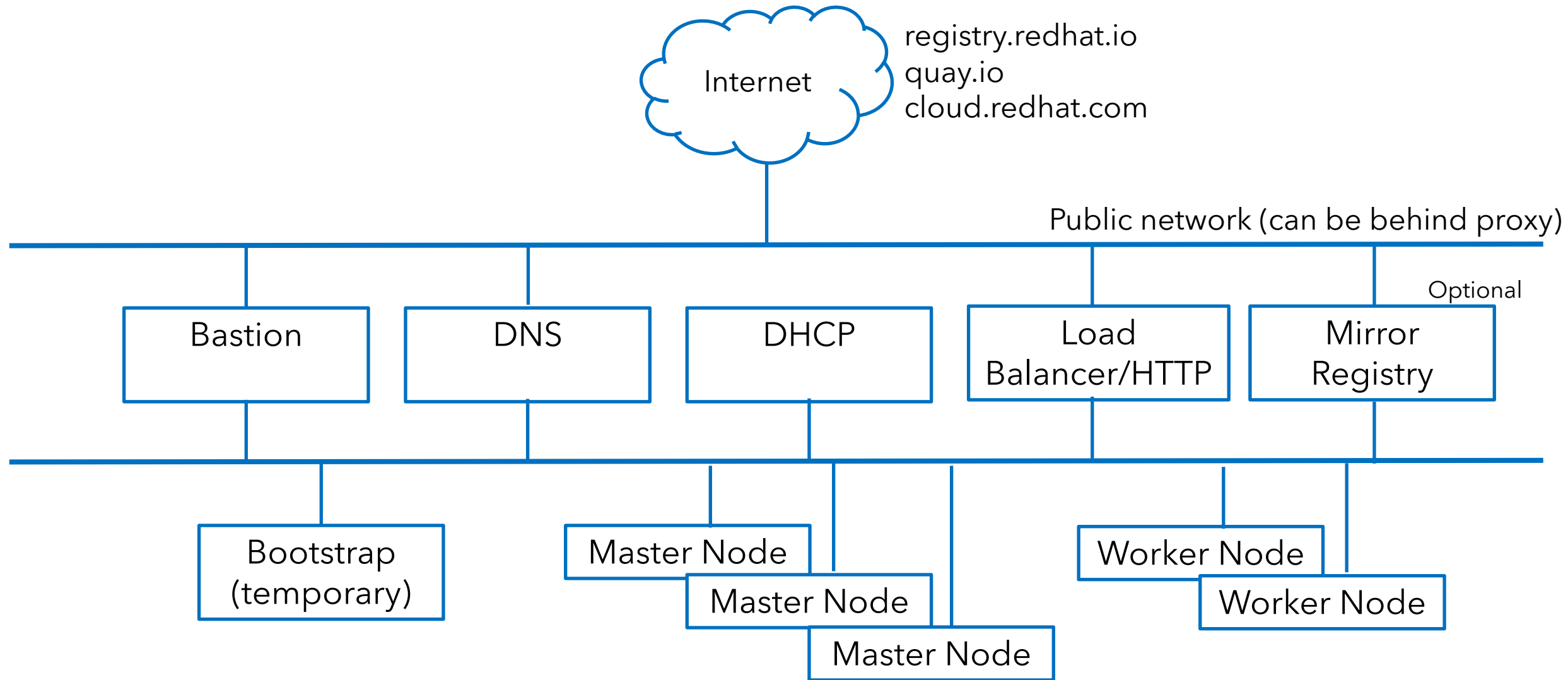
Minimum number of VMs/LPARs (all running RHCOS)

- One temporary bootstrap node
- Three control plane (master) nodes
- At least two compute (worker) nodes

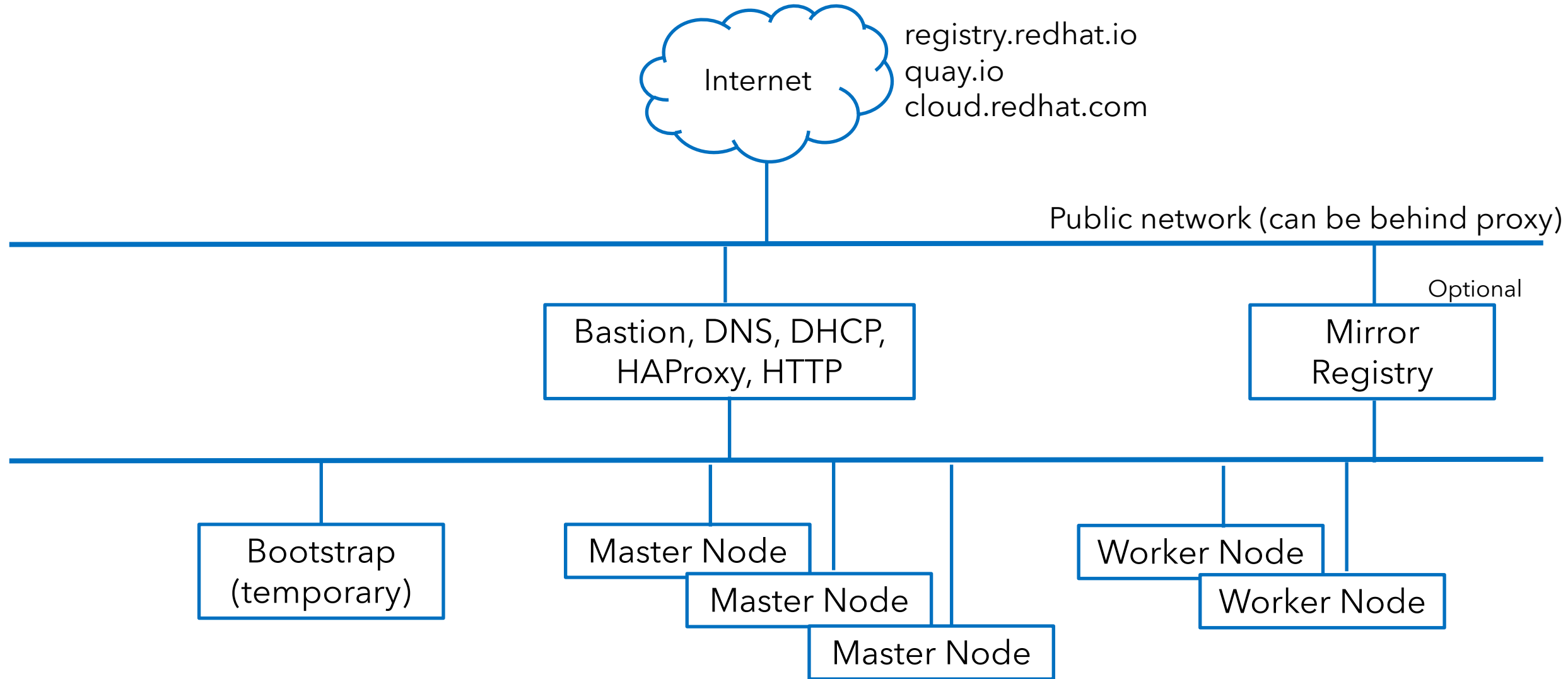
Requirements for UPI cluster

- Access to Red Hat OpenShift Cluster Manager
- Downloaded cluster install binaries, client binaries and pull secret
- Access to internet to obtain packages during install (or mirror repository)
- Red Hat CoreOS image for the VMs shown below
- VMs/LPARs (all running RHCOS)
 - One temporary bootstrap node
 - Three control plane (master) nodes
 - At least two compute (worker) nodes
- During initial boot all RHCOS nodes need a DHCP server or static IP set to establish connection to download their 'ignition' config files.
- Network access to the configuration/install server.
- DNS server updated with node IPs and cluster details.
- Load Balancers to provide controller and application ingress.
- SSH key to allow post deployment problem determination of the RHCOS nodes.

Installation process for UPI - Production



Installation process for UPI - Dev or PoC



Preparing the UPI Installation Process

From the configuration server setup the files you downloaded in Step 1

Untar the openshift-install file:

```
# openshift-install version
```

```
openshift-install 4.3.28
```

```
built from commit d6b6f1a91be310a647f9f91790020c5f11b2b2ff
```

```
release image quay.io/openshift-release-dev/ocp-
```

```
release@sha256:be44ba4532b9e1b695e5aeabbefe8855477d14498271c7637a5929c1035749c3
```

Untar the openshift-client file:

```
# oc version
```

```
Client Version: 4.3.28
```

Create a pull-secret.txt file from the pull secret:

```
# cat pull-secret.txt
```

```
{"auths":{"cloud.openshift.com":{"auth":"b3BlbnNoaWZ0LXJlbGVhc2UtZGV2K3N0dWN1bmxpZmZIMW81djFkY3Fscmd3andxbGR5aXU1aWVoenYxOIRUUIBVTE4wSVVKVklvNVVPUTIVWIBIQkg5M1dYMDhNQTgzMDVPSzFZMVZKUVA3....."}}
```


UPI Installation Process

The actual installation of OCP on User Provisioned Infrastructure, requires just 3 steps:

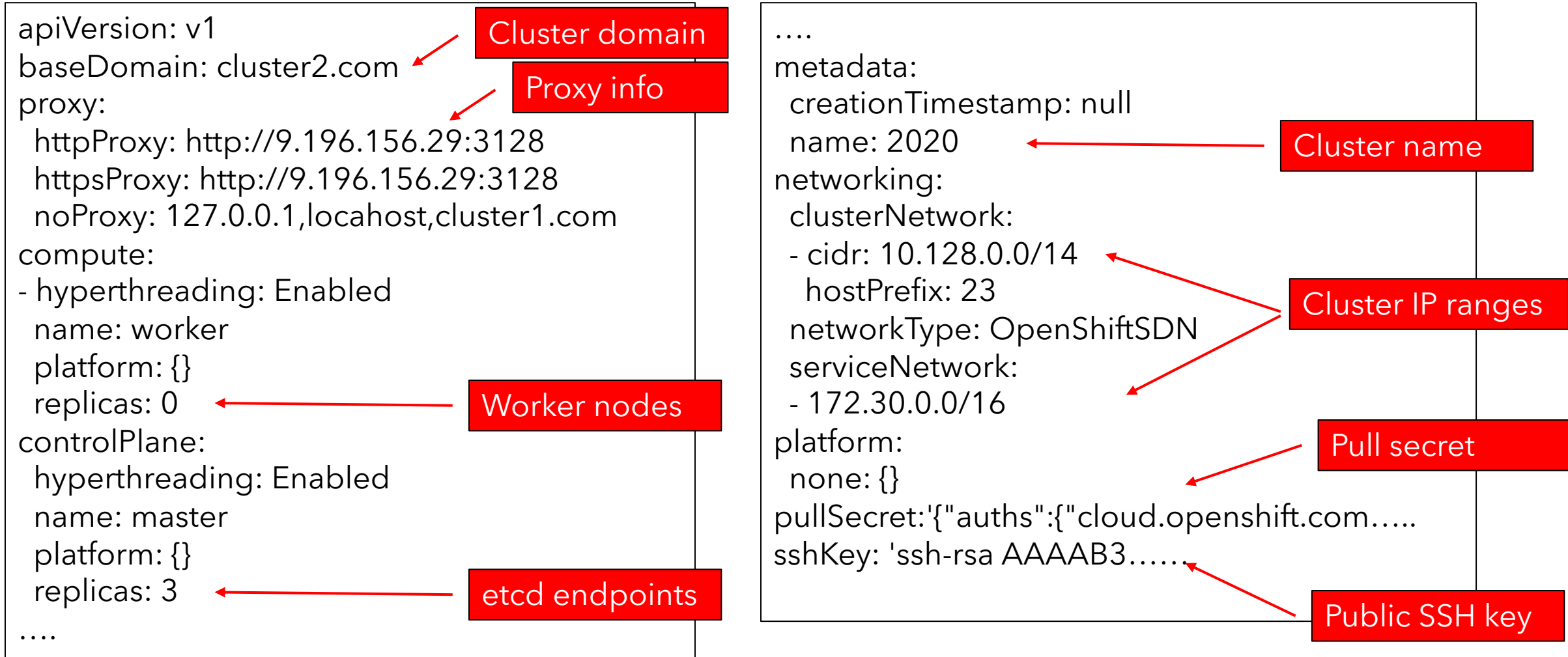
1. Prepare the 'install-config.yaml' file
2. Generated the manifests based on the install-config
3. Generate the ignition files based on the manifests

The install is performed automatically as the RHCOS nodes boot and obtain their ignition files.

NOTE: Ignition file contains installation certification which expires within 24 hours. You need to recreate ignition files after 24 hours. To recreate ignition files, you should generate them into empty directory.

Prepare install-config.yaml

During install the cluster configuration is read from install-config.yaml



Creating the Manifests and Ignition Files

Once you have your install-config.yaml file created you need to create the manifests:

```
# ./openshift-install create manifests --dir=<installation_directory>  
(where installation_directory is the location of your install-config file)
```

Once the manifests have been build you can create the ignition files. These ignition files are called by the new RHCOS VMs as they boot to configure themselves:

```
# ./openshift-install create ignition-configs --dir=<installation_directory>  
(where installation_directory is the location of your install-config file and manifests)
```

These ignition files can be modified should you require, and made available for the new VMs via the HTTP server configured.

Creating your RHCOS VMs

PowerVC/OpenStack

If you are using PowerVC you can build the VMs simply using the image you created previously.

Using ISO images

Download the ISO file and the RAW disk file (shown previously), e.g.

- ISO: rhcos-<version>-installer.<architecture>.iso
- Compressed metal RAW: rhcos-<version>-metal.<architecture>.raw.gz

Upload the RAW RHCOS image file to the HTTP server (same location as Ignition files)

Use the ISO to start the RHCOS installation

Edit the kernel during boot to point at the install file, ignition file and DHCP server

Using PXE boot

Download the RHCOS ISO image, compressed metal RAW image, kernel and initramfs files

- ISO: rhcos-<version>-installer.<architecture>.iso
- Compressed metal RAW image: rhcos-<version>metal.<architecture>.raw.gz
- kernel: rhcos-<version>-installer-kernel-<architecture>
- initramfs: rhcos-<version>-installer-initramfs.<architecture>.img

Upload the compressed metal RAW image and the kernel and initramfs files to your HTTP server.

Configure the PXE boot to new file location, DHCP server etc.

UPI Installation Process

Bootstrapping a cluster involves the following steps:

- The bootstrap machine boots and starts hosting the remote resources required for the master machines to boot. The master machines fetch the remote resources from the bootstrap machine and finish booting.
- The master machines use the bootstrap machine to form an etcd cluster.
- The bootstrap machine starts a temporary Kubernetes control plane using the new etcd cluster.
- The temporary control plane schedules the production control plane to the master machines.
- The temporary control plane shuts down and passes control to the production control plane.
- The bootstrap machine injects OpenShift Container Platform components into the production control plane.
- The installation program shuts down the bootstrap machine. (Requires manual intervention in UPI)
- The control plane sets up the worker nodes.
- The control plane installs additional services in the form of a set of Operators.

The result of this bootstrapping process is a fully running OpenShift Container Platform cluster.

The cluster then downloads and configures remaining components needed for the day-to-day operation, including the creation of worker machines in supported environments.

Installing via Terraform

<https://medium.com/@yussufshaikh/install-openshift-4-3-upi-on-powervm-using-powervc-a1e5584f5d13>

Install OpenShift 4.3 (UPI) on PowerVM using PowerVC



Yussuf Shaikh [Follow](#)
Apr 20 · 17 min read



In this post, I will explain the detailed steps required for installing OpenShift 4.3 on PowerVM using PowerVC. Also, I will share the Terraform module created using the UPI mode of installation on PowerVC.

OpenShift 4.3 is now GA to install from <https://cloud.redhat.com/openshift/install/power/user-provisioned>. Please refer to [OpenShift on IBM Power](#) blog for more information.

Table of Contents

- [Introduction](#)
- [Collecting Information](#)
 - [Things to keep handy](#)
- [Install Configuration](#)
 - [Bastion node](#)
 - [Installer Binary](#)
 - [Install Config](#)
 - [Manifests](#)
 - [Ignition Config](#)

Introduction

To install OCP 4.3 on PowerVM we refer to the UPI documentation from OpenShift. Here the bastion node and cluster nodes are created by the user on which OpenShift will be installed. We also need to set up a DHCP, DNS and HAPROXY service for the working of the cluster. The bastion node will be used for this purpose. Also, note that we will be using the HTTP server on the bastion node to host the bootstrap ignition file required for booting up the bootstrap node.

Also to create the resources on PowerVC we will make use of OpenStack client which is already installed and configured. It is better to use the client as we will be using CLI throughout. You could use alternative ways as well.

Installing OCP v4.3 Cluster via Terraform

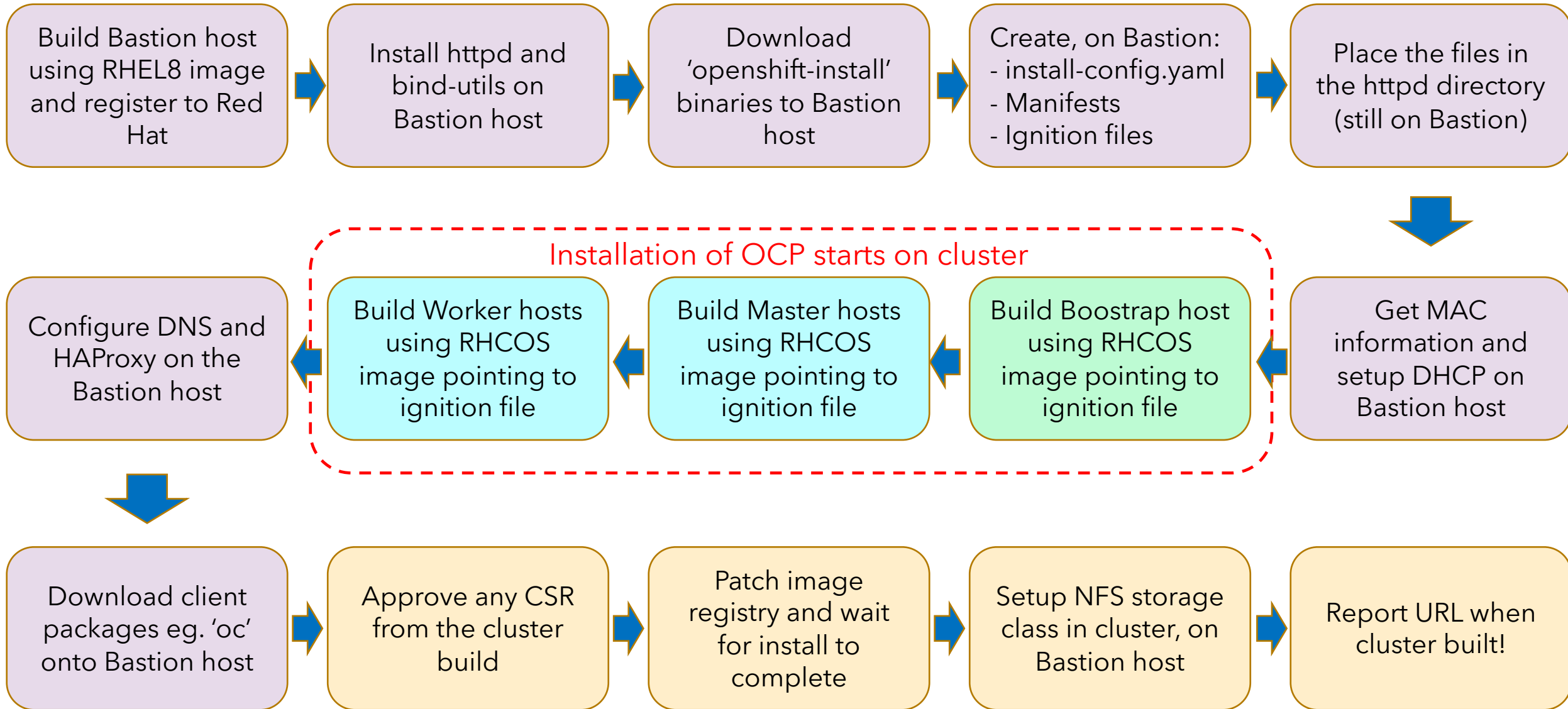
<https://github.com/ocp-power-automation/ocp4-upi-powervm>

```
[Stus-MBP-2:ocp4_upi_powervm stucunliffe$ ls -l *.tf
-rw-r--r--  1 stucunliffe  admin  6406 12 Jun 16:52 ocp.tf
-rw-r--r--  1 stucunliffe  admin  2184  1 May 13:37 outputs.tf
-rw-r--r--  1 stucunliffe  admin  6451  1 May 13:37 variables.tf
[Stus-MBP-2:ocp4_upi_powervm stucunliffe$
[Stus-MBP-2:ocp4_upi_powervm stucunliffe$ ls -l var.tfvars
-rw-r--r--  1 stucunliffe  admin  1841  9 Jul 18:30 var.tfvars
[Stus-MBP-2:ocp4_upi_powervm stucunliffe$
[Stus-MBP-2:ocp4_upi_powervm stucunliffe$ ls modules/
1_bastion      2_preinstall  3_network_    4_nodes      5_dns_haproxy  6_install    7_storage
```

Terraform files

Terraform modules to perform step by step installation

Installing OCP v4.3 Cluster via Terraform

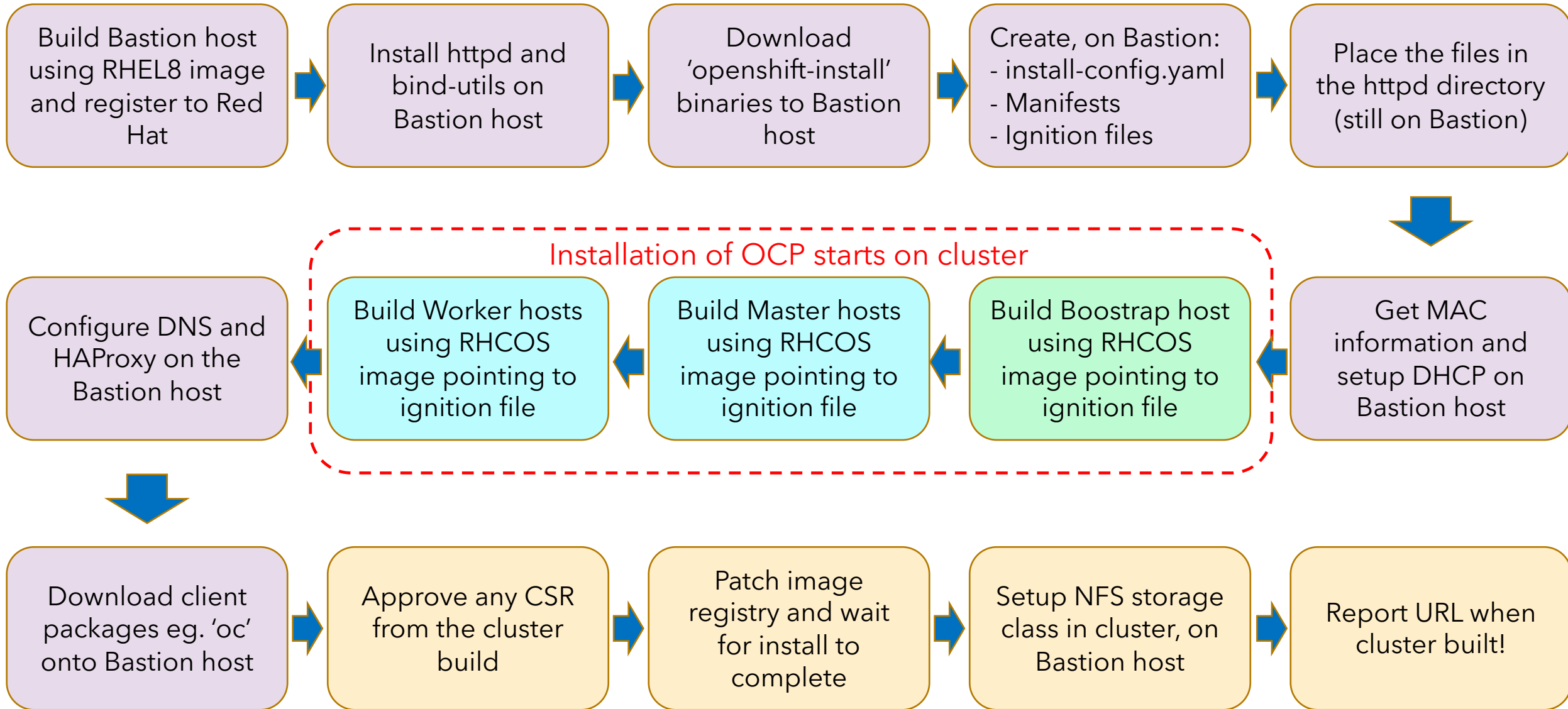


Installing OCP v4.3 Cluster via Terraform



Demo

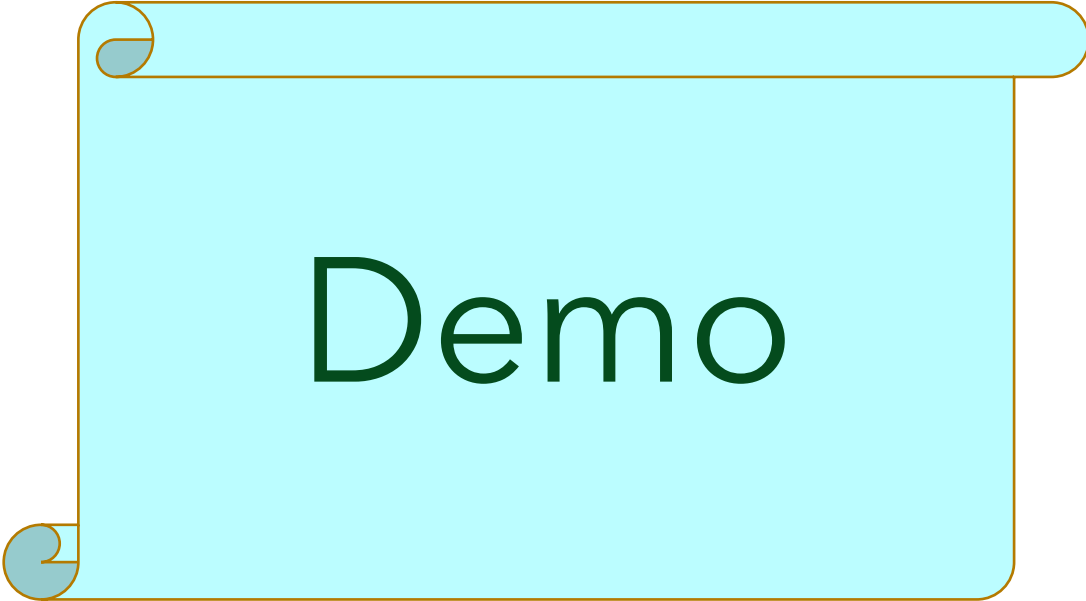
Installing OCP v4.3 Cluster via Terraform



Upgrading OCP v4.3 including RHCOS



Demo



Summary

- OCP version 4.3+ is now supported on IBM Power
- It is supported on Power9 but runs on my Power8 servers in my lab
- If you have ~6 free cores you can setup a PoC in your IBM Power environment

Overview

- <https://www.ibm.com/ibm/clientcenter/montpellier/index.shtml>
- <http://www.redbooks.ibm.com/abstracts/redp5599.html?Open>
- <https://www.openshift.com/try>

Installation

- <https://medium.com/@yussufshaikh/install-openshift-4-3-upi-on-powervm-using-powervc-a1e5584f5d13>
- <https://cloud.redhat.com/openshift/install/power/user-provisioned>
- https://docs.openshift.com/container-platform/4.3/installing/installing_ibm_power/installing-restricted-networks-ibm-power.html
- https://docs.openshift.com/container-platform/4.3/installing/installing_ibm_power/installing-ibm-power.html

Thank you

email: s_cunliffe@uk.ibm.com

Twitter: [@StuCunliffe](https://twitter.com/StuCunliffe)

Notices and disclaimers

- © 2019 International Business Machines Corporation. No part of this document may be reproduced or transmitted in any form without written permission from IBM.
- **U.S. Government Users Restricted Rights – use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.**
- Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed “as is” without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.
- IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.”
- **Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.**
- Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those
- customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.
- References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.
- Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.
- It is the customer’s responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer’s business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimers continued

- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**
- The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.
- IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml