

Install IBM Cloud Private v2.1 on Red Hat Enterprise Linux v7

Introduction

This is a guide to creating an IBM Cloud Private (ICP) cluster on RHEL v7.x. It is intended to enhance the installation instructions available in the [ICP Knowledge Center \(KC\)](#).

The phrase "ICP instance" is used in this document to refer to an installation of ICP.

The term "cluster" is used to mean the VMs that are under the control of a given ICP instance. At times in this guide the word "cloud" is used to mean the same thing as cluster.

This instruction guide assumes the VMs to be used for the ICP instance are provided to you by some providing entity.

Root access, either directly or through sudo, must be available to the person doing the ICP installation for all of the ICP machines (VMs) that will be members of the ICP cluster. If sudo is used, it is recommended that passwordless sudo is configured. Otherwise, the cluster configuration file (config.yml) will need the sudoer's password specified. An ICP installation as non-root is beyond the scope of this document.

IBM Cloud Private is built on Docker and Kubernetes.

IBM Cloud Private Cloud Foundry (ICP-CF) is packaged as part of the IBM Cloud Private product. However, ICP-CF has a completely separate installation process. In addition ICP-CF is a completely separate run-time environment that has no direct integration points or any dependencies on ICP itself. The installation and deployment of IBM Cloud Private Cloud Foundry is outside the scope of this guide.

IBM Cloud Private has certain components (the "master" and "proxy" nodes, for example) that may be deployed as singletons in a simple, "sandbox" installation. A production installation should include three (3) or five (5) master nodes and (3) proxy nodes. A simple installation of ICP will combine the boot, master and "management" components into a single node. A production installation will separate the management nodes from the master nodes to provide sufficient resources to each and avoid resource contention. A simple ICP installation may use a singleton NFS server to provide a shared, persistent storage service to all of the worker nodes. A production ICP installation will use something more sophisticated such as GlusterFS or IBM Spectrum Storage for persistent storage. (This guide provides a reference to a

GlusterFS install guide.)

Deployment	Sandbox Instance	Production Instance
Boot	Combined w/ Master	Possibly separate, but usually combined with Master01
Master	1	3 or 5 (odd number required to support quorum voting)
Proxy	1	3
Worker	1 to 3	Typically, 5 or more
Management	Combined w/Master	2 or 3
File Store	Singleton NFS server	GlusterFS (independent server cluster of 3 nodes)

The installation described in this document uses Docker CE. Installing Docker EE is outside the scope of this document.

NOTE: As of March 2017, Docker broke out two builds, one referred to as Community Edition (CE) and the other referred to as Enterprise Edition (EE). Docker EE is intended to be used on RHEL. Docker EE requires a subscription and payment for a license. Docker CE and Docker EE are intended to be functionally identical. Docker EE comes with full support as opposed to Docker CE, which only has "community" support, i.e., if you run into a problem you may not be able to get support in a timely manner.

NOTE: If you want to use Docker CE on RHEL, get the [Docker CE packages built for Centos](#). Packages built for Centos have binary compatibility with RHEL.

For Docker on RHEL information see: "[Get Docker EE for Red Hat Enterprise Linux](#)".

NOTE: In this document the standard root prompt `#` for sample shell commands is replaced with `>` to avoid having random sample commands show up in the Atom editor's markdown section navigation pane. There are some file snippets where comment lines start with `#` that do continue to show up in the section navigation pane.

Getting started overview

This section provides a big picture summary of the installation process.

- An IBM Cloud Private (ICP) installation is usually made up of multiple virtual machines that are collectively referred to as a cluster. (This document may also refer to the collection of machines as a cloud or an ICP instance.) Individual machines are referred to as nodes. An ICP installation includes machines with a special role as itemized here:
 - Boot-master - The machine where the ICP control is installed. (It is typical to combine the "boot" and "master" into one machine.)
 - Management - The simple installation used to create this document combines the management role with the boot-master node.
 - Proxy - Machine used for request routing
 - Multiple worker nodes
- The ICP v2.1 Knowledge Center section [Supported operating systems and platforms](#), has a list of supported operating systems. ICP is supported on RHEL 7. ICP is not supported on RHEL 6.
- System requirements: The ICP v2.1 Knowledge Center section [Hardware Requirements and Recommendations](#), has a table describing the system requirements for each of the types of nodes in an ICP instance. The installation described in this document is a "multi-node cluster".
- It is a really good idea that all machines used for ICP have access to the RHEL yum repositories (os, optional and extras) in order to install various RHEL packages that are pre-requisites for ICP. Many data centers have a Red Hat Satellite server available. (A public Centos yum repository is available at <http://mirror.centos.org/>.)
- It is a really good idea for all the machines in the ICP cluster to have access to the public Internet. The public Docker yum repository makes it convenient to install the latest version of Docker CE. Docker Hub is convenient for access to commonly available Docker images. The latest Gluster images are also easily retrieved from public sources.
- The details of an "air-gap" install are not covered completely in this document. Needless to say, doing an air-gap installation is more challenging due to the inconvenience of pulling together all of the RPMs and other artifacts needed to do an installation.

The simple, "sandbox" ICP installation in a nutshell:

See the ICP Knowledge Center section, [Installing an IBM Cloud Private Enterprise environment](#)

1. Customize RHEL for Docker and ICP.
2. Install Docker on the boot-master.
3. Set up RSA based ssh login from the Boot-Master to all nodes in the cluster.
4. Run the ICP inception installer on the boot-master.

Suggested ICP "sandbox" deployment resource allocations are described in the table below.

Machine Role	Number	vCPU/Core	Memory (GB)	Storage Disks x Size (GB)
Boot/Master/Mgmt	1	2	8	1 x 250
Proxy	1	2	4	1 x 100
Worker	2 or 3	2	8	1 x 100
NFS Server	1			1 x 100

A production ICP installation in a nutshell:

See the ICP Knowledge Center section, [Installing an IBM Cloud Private Enterprise HA environment](#)

1. Customize RHEL for Docker and ICP.
2. Install Docker on the boot-master.
3. Set up RSA based ssh login from the Boot-Master to all nodes in the cluster.
4. Run the ICP inception installer on the boot-master.
5. Install GlusterFS server on the GlusterFS nodes.
6. Install GlusterFS client on all ICP cluster members.
7. Install Heketi for storage administration.
8. Configure the shared Docker repository and audit log for the master nodes.
9. Configure LDAP registry from authentication and role based access control.

Suggested ICP production deployment resource allocations are described in the table below.

Machine Role	Number	vCPU/Core	Memory (GB)	Storage Disks x Size (GB)

Master	3 or 5	8	32	1 x 250
Proxy	3	2	4	1 x 220
Management	2 or 3	8	32	1 x 250
Worker	5+	4	32	1 x 220
GlusterFS	3+	4	16	1 x 40 (/dev/sda) 1 x 128 (/dev/sdb) 1 x 128 (/dev/sdc)

ICP Master and **ICP Management** nodes suggested disk partitioning (250 GB disk).

File System Name	Mount Point	Size (GB)
system (aka root)	/	40
boot	/boot	256 MB
swap		8
var	/var	50
tmp	/tmp	20
home	/home	10
opt	/opt	120

ICP Proxy node suggested disk partitioning (220 GB disk).

File System Name	Mount Point	Size (GB)
system (aka root)	/	20
boot	/boot	256 MB
swap		4
var	/var	50
tmp	/tmp	10
home	/home	10

opt	/opt	120
-----	------	-----

ICP Worker node suggested disk partitioning (220 GB disk)

File System Name	Mount Point	Size (GB)
system (aka root)	/	20
boot	/boot	256 MB
swap		4
var	/var	60
tmp	/tmp	10
home	/home	10
opt	/opt	130

Additional steps depending on specific circumstances and requirements:

- Configure access to RHEL yum repositories.
- Configure `/etc/hosts` files on all cluster members if DNS is not available to resolve host names and IP addresses.
- Update RHEL to the latest patch level.
- Install NTP.
- Install Python Docker modules to support the convenient use of Docker APIs in Python scripts.
- Install Docker on each cluster member VM in addition to the boot-master VM. (This gives you full control over what version of Docker is installed, but more importantly, Docker on each VM is needed for the next step.)
- Use Docker on each cluster member VM to pre-load the ICP Docker images rather than let the inception installation load the ICP Docker images. (It turns out to be expedient to copy the ICP image tar-ball to each cluster member VM and then load the local Docker registry from that tar-ball rather than waiting for the inception installer to do that part of the installation.)
- Install `kubectl` on the boot-master node. `kubectl` is useful for interacting with the ICP cluster.
- Install `helm` on the boot or boot-master node. Helm is useful for installing additional

software on the cluster.

- Install Ansible on the boot or boot-master node or, even better, on the administrator's desktop/laptop. Ansible is very useful for administration when dealing with multiple machines. (The instructions in the [Install and configure the GlusterFS server cluster] (#Install and configure the GlusterFS server cluster) section assumes the availability of Ansible.) See the [Installing Ansible](#Installing Ansible) section for guidance on installing and configuring Ansible.

Basic RHEL configuration

The sub-sections in this section describe some basic RHEL configuration required to support ICP cluster members. We are calling this "basic" configuration because it is typical that a RHEL VM will be configured as described in these sub-sections.

If you are creating your own virtual machine, you can do all these steps on your initial VM before cloning it.

If you are using VMs that were deployed for you, then it is likely the VMs are already configured as described in these sub-sections. You should confirm with your VM provider or by doing the basic checks described below, that all the VMs that are going to be part of the ICP cluster have been configured as described in these sub-sections.

Configure network interface to start on boot

NOTE: This section describes steps you won't need to do for a virtual machine deployed for you in a typical virtualization platform. If you are building your own VM, you will likely need to complete these steps.

By default, RHEL 7 network interfaces are not started at boot time. This tends to be inconvenient. This section describes the steps to configure a network interface to start at boot time.

To find out the names of the network interfaces on a machine you can use the `ifconfig` command. The name of each network interface is in the first column.

The network interface configuration files are in `/etc/sysconfig/network-scripts/`. The file names include the name of the network interface as seen in the `ifconfig` command output.

1. Edit the file in `/etc/sysconfig/network-scripts` associated with the network interface to be started at boot time, e.g., `/etc/sysconfig/network-scripts/ifcfg-ens33`.

2. Change the value of `ONBOOT` from no to yes: `ONBOOT=yes`
3. Save the changes.
4. If desired, test by rebooting:

```
shutdown -r now
```

When the VM comes back up the network interface should be UP/RUNNING. The `ifconfig` command includes the status of each interface.

Configure the host name

This section describes the configuration of the host name for RHEL 7 VMs.

If you are cloning VMs, then you will need to change the host name on the cloned VM.

NOTE: This step is unnecessary and is **not advisable** in scenarios where the VMs were provided to you. Check with your VM provider, system administrator and/or network administrator before changing a host name.

Host name considerations usually involve using naming conventions provided by the network and system administrators.

Be sure to use a fully qualified domain name (FQDN) for the value of the host name.

Some sample "toy" host names: `bootmaster` , `proxy` , `worker01` , etc.

There are a couple of different places where the host name is specified.

For RHEL 7 the host name can be set using the `hostnamectl` command. For example:

```
hostnamectl set-hostname bootmaster01.site.org.com
```

See the man pages for `hostnamectl` for more information.

Additional notes on host name:

1. The host name is stored in the `/etc/hostname` file. (NOTE: This file should only have the

text of the host name on a single line with no newline character.)

2. There may be a HOSTNAME directive in the `/etc/sysconfig/network` file.
3. A reboot is needed to have the host name change take effect: `shutdown -r now`

Enable remote login for root via ssh

NOTE: For RHEL v7.x, by default root can login via ssh using a password.

The configuration for ssh is in `/etc/ssh/sshd_config`. You will notice that `PermitRootLogin` is commented out. However, `UsePAM` is set to yes. A PAM configuration file for `sshd` is in `/etc/pam.d`. Configuration of the PAM plugin for `sshd` is beyond the scope of this document.

Yum repository configuration

NOTE: A RHEL virtual machine provided for you will very likely already have a yum repository configured, most likely using a Red Hat Satellite (RHS) server. This section is intended to provide guidance for those who need to configure a yum repository.

NOTE: You may need to provide a userid and password as part of the yum repository URL. If you are using an ID with an @ character in it, e.g., an Internet email address, use %40 in place of the @ character. Otherwise the repository URL is misinterpreted because an @ character marks the beginning of the host name in the URL.

You need to set up the repos for yum to be able to install additional packages and get OS updates.

The yum repo definition files are in `/etc/yum.repos.d/`. Any file in that directory with a `.repo` extension will be treated as a repo definition.

Each file may have multiple repositories defined.

Each repo definition typically has at least 5 attributes:

```
[rhel-os]
name=Red Hat Enterprise Linux at my site
enabled=1
gpgcheck=0
baseurl=<protocol>://<userid>:<password>@<repo_host>/<repo_path>
```

The `<protocol>` could be ftp or http(s) for remote repos, or file for a local repo.

If the repository requires a `<userid>` and `<password>`, that is included in the URL as shown.

See the Red Hat documentation [Configuring Yum and Yum Repositories](#) for more details.

The `<repo_path>` needs to point to a directory with a `repodata` sub-directory where a file named `repomd.xml` is found. If you need to explore a repository to determine the , something like Filezilla is very handy to use for exploration. You can also confirm your user ID and password in getting to the repo. (See figure below.)



In the figure above the `<repo_path>` text would be:

`/redhat/rhs6/server/7/7Server/x86_64/extras/os/`

You may need to explore a given collection of yum repositories in order to figure out where various packages are located. Packages are typically spread across multiple directories such as "os", "optional" and "extras".

Depending on how DNS is configured, you may need to add an entry in the `/etc/hosts` file for the `<repo_host>`.

Once you have configured the yum repository you can do a quick test to confirm the configuration file is correct:

```
yum repolist
```

Yum caches its repository information as a performance enhancement. If you want to "clean" the cache to make sure you are not using stale information about yum repositories, use:

```
yum clean all
```

Once you have the desired yum repositories configured, you can proceed with any RHEL configuration that requires additional packages (rpms) to be installed.

At some point while using yum, you may see "Given file does not exist" errors such as:

```
rhel-optional/updateinfo          FAILED
ftp://<userid>:<password>@<repo_host>:/redhat/rhs6/server/7/7Server/x86_64/optional
Trying other mirror.
rhel-optional/primary             FAILED
ftp://<userid>:<password>@<repo_host>:/redhat/rhs6/server/7/7Server/x86_64/optional
Trying other mirror.
ftp://<userid>:<password>@<repo_host>:/redhat/rhs6/server/7/7Server/x86_64/optional
Trying other mirror.
ftp://<userid>:<password>@<repo_host>:/redhat/rhs6/server/7/7Server/x86_64/optional
```

If you inspect the yum repository, you will notice that the `<uuid1>-updateinfo.xml.gz` and `<uuid2>-primary.xml.gz` files have a new UUID.

A `yum clean all` will likely clean up such errors.

Update RHEL

This section describes the steps to update RHEL.

NOTE: This is an optional step and may not be necessary depending on the virtual machine that has been provided to you. You may also not be permitted to do a full update, as that may move your RHEL to a version that is not yet supported by your operations team. For example, if the VM is running RHEL 7.3 a full update will result in RHEL 7.4.

If you are building a VM from scratch, then you should do a full RHEL update to pick up updates since the release of the RHEL image you are using for the initial install.

It is assumed you have configured your VM with yum repositories.

1. Do a yum update (`yum -y update`) - This gets all the latest patches for everything already installed.
2. Then reboot. (`shutdown -r now`). You need to reboot to get to the latest kernel.

Install NTP

Some mechanism is needed on each VM to keep the time synchronized with the rest of the world. All of the VMs in the ICP cluster will need to share a common notion of time, and the usual approach to keeping time is to use NTP.

NOTE: If you are using virtual machines provided to you, it is very likely NTP is already installed and enabled for startup at machine boot.

NOTE: If you are using virtual machines provided to you, and NTP is not installed and in use, the VMs may be using some other time provider, such as the hypervisor. Check with your provider to determine if it is necessary to install NTP.

- Install NTP

```
yum -y install ntp
```

- Check the NTP configuration in `/etc/ntp.conf`

See the Red Hat documentation, [Configure NTP](#), for more detailed guidance.

NOTE: The default `/etc/ntp.conf` content is likely sufficient. You may want to add one or more local time providers to the list of servers provided by Red Hat:

```
server 0.rhel.pool.ntp.org iburst
server 1.rhel.pool.ntp.org iburst
server 2.rhel.pool.ntp.org iburst
server 3.rhel.pool.ntp.org iburst
```

- Enable the NTP daemon. (This will make sure ntpd starts up when the machine is booted.)

NOTE: The name of the service is **ntpd**, not ntp. (Go figure.)

```
systemctl enable ntpd
```

- Start the NTP service.

```
systemctl start ntpd
```

- Check the NTP service status.

```
systemctl status ntpd
```

- If you need to stop the NTP service:

```
systemctl stop ntpd
```

- To get a list of peer servers in use:

```
ntpq -p
```

NOTE: Leave NTPD started and enabled (so that it starts at boot time).

Configure /etc/hosts

This section describes the steps to add entries to the `/etc/hosts` file of each cluster member for all the hosts in the cluster.

NOTE: This section can be skipped if DNS is being used for host name resolution. You can


use `nslookup` on the IP addresses of a sampling of cluster members to determine if there are DNS entries for the cluster member VMs. If not, then you need to configure `/etc/hosts`.

NOTE: The "minimal" RHEL install does not include `bind-utils`, which is the RPM that contains the `nslookup` command. If you are using a minimal RHEL image, then you need to install `bind-utils` if you want to use `nslookup`. (`yum -y install bind-utils`)

- For each VM in the cluster, edit `/etc/hosts` and add an entry for each VM in the cluster.

In some circumstances you can edit `/etc/hosts` once on the boot-master and then `scp` the hosts file to the other members of the cluster. This expedient is only feasible if the all VMs in the cluster are freshly deployed VMs and they all have the same content in `/etc/hosts` when they are initially deployed, e.g., the default content.

Check the file system sizing

The file system size minimum requirements are described in the ICP v2.1 Knowledge Center section,  (See Table 3). The "important" note following the table suggests mounting `/var/lib/etcd`, `/var/lib/registry` and `/opt/ibm/cfc` on separate paths associated with larger disks.

NOTE: Do not use `/opt/ibm/cfc` as a file system mount point. The ICP uninstall has a step that deletes the `/opt/ibm/cfc` directory. If that directory is a mount point for a file system, the delete operation will fail. Use either `/opt` or `/opt/ibm` as the mount point for the file system.

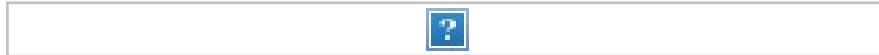
TDB: What mount point should be used for `/var/lib/etcd`, `/var/lib/registry` and the other ICP related content in `/var/lib`? (The safe thing to do is mount the file system at `/var` or `/var/lib`.)

The file system requirements as described in the ICP v2.1 KC documentation are reproduced in the following table:

Location	Minimum Disk Space
<code>/var/lib/docker</code>	>50 GB (This is more than the KC calls for.)
<code>/var/lib/etcd</code>	>1 GB
<code>/var/lib/registry</code>	Large enough to host the Docker images you expect to load into the local registry. Docker images for ICP itself are on the order of 8 GB.

<code>/opt/ibm/cfc</code>	>100 GB
<code>/var/lib/kubelet</code>	>10 GB w/o Vulnerability Advisor
<code>/var/lib/kubelet</code>	>100 GB w/ Vulnerability Advisor

NOTE: See the section in the ICP v2.1 Knowledge Center,



NOTE: If you are creating your own RHEL VMs, there is a step at the point where the install is being set up where you can customize the disk partitions rather than take the automatic defaults. It is recommended that you choose to customize the disk partitions to ensure the VM meets the ICP file system sizing requirements. The RHEL 7 documentation for installation that describes the file system configuration is in the section [Installation Destination](#). Scroll down to the "Manual Partitioning" section to get to the details that describe the manual partitioning steps. The section below, [Installing RHEL 7 - a sample](#Installing RHEL 7 - a sample) provides a step-by-step example of installing RHEL 7, including creating custom disk partitions.

The RHEL 7 documentation for  provides detailed information about the mount command.

Special RHEL configuration

The sub-sections in this section describe RHEL configuration that is not typical. Even on a VM provided to you, you will very likely need to take the steps described in these sub-sections in order to prepare the machine to be a member of an ICP cluster.

vm.max_map_count

The `map_max_count` determines the maximum number of memory map areas a process can have. Docker requires that the `max_map_count` be substantially greater than the default (65530). This section describes the steps to setting the `vm.max_map_count` to an appropriate number.

NOTE: Do not confuse the `sysctl` command with the `systemctl` command. The command for getting and setting system (kernel) parameters is `sysctl` not `systemctl`.

To make an immediate change to the `vm.max_map_count` system parameter:

```
sysctl -w vm.max_map_count=262144
```

NOTE: The above command writes the value of `max_map_count` to the file `/proc/sys/vm/max_map_count`. However, the `vm.max_map_count` will revert to its default value on reboot.

NOTE: Consult your system administrator to determine the preferred place to persist the value of `vm.max_map_count`.

In order to have the `vm.max_map_count` carry over through a reboot, edit the `/etc/sysctl.conf` file and add a line to define the value:

```
vm.max_map_count=262144
```

or

```
echo "vm.max_map_count=262144" >> /etc/sysctl.conf
```

If you want to see the value of a system control variable just use: `sysctl <name>`, e.g.,

```
sysctl vm.max_map_count
```

Delete the `/var/lib/mysql` directory

If the `/var/lib/mysql` directory exists on any of the ICP cluster VMs, the install will fail on that VM. At least on some RHEL 7 installations, it seems mysql gets a directory in `/var/lib` even if it is not used for anything.

- On all VMs in the cluster/cloud, make sure there is no `mysql` directory in `/var/lib`, i.e.,

```
rmdir /var/lib/mysql
```

Installing Docker CE using the public

Docker yum repository

You have two options for installing Docker:

1. Install Docker only on the boot-master machine and let Docker be installed on all of the other cluster members as part of the ICP installation.
2. Install Docker on the boot-master and all of the cluster members. This install guide uses this option because it tends to be faster to install Docker on each machine. Once Docker is installed on each machine, the ICP Docker images can be loaded into the local Docker registry on each machine, which is another trick to speed up the ICP installation.

This section describes the steps to install Docker Community Edition on RHEL v7 using the public Docker yum repository. It is assumed the VM(s) where Docker is to be installed have access to the Internet and can get to **download.docker.com** and you can use the yum repository defined there.

Docker Community Edition is identical in behavior to Docker Enterprise Edition. IBM Cloud Private is supported on Docker Community Edition, or older versions of Docker before the differentiation occurred between Docker editions. The installation of Docker Enterprise Edition is beyond the scope of this document.

A typical RHEL yum repository may have the docker RPMs, but they may not be the current version. It is recommended that the docker version available at **download.docker.com** be used.

NOTE: This installation scenario assumes Internet connectivity and access to **docker.com**.

The Docker documentation, [Get Docker CE for CentOS](#) has detailed information on [Docker Community Edition installation](#). The instructions here are derived from the Docker documentation.

NOTE: The yum-config-manager utility is part of the yum-utils RPM. If your machine doesn't have yum-config-manager then you need to install yum-utils (`yum -y install yum-utils`). (A RHEL "minimal install" does not include yum-utils. Yum-utils has some Python related pre-reqs that also get installed.)

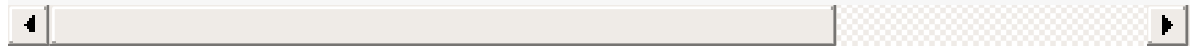
- Set up the public docker yum repository

```
yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-c
```



You should see output similar to this:

```
Loaded plugins: langpacks, product-id
adding repo from: http://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to /etc/y
repo saved to /etc/yum.repos.d/docker-ce.repo
```



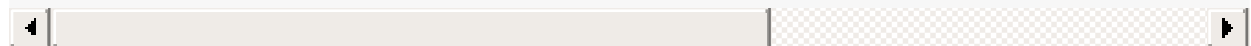
- Install Docker CE.

```
yum -y install docker-ce
```

NOTE: The docker-ce install has a pre-req of container-selinux. If there is no container-selinux in your yum repositories the above command will fail. (The container-selinux package is in the RHEL "extras" repo.)

A work-around to the above install is described here: <https://github.com/docker/for-linux/issues/201> The following command was taken from the work-around:

```
yum -y install --setopt=obsoletes=0 docker-ce-17.03.2.ce-1.el7.centos.x86_64 docker
```



- Start and enable the docker daemon.

```
systemctl start docker
systemctl enable docker
```

- Run the usual `docker run hello-world` to confirm the installation.

```
docker run hello-world
```

The main thing to check for in the output from hello-world:

```
Hello from Docker!  
This message shows that your installation appears to be working correctly.
```

You shouldn't see any errors when running the docker hello-world smoke test.

As an installation expedient, it is recommended that you repeat the Docker installation on all the other VMs in the ICP cluster.

Things that can go wrong with Docker install

1. Your RHEL install doesn't have a `container-selinux` version greater than 2.9. See <https://stackoverflow.com/questions/45272827/docker-ce-on-rhel-requires-container-selinux-2-9>
2. Your RHEL install doesn't have yum-config-manager. You need to install yum-utils (`yum install -y yum-utils`)

Installing Docker CE using the docker install binary from Passport Advantage

If your VM doesn't have access to the public docker repository, then you can install Docker using the Docker install binary available on Passport Advantage along with the ICP install archive.

- If you don't already have it, download the Docker install binary from Passport Advantage. (It is very small so that won't take long.)
- Copy the Docker install binary to some staging directory on your VM.
- Run the following commands as root:

```
chmod +x icp_docker_17.09_rhel_x86_64.bin  
sudo ./icp_docker_17.09_rhel_x86_64.bin
```

The actual name of the docker install binary available at Passport Advantage may be different due to version differences.

- To finish things off, start and enable docker and run the hello-world smoke test as described in the above section.

- As an installation expedient, it is recommended that you repeat the Docker installation on all the other VMs in the ICP cluster.

Installing docker community edition using RPMs

In some contexts, you may not be able to get access to the Docker repo site or any other external site that has the docker packages. In that case you need to install using the RPMs.

For RHEL, get the Centos RPMS. You can get the Centos RPMs here:

https://download.docker.com/linux/centos/7/x86_64/stable/Packages/



.. /			
docker-ce-17.03.0.ce-1.el7.centos.x86_64.rpm	2017-03-01 11:10	19M	
docker-ce-17.03.1.ce-1.el7.centos.x86_64.rpm	2017-03-28 04:46	19M	
docker-ce-17.03.2.ce-1.el7.centos.x86_64.rpm	2017-06-28 03:35	19M	
docker-ce-17.06.0.ce-1.el7.centos.x86_64.rpm	2017-06-28 05:11	21M	
docker-ce-selinux-17.03.0.ce-1.el7.centos.noarch.rpm	2017-03-01 11:10	29K	
docker-ce-selinux-17.03.1.ce-1.el7.centos.noarch.rpm	2017-03-28 04:46	29K	
docker-ce-selinux-17.03.2.ce-1.el7.centos.noarch.rpm	2017-06-28 03:35	29K	

You need the SELinux RPM.

See the ICP Knowledge Center section, [Supported Docker versions](#) for a list of supported Docker versions.

1. Copy a container-selinux package to the machine where Docker is to be installed.
2. Copy the RPMs to the machine where the Docker install is to be completed.
3. The install order is container-selinux, docker-ce-selinux

This sample command line is intended to be executed in the directory where the RPMs are located:

```
yum -y install container-selinux-1.12.5-14.el7.x86_64.rpm
yum -y install docker-ce-selinux-17.03.2.ce-1.el7.centos.noarch.rpm
yum -y install docker-ce-17.03.2.ce-1.el7.centos.x86_64.rpm
```

That should complete the Docker installation.

To finish things off, start and enable docker and run the hello-world smoke test as described in the above section.

As an installation expedient, it is recommended that you repeat the Docker installation on all the other VMs in the ICP cluster.

Install Python Docker support

This section describes the steps for installing Python Docker support modules. The Docker modules that get installed allow all the usual Docker commands to be used within a Python script.

NOTE: This section may be skipped. Installing the Python Docker support modules is optional. If you don't intend to use Python for scripting of Docker operations, then this section can be skipped. It is **not** necessary to install pip as a pre-requisite to installing ICP.

NOTE: Python has a `docker` package and a `docker-py` package. The documentation gives the impression they are synonymous. However, in comparing the effects of doing the install of `docker` vs `docker-py`, they do not appear to be equivalent. The installation of the `docker` package appears to include the `docker-py` package, but not vice versa. More investigation is needed.

In order to install the Python Docker support modules, pip needs to be installed. (Pip is the Python package manager.) In order to install pip, the python-setuptools package needs to be installed.

- Install Python setup tools. (Python setuptools may already be installed.)

```
yum -y install python-setuptools Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager Package python-setuptools-0.9.8-4.el7.noarch already installed and latest version Nothing to do
```

- Install pip.

```
easy_install pip
Searching for pip
Reading https://pypi.python.org/simple/pip/
Best match: pip 9.0.1
...
Adding pip 9.0.1 to easy-install.pth file
Installing pip script to /usr/bin
Installing pip2.7 script to /usr/bin
Installing pip2 script to /usr/bin

Installed /usr/lib/python2.7/site-packages/pip-9.0.1-py2.7.egg
Processing dependencies for pip
Finished processing dependencies for pip
```

- Install Python Docker support modules.

```
pip install docker
```

After the install of the Python Docker support modules you should see the following directories:

```
> ls -l /usr/lib/python2.7/site-packages/docker*
/usr/lib/python2.7/site-packages/docker:
api  auth.py  client.py  constants.py  errors.py  __init__.py  models  tl
auth  auth.pyc  client.pyc  constants.pyc  errors.pyc  __init__.pyc  ssladapter  tl

/usr/lib/python2.7/site-packages/docker-2.5.1.dist-info:
DESCRIPTION.rst  INSTALLER  METADATA  metadata.json  RECORD  top_level.txt  WHEEL

/usr/lib/python2.7/site-packages/docker_py-1.10.6.dist-info:
DESCRIPTION.rst  INSTALLER  METADATA  metadata.json  RECORD  top_level.txt  WHEEL

/usr/lib/python2.7/site-packages/dockerpycreds:
constants.py  constants.pyc  errors.py  errors.pyc  __init__.py  __init__.pyc  stor

/usr/lib/python2.7/site-packages/docker_pycreds-0.2.1.dist-info:
DESCRIPTION.rst  INSTALLER  METADATA  metadata.json  RECORD  top_level.txt  WHEEL
```

Things that can go wrong with the Python Docker support installation

- Pip needs access to the public Internet to get the modules. Public access to the Internet may not be available in all contexts. In such cases, you need to configure a private pip repo.

MountFlags in docker.service

The Kubernetes kubelet process that runs on each VM in the ICP cluster needs `MountFlags=shared` in docker.service configuration file.

The MountFlags setting needs to be done on all machines in the cluster/cloud.

It is assumed that docker has been installed. (You won't see a `docker.service` file in `/lib/systemd/system` if docker has not been installed.)

1. Edit the file: `/lib/systemd/system/docker.service`
2. To the `Service` section, add the line:

```
MountFlags=shared
```

Starting, stopping and enabling Docker

This section has commands for starting, stopping and enabling docker and checking its status.

After Docker is installed, you need to start it. You need root privileges to start, stop and enable a service.

To start docker:

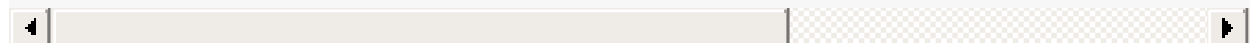
```
> systemctl start docker
```

To get docker status:

```
> systemctl status docker
```

To enable docker (so that it starts on machine reboot):

```
> systemctl enable docker  
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to
```



To stop docker:

```
> systemctl stop docker
```

Configure passwordless ssh among cluster/cloud members

The "boot master" VM needs to have root access via ssh to the other members of the cloud.

NOTE: In the description below, it is assumed that DNS is in use and the host names for the ICP cluster VMs are registered in the DNS. If DNS is not in use, then the `/etc/hosts` files on each of the ICP cluster VMs must have been set up to map host names to IP addresses. Hence, host names are used in the samples. (The `ssh-copy-id` command requires the use of host names.)

NOTE: Substitute your actual host names in the sample commands in this section. For the ICP KC instructions to do this work, see [Sharing SSH keys among cluster nodes](#)

- Login to the boot-master node as root
- On the boot-master, as root, from root's home directory (/root) execute:

```
> ssh-keygen -b 4096 -f ~/.ssh/id_rsa -N ""
```

The above command requires no responses to prompts. You should see something like:

```
Generating public/private rsa key pair.  
Your identification has been saved in /root/.ssh/id_rsa.  
Your public key has been saved in /root/.ssh/id_rsa.pub.  
The key fingerprint is:  
REDACTED root@REDACTED  
The key's randomart image is:  
REDACTED
```

- Now, executing a directory listing on /root/.ssh should show two files: `id_rsa`, `id_rsa.pub`. (A `known_hosts` file may also be present.)

```
> ls -l ~/.ssh
```



```
total 8
-rw-----. 1 root root 1675 Jun 30 12:11 id_rsa
-rw-r--r--. 1 root root 402 Jun 30 12:11 id_rsa.pub
```

- Using the `ssh-copy-id` command from root's home directory, copy the resulting `id_rsa` key file to each node in the cluster (including the boot-master node on which you are currently operating).

NOTE: The copy of the SSH ID to other hosts requires the use of the target host name in the `ssh-copy-id` command. Do not use an IP address, you will not be able to get past the authentication step when you attempt to enter the password for root on the target host.

In the command below `<master>` is used as a placeholder for the actual boot-master fully qualified host name (FQDN).

```
> ssh-copy-id -i ~/.ssh/id_rsa.pub root@<master>
```

You will be prompted to confirm that you want to connect to the `<master>`. Then you will be prompted for root's password on `<master>`, which is the target for this first passwordless ssh configuration.

NOTE: If you are not prompted to confirm that you want to connect to the target machine, and for the root password of the target machine, then make sure the target machine has a `.ssh` directory in `/root`. The permissions on the `.ssh` directory should be 700. Also make sure you can ping the target host by host name. (DNS or `/etc/hosts` on the boot-master needs to be configured to allow the target host name to be resolved.)

- Now try logging into the machine, with: `ssh root@<master>` and check to make sure that only the key(s) you wanted were added.

At this point you should see two additional files in the `.ssh` directory:

```
> ls -l ~/.ssh
total 16
-rw-----. 1 root root 402 Jun 30 12:17 authorized_keys
-rw-----. 1 root root 1675 Jun 30 12:11 id_rsa
-rw-r--r--. 1 root root 402 Jun 30 12:11 id_rsa.pub
-rw-r--r--. 1 root root 191 Jun 30 12:17 known_hosts
```

- Repeat for each additional server in the cluster/cloud. (As above, you will need to answer

yes to add the ECDSA key for each host to the known_hosts file and provide the root password of the target host.)

In the commands below, `<proxy>` and `<worker_##>` is used as a placeholder for the fully qualified host names (FQDN) for machines in the cluster.

```
> ssh-copy-id -i ~/.ssh/id_rsa.pub root@<proxy>
> ssh-copy-id -i ~/.ssh/id_rsa.pub root@<worker_01>
> ssh-copy-id -i ~/.ssh/id_rsa.pub root@<worker_02>
etc
```

- When this is complete, you should be able to ssh from the boot-master node to each of the other nodes as root without having to provide a password. You can test this by executing and ssh from the boot-master host to any of the other members of the ICP cluster:

```
[root@<master> ~]# ssh root@<proxy>
Last login: Thu Jun 29 14:44:34 2017
[root@<proxy> ~]# exit
logout
Connection to <proxy> closed.
[root@<master> ~]# ssh root@<worker_01>
Last login: Fri Jun 30 09:39:31 2017
[root@<worker_01> ~]# exit
logout
Connection to <worker_01> closed.
[root@<master> ~]#
etc
```

If you cannot gain access via SSH without a password, check the known_hosts and authorized_keys files on the hosts other than the boot-master.

Install and configure the GlusterFS server cluster

If you are not going to use GlusterFS, then this section can be skipped.

This section describes the steps for installation and configuration of a 3 node GlusterFS server cluster. The GlusterFS server cluster is not included in the ICP cluster itself in order to keep the GlusterFS servers dedicated to the role of providing a file sharing service.

Heketi is the administrative client for Gluster. The section, [Install Heketi administration client for Gluster](#Install Heketi administration client for Gluster) describes the installation and configuration of Heketi.

The instructions in this section describe how to run Gluster from a docker container. Other approaches run Gluster "natively".

NOTE: It is recommended that the GlusterFS server cluster be created prior to the actual installation of ICP. You can choose to incorporate a GlusterFS cluster after ICP is installed.

A sample resource configuration for the GlusterFS VMs is summarized in the table below.

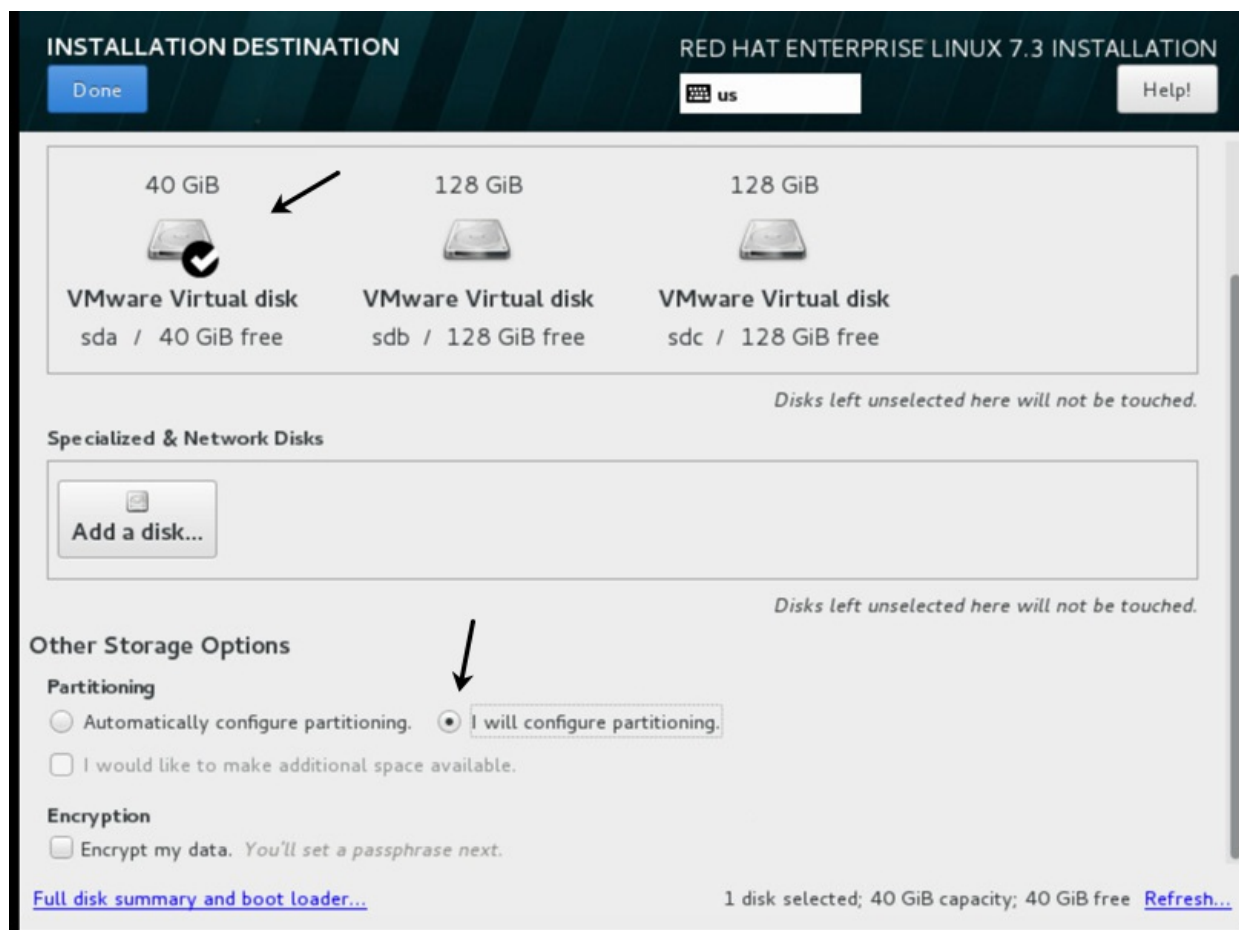
Machine Role	Number	vCPU/Core	Memory (GB)	Storage Disks x Size (GB)
GlusterFS	3+	4	16	1 x 40 (/dev/sda) 1 x 128 (/dev/sdb) 1 x 128 (/dev/sdc)

- During the VM creation using vCenter, you can add disks when you get to the "machine settings" screen. At the bottom of the screen there is a "New devices" pull-down menu. Select "Hard disk" and click the add button. (See figure below.)

Virtual Hardware	VM Options	SDRS Rules
▶ *CPU	4	
▶ *Memory	16	GB
▶ *New Hard disk	40	GB
▶ New SCSI controller	VMware Paravirtual	
▶ New Network	csplab	<input checked="" type="checkbox"/> Connect...
▶ New CD/DVD Drive	Client Device	<input type="checkbox"/> Connect...
▶ New Floppy drive	Client Device	<input type="checkbox"/> Connect...
▶ Video card	Specify custom settings	
▶ VMCI device		
▶ New SATA Controller		
▶ Other Devices		
▶ New Hard disk	128	GB
▶ New Hard disk	128	GB

New device: New Hard Disk Add

Compatibility



- The GlusterFS disks can be thin provisioned.
- You can add disks to an already deployed VM from the vCenter console by opening the "Machine Settings" dialog and adding the disks as described above.
- Only the system disk needs to be partitioned. Heketi will do all the configuration of the storage drives.

The following table is a summary of the system disk partitioning on a GlusterFS server.

File System Name	Mount Point	Size (GB)
system (aka root)	/	32
boot	/boot	256 MB
swap		8

NOTE: The remainder of these instructions assume that Ansible is set up on an administrator machine or some machine that can be used for running Ansible command lines. The Ansible hosts file is assumed to have a group defined named "gluster" with the GlusterFS servers in it.

In the example Ansible command lines, the default Ansible hosts file in `/etc/ansible/hosts` is used. You may prefer to create a hosts file and pass its path in on the command line with the `-i` option.

NOTE: It is assumed the user running Ansible has passwordless sudo root privileges on the GlusterFS servers.

NOTE: In the Ansible commands the `-b` and `--become` options are synonyms. Usually the module or shell command to be executed requires that the user privileges become elevated to root.

NOTE: The simplest instructions assume the GlusterFS server VMs have public Internet access to get to the site where Docker can pull the latest GlusterFS image. If that is not the case, we provide alternate steps that assume only that the Ansible control machine has access to the public Internet.

On each GlusterFS server in the GlusterFS cluster:

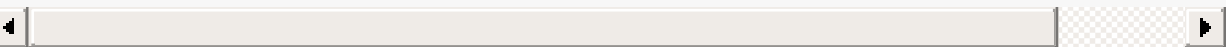
- Set up yum repository for RHEL
- Update RHEL
- Install NTP, start and enable NTPD
- Install yum-utils (if you want to use yum-config-manager)
- Install bind-utils
- Install Docker, start and enable Docker
- Stop, disable firewalld
- Create an "Ansible user", e.g., `icpmaestro` and include that user in the wheel group.
- Modify the `/etc/sudoers` file (`visudo`) to allow the wheel group sudo any command without password.
- Install the latest version of GlusterFS

```
ansible gluster -b -m shell -a "docker pull gluster/gluster-centos:latest"
```

If the GlusterFS VMs cannot get directly to the public Internet, then the following steps can be used to install GlusterFS. We're assuming your Ansible control machine has Docker installed on it.

- Pull the latest Gluster image on the Ansible control machine; save it; and push it to the GlusterFS server nodes using Ansible. (This step is a substitute for the previous step when the GlusterFS server VMs do not have access to the public Internet.)

```
docker pull gluster/gluster-centos:latest
docker save gluster/gluster-centos:latest | gzip -c > gluster-centos.tar.gz
ansible gluster -b -m copy -a 'src=gluster-centos.tar.gz dest=/tmp/gluster-centos.t
ansible gluster -b -m shell -a 'tar -xf /tmp/gluster-centos.tar.gz' -0 | docker loa
```



- To check that the gluster image got loaded:

```
$ ansible gluster -b -m shell -a 'docker images'
```

- Create a `/var/lib/heketi` directory on the GlusterFS machines. Gluster mounts are to be persisted in `/var/lib/heketi/fstab` on each host.

```
$ ansible gluster -b -m file -a 'path=/var/lib/heketi state=directory'
```

- Start docker in privileged mode running gluster. (Most of the command below is bind mounting container "volumes" to host directories.) (For details on options used in the docker run command below, see Docker documentation: [docker container run](#))

```
$ ansible gluster -b -m shell -a 'docker run --restart always -v /etc/glusterfs:/et
```



- To check that gluster is running on each GlusterFS machine, you can use the following command:

```
$ ansible gluster -b -m shell -a 'docker ps -a'
```

NOTE: For information about working with RHEL 7 kernel modules see the Red Hat

documentation section [Working with Kernel Modules](#)

- Configure kernel to use the `dm_thin_pool` module. (The `dm_thin_pool` module supports LVM thin provisioning. Kernel modules to be loaded at startup are listed in `*.conf` files in `/etc/modules-load.d/`.)

```
ansible gluster -m shell -a 'modprobe dm_thin_pool' --become
ansible gluster -m shell -a 'echo dm_thin_pool | tee -a /etc/modules-load.d/modules
```

If you want to check that `dm_thin_pool` got loaded in the docker image repository:

```
$ ansible gluster -b -m shell -a 'lsmod | grep dm_thin_pool'
gluster03.zzz.yyyy.xxx | SUCCESS | rc=0 >>
dm_thin_pool          65565  0
dm_persistent_data    67216  1 dm_thin_pool
dm_bio_prison         15907  1 dm_thin_pool
dm_mod                114430  11 dm_log,dm_persistent_data,dm_mirror,dm_bufio,dm_th

gluster01.zzz.yyyy.xxx | SUCCESS | rc=0 >>
dm_thin_pool          65565  0
dm_persistent_data    67216  1 dm_thin_pool
dm_bio_prison         15907  1 dm_thin_pool
dm_mod                114430  11 dm_log,dm_persistent_data,dm_mirror,dm_bufio,dm_th

gluster02.zzz.yyyy.xxx | SUCCESS | rc=0 >>
dm_thin_pool          65565  0
dm_persistent_data    67216  1 dm_thin_pool
dm_bio_prison         15907  1 dm_thin_pool
dm_mod                114430  11 dm_log,dm_persistent_data,dm_mirror,dm_bufio,dm_th
```

At this point, the GlusterFS server cluster is up and running and you can proceed with the ICP installation.

Install Heketi on RHEL (aka "native" install)

This section describes the installation of Heketi on RHEL. Heketi and the Heketi client will be running directly on the VM rather than in Kubernetes pods.

Heketi 5.0.0 is installed using the GlusterFS 3.13 yum repository.

Things that can go wrong with the Heketi install

unknown shorthand flag: 'c'

With Heketi 4.0.0, the service would fail to start. The error in `/var/log/messages` was:

```
Error: unknown shorthand flag: 'c' in -config=/etc/heketi/heketi.json
pvs-master01 heketi: unknown shorthand flag: 'c' in -config=/etc/heketi/heketi.json
```

There is nothing actually wrong with the `heketi.json`. (You can paste the content into a JSON validator to convince yourself.)

The problem is the `-config` option to the Heketi executable.

See https://bugzilla.redhat.com/show_bug.cgi?id=1439120

You need to edit the `/usr/lib/systemd/system/heketi.service` file and change the `-config` to `--config`.

Things that can go wrong with Heketi CLI

Error: Unable to get topology information: Unknown user

TBD - This is caused by not having a `--user` argument or `HEKETI_CLI_USER` set.

Error: Unable to get topology information: signature is invalid

This error may occur when you run:

```
heketi-cli topology load --json=topology.json
```

The above assumes the `topology.json` file is in the current directory.

The "signature" in question has nothing to do with the content of the `topology.json` file or some digital signature it might be missing.

When you run an `heketi-cli` command, and you have JWT authentication enabled (see your `heketi.json`), you need to provide a user and "key" (aka password or secret). One way to provide the key is to set the environment variable, `HEKETI_CLI_KEY`. You can also pass it in on the `heketi-cli` command line with the `--secret` option. (See `heketi-cli --help` usage info.)

If the password/secret you provide does not match up with the user and key in the `heketi.json` file in current use by the Heketi service, then you will get the "signature is invalid" error.

NOTE: If you change anything in the `heketi.json` file you need to restart the Heketi service (`systemctl restart heketi`) to pick up the changes.

Install Heketi administration client for Gluster

NOTE: The installation of Heketi is very confusing. This section is currently under review and should be taken as a collection of notes rather than verified guidance.

If you are not using Gluster for the shared file service, this section can be skipped.

This section describes the steps to install the Heketi administration client for Gluster.

A containerized Heketi is installed. Another option is to do a "native" Heketi installation.

Public Heketi install guide: [Heketi Install for Kubernetes](#)

The Heketi client is installed on the boot-master machine. (A Heketi client can be installed where you prefer, including on an administrator's desktop machine. The Heketi client obviously must have network access to the Gluster servers to be managed.)

- Follow the instructions for the Heketi install in the `install-kubernetes.md` doc (link above). (Ignore the `glusterfs` installation instructions. The `glusterfs` install was done into docker containers on each of the gluster servers. (See above section of this guide.) Important to note that the `glusterfs` install uses docker only, not kubernetes. *TBD:* Should we create a separate kubernetes cluster for `glusterfs`? That seems like overkill. I would do a "native" `glusterfs` install instead.)

- Get the [Heketi CLI](#) for the current release. (*TBD*: The instructions mention that this has heketi-cli in it. However, it is vague as to what it means to "install" the heketi-cli. I'm not sure what it is used for.)
- Extract the archive on the machine where are doing the installation. (Created a heketi directory in /root on master01 and extracted the archive there. `tar -xvf heketi-client-v5.0.1.linux.amd64.tar.gz` . The root of everything in the archive is `heketi-client` , so no need to create a separate directory where the archive is extracted.)
- *TBD*: Also cloned the heketi git hub. There appears to be more useful content in the git repo than what comes with the release tar ball. (The following clone command was executed from the /root/heketi staging directory.) Is this a reasonable step or unnecessary? Or should this be done instead of downloading and extracting the heketi tar ball? The heketi release tar ball has the heketi-cli executable in it. But I'm not sure what that is used for.

```
git clone https://github.com/heketi/heketi heketi-git-hub
```

- Create a kubernetes service account.

```
kubectl create -f heketi-client/share/heketi/kubernetes/heketi-service-account.json
```

- Create a cluster role binding for the service account so it can administer the gluster servers.

```
kubectl create clusterrolebinding heketi-gluster-admin --clusterrole=edit --service
```

- Create kubernetes secret:

```
kubectl create secret generic master01-root-ssh-key --from-file=/root/.ssh/id_rsa
```

TDB: What/where does this secret get used? Looks like it should be referenced in the heketi.json config file, but I'm not sure how yet. That's where the heketi executor is set and in

this case `ssh` should be used for the executor.

- Make a copy of the `heketi.json` to edit for the install. (*TBD*: There are a number of `heketi.json` files in the git repo. I started with the one in `<repo-clone>/etc/heketi.json`)
- Things I changed in `heketi.json` :
 - Server port: 8081 (8080 is already in use on master01, *TBD*: Find out what is using 8080.)
 - `use_auth`: True
 - admin key, user key (*TBD*: Are these two secrets a password? Or the name of a kubernetes secret? Looks like they are supposed to be a password.)
 - glusterfs executor: `ssh`
 - user: root (*TBD*: Likely need to create a separate heketi user rather than use root.)
 - key file: `/root/.ssh/id_rsa`
 - `fstab`: (*TBD* Not sure what to use here. On the gluster server nodes we created a `/var/lib/heketi/` where `fstab` is intended to go. So for now, use `/var/lib/heketi/fstab`.)
 - kubernetes exec (`kubeexec`)
 - host: `https::8443`
 - cert: don't care (*TBD*: What would this be in a more realistic configuration.)
 - `insecure`: true (*TBD*: Should be false in a realistic deployment.)
 - user: admin
 - password: admin
 - namespace: default (*TBD*: Not sure what this should be. Maybe `service`)
 - `fstab`: `/var/lib/heketi/fstab` (*TBD*: Need to confirm this is the correct path.)
 - `auto_create_block_hosting_volume`: false (*TBD*: Confirm this is correct. I'm pretty

sure we don't want this.)

- `block_hosting_volume_size`: 100 (*TBD*: I'm pretty sure the units for this are GB. So even if we want automatic creation of block hosting volume, the default of 500 GB is likely too large.)
- The next step in the heketi install guide is to create a secret based on the `heketi.json`. (*TBD*: Not really sure what it means to use that whole config file to create a "secret". How is that secret used?)

```
kubectl create secret generic heketi-config-secret --from-file=./heketi.json
```

- Make a copy of `heketi-bootstrap.json` that comes in the `<repo-clone>/extras/kubernetes/`
- Edit the `heketi-bootstrap.json` file to make sure it is what you want.
 - Originally I had replicas set at 2. But then realized that for the heketi bootstrap service, likely I only need 1 replica.
 - I changed all the port 8080 to 8081. (*TBD*: Need to confirm.)
 - There is an env var `HEKETI_EXECUTOR` that I changed to `ssh` from `kubernetes`.
 - The names of some other things are in this file. They match the names used in earlier commands. If different names are used, then corresponding changes would need to be made in this file.
- Then do the following:

```
kubectl create -f heketi-bootstrap.json
```

- The first run of the heketi-bootstrap create failed. Below is the event list from `kubectl describe pods`:

Events:					
Type	Reason	Age	From	Message	
----	-----	----	----	-----	
Normal	Scheduled	4m	default-scheduler	Success	
Normal	SuccessfulMountVolume	4m	kubelet, 172.16.249.84	MountVo	
Normal	SuccessfulMountVolume	4m	kubelet, 172.16.249.84	MountVo	
Normal	SuccessfulMountVolume	4m	kubelet, 172.16.249.84	MountVo	
Warning	FailedCreatePodSandBox	4m (x7 over 4m)	kubelet, 172.16.249.84	Failed	
Normal	SandboxChanged	4m (x7 over 4m)	kubelet, 172.16.249.84	Pod san	



- Delete things you need to do the following:
 - Delete the heketi deployment: `kubectl delete deployment deploy-heketi`
 - Deleting the service as well: `kubectl delete service deploy-heketi`
 - NOTE: Deleting the deployment, deletes the pod(s), but it doesn't delete the service.
(TBD: Not sure why deleting a deployment does not delete the service.)
- Second try with only 1 replica.

```
kubectl create -f heketi-bootstrap.json
```

Install IBM Cloud Private v2.1

This is the start of the description of the steps to install IBM Cloud Private.

Prerequisite steps to Install IBM Cloud Private v2.1

1. If you are building your own VM, you should have already created a clone of the ICP base VM for each VM in the cluster. At a minimum, this would be the boot-master, the proxy and a couple of worker nodes.
2. If you are using VMs provided to you, then you should have completed all the steps to getting Docker running on at least the boot-master machine. The install process includes installing Docker on each cluster member. As an expedient we recommend pre-installing Docker on each cluster member. (See the section [Copy and load ICP docker image tar ball to all cluster VMs](#Copy and load ICP docker image tar ball to all cluster VMs) below for details.)
3. DNS or the `/etc/hosts` file on each VM should be configured with the proper entries so that each VM can resolve the address of the other members of the cluster/cloud.
4. SSH needs to have been set up such that the "boot master" VM can ssh to each of the other VMs in the cluster/cloud as root without using a password.
5. The product install image should be available on the boot-master machine. If not,

download the product archives from Passport Advantage (IBM customer) or eXtreme Leverage (IBM internal). You can find the GA release by searching on, *IBM Cloud Private*.

6. The ICP Knowledge Center (KC) installation instructions for ICP Cloud Private Enterprise are rooted in the section: [Intalling ICP Cloud Private Enterprise](#).
7. On all VMs in the ICP cluster, if firewalld is running, stop it and disable it until after the ICP install completes.

To see if firewalld is running, use:

```
> systemctl status firewalld
```

To stop and disable firewalld, use:

```
> systemctl stop firewalld
> systemctl disable firewalld
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
Removed symlink /etc/systemd/system/basic.target.wants/firewalld.service.
```

NOTE: The firewall only needs to be disabled during install. It gets enabled again on all members of the cluster/cloud after the install has completed.

NOTE If you are creating an ICP cluster with VMs (members) on more than one network segment/VLAN, then there may be physical firewalls that need to be configured to allow the ICP installation to proceed. See the ICP Knowledge Center section, [Default ports](#), for the list of ports that must be open for installation and configuration of an ICP instance.

Some additional "boot master" pre-installation steps

This section has some steps that need to be taken on the boot master before the actual installation command can be run.

NOTE: In these instructions, the root directory of the installation is referred to as `<ICP_HOME>`. A common convention is to install ICP in a directory that includes the ICP version in the directory name, e.g., `/opt/icp2.1`.

- It is assumed that Docker is installed and running on the boot-master machine.

- It is assumed that the ICP images archive has been loaded into the Docker registry on the boot-master machine. (NOTE: The actual archive file name may be different depending on the version of ICP you are installing.)

```
tar -xf ibm-cloud-private-x86_64-2.1.0.1.tar.gz -0 | docker load
```

- (On the boot-master) Extract the ICP boot meta-data to the `<ICP_HOME>/cluster` directory:

```
> cd <ICP_HOME>
> docker run -v $(pwd):/data -e LICENSE=accept ibmcom/icp-inception:2.1.0.1-ee
```

NOTE: You may need to use a different version tag for the `icp-inception` image. Use `docker images | grep icp-inception` to see the version tag in your image repository.

The above command creates a directory named `cluster` in `<ICP_HOME>`. The `cluster` directory has the following contents:

```
> ls -l cluster
-rw-r--r--. 1 root root 3998 Oct 30 06:37 config.yaml
-rw-r--r--. 1 root root 88 Oct 30 06:37 hosts
drwxr-xr-x. 4 root root 39 Oct 30 06:37 misc
-r----- 1 root root 1 Oct 30 06:37 ssh_key
```

- Add the IP address of all the cluster/cloud members to the `hosts` file in `<ICP_HOME>/cluster`.

NOTE: The ICP hosts file must use IP addresses. Host names are not used.

- Copy the ssh key file to the `/cluster`. (This overwrites the empty `ssh_key` file already there.)

```
'''
```

```
cp ~/.ssh/id_rsa ssh_key
cp: overwrite 'ssh_key'? y
```

```
'''
```

- Check the permissions on the `ssh_key` file and make sure they are read-only for the owner

(root). If necessary, change the permissions on the ssh_key file in `<ICP_HOME>/cluster` to "read-only" by owner, i.e., root.

- Check the access:

'''

```
ls -l ssh_key
-r-----. 1 root root 1675 Jun 30 13:46 ssh_key
'''
```

- If the access is not read-only by owner, then change it:

'''

```
chmod 400 ssh_key
'''
```

- Check again to make sure you changed it correctly.
- Copy/move the "image" archive (`ibm-cloud-private-x86_64-2.1.0.1.tar.gz`) to the images directory in `<ICP_HOME>/cluster` . (You first need to create the images directory.) In the command below it is assumed the image archive is located initially in `<ICP_HOME>` .

From `<ICP_HOME>/cluster` :

```
> mkdir images
> mv `<ICP_HOME>/ibm-cloud-private-x86_64-2.1.0.1.tar.gz` images
```

Working with the config.yaml file is described in the next section.

Configuring config.yaml on the boot master

For information on the content of config.yaml, see the ICP KC section, [Cluster configuration settings](#).

For a simple sandbox deployment, the content of config.yaml can remain as is.

NOTE: The `network_cidr` and `service_cluster_ip_range` are set to "10." IP networks. If your cloud provide is using that same address range, then change the values to something else, e.g., some other "10." subnet or the "172.16." networks.

Things that can be left as-is for a small sandbox environment:

- network_type calico
- network_cidr: 10.1.0.0/16
- service_cluster_ip_range: 10.0.0.1/24
- For a simple cluster, everything else in config.yaml remains commented out.

There are many parameters that may be set in config.yaml. It is a good idea to read through the file to become familiar with the options.

Additional things that need to be set for a production environment:

- vip_iface, cluster_vip
- proxy_vip_iface, proxy_vip

NOTE: Gluster configuration in config.yml is not necessary when the Gluster servers are set up outside the ICP cluster, which is the topology used in this guide. **No not** do any Gluster configuration in config.yml.

Other things that you may need to check

This section has a collection of items that have led to a failure in the installation process. This is a work in progress and is a place to keep track of this sort of stuff that seems a bit random.

- Make sure all the VMs in the cluster/cloud are running.

You may want to double check the following on each VM that is a member of the cluster:

- The network interface on each VM is started.
- The firewall on each VM is disabled.
- If you pre-installed Docker on each VM, then check that Docker is running on each VM.
- Docker must be installed and running on the boot-master VM.
- The ICP docker images must be loaded into the Docker registry on the boot-master VM.

Copy and load ICP Docker images tar ball to all cluster VMs

This section assumes that Docker is pre-installed on all of the cluster member VMs. Installing Docker on each VM uses the same steps as installing Docker on the boot-master VM.

It is expedient to pre-load the Docker registry on each VM that is a member of the ICP cluster. The installation process run from the boot-master machine will recognize that the Docker registry is up-to-date on the other cluster member machines and skip the step of copying and

loading the image tar ball to the Docker registry on the given machine. The gain in the time it takes to load the registry is achieved because you can open as many shells as needed to do the copy and load operations concurrently.

- Copy the ICP image tar ball to all machines. (You can open multiple shells on the boot-master machine and start an `scp` of the image tar ball to each machine in the cluster.)
- Open a shell on each VM in the cluster and extract the docker images and load them into the docker registry:

```
tar -xvf ibm-cloud-private-x86_64-2.1.0-beta-2.tar.gz -0 | docker load
```

If you run out of file system space during the above command, use `df -h` to view file system utilization. You can use `df -ih` to view inode utilization. Make sure the file systems are adequately provisioned as described in the [Check the file system sizing] (#Check the file system sizing) section above.

- On all but the boot-master machine, the ICP image tar file can be removed once the docker load completes. On the boot-master machine the ICP image tar file gets moved to an images directory in `<ICP_HOME>/cluster`.

NOTE: If you are building an ICP virtual machine, you can run the extract and load on the base VM image so that all cloned VMs have the docker registry pre-loaded with the ICP images.

No space left on device

During the load of the ICP images you may get something like:

```
""Error processing tar file(exit status 1): write
```

```
/opt/ibm/wlp/output/.classCache/C280M4F1A64P_liberty-root_G33: no space left on device
```

Do a ``df -H`` to see what is happening with file space.

Run the ICP install command

Docker is used to run the install for all members of the cluster/cloud. The command

NOTE: It is assumed Docker is installed on the boot-master VM.

NOTE: It is assumed the ICP v2.1 images have been loaded into the local docker registry

NOTE: In the docker commands below, `$(pwd)` is the current working directory of the

NOTE: It is OK to run this command multiple times to get things installed on all

NOTE: During **the** installation all information messages go **to** stdout/stderr. If y

```
cd /cluster
mkdir logs
docker run --net=host -t -e LICENSE=accept -v $(pwd):/installer/cluster ibmcom/icp-
inception:2.1.0.1-ee install -v | tee logs/install-1.log
```

A common convention (**not** shown here) is **to include a date** stamp **in the file** name of

NOTE: If you need **to get** more detail **for** installation problem determination purpose

```
docker run -e LICENSE=accept --net=host --rm -t -v $(pwd):/installer/cluster
ibmcom/icp-inception:2.1.0.1-ee install -vvv | tee logs/icp_install-2.log
```

- When the **install** completes, you want **to** see all **"ok"** and **no "failed"** in the recap

PLAY RECAP *

```
xxx.xx.xxx.50 : ok=109 changed=36 unreachable=0 failed=0
xxx.xx.xxx.52 : ok=109 changed=36 unreachable=0 failed=0
xxx.xx.xxx.57 : ok=137 changed=36 unreachable=0 failed=0
xxx.xx.xxx.60 : ok=163 changed=55 unreachable=0 failed=0
localhost : ok=216 changed=114 unreachable=0 failed=0
```

- Problem determination is based on the installation log. The error messages are r
- Assuming the **install** went correctly **move on to some** basic **"smoke tests"** descri
- Once the installation completes successfully **and** you have performed **some** basic

Start and enable the firewalld **on** all cluster members

You may want **to** hold **off on** this step **until some** basic smoke tests have been execut

After the **install** completes, configure firewall **rules on each** cluster **member** accord

Simple ICP **"smoke"** tests

This **section** documents **some** basic **measures to confirm** correct ICP operation.

1. The simplest "smoke test" is to fire up the ICP admin console:

```
https://<boot_master>:8443/
```

Default user ID and password: admin/admin

2. Check that all processes are "available". In the ICP admin console you can s

![[ICP Admin Console Workloads Link](images/ICP_Console_Workloads.png)]

Troubleshooting installation issues

This section is a holding area for a collection of troubleshooting tips.

1. Install using the ``-vvv`` and piping output to tee to a log file is the first
2. Make sure the docker images repository has the correct images in it. For exam

Uninstall IBM Cloud Private

This section describes the steps to uninstalling ICP. You may need to do this if t

```
docker run -e LICENSE=accept --net=host --rm --name=installer -t -v
$(pwd):/installer/cluster ibmcom/icp-inception:2.1.0-ee uninstall
```

The following directories should **not** exist on any nodes in the cluster/cloud:

```
/var/lib/etcd/
/var/lib/kubelet/
/etc/cfc/
```

Miscellaneous useful docker commands

This section holds a collection of sub-sections where various useful docker command

Getting a list of docker container status

The ``docker ps`` command is likely one of the first commands you will want to know a

```
docker ps
```

or

```
docker ps -a
```

use

```
docker help ps
```

to get more information on ps options.

Getting a shell inside the container

The command **below** will open a shell console **in** a container **for the given** docker ima



```
docker run -e LICENSE=accept --net=host --rm -it -v "$(pwd)":/installer/cluster --
entrypoint=/bin/bash ibmcom/icp-inception:2.1.0-ee
```

Getting a list of local docker registry content

It **is** useful **to get** a **list** of what **is in the local** docker registry. (The term regi

The docker command **is** simple:



docker images

Once ICP is loaded **into the local** docker repository, there are **a lot of** images. You

Installing Ansible

Ansible is very useful **for** administration **of a** collection **of** machines such **as an** IC

See **the** Ansible documentation **for** [installation instructions](<http://docs.ansible.c>

Your RHEL yum repository is likely **to** have **the** Ansible RPM **in the** **"extras"** director

If you have access **to the** public Internet you can **get** Ansible RPMs via yum, here: **h**

To **add the** public Ansible yum repo **do**:



```
yum-config-manager --add-repo
```

https://releases.ansible.com/ansible/rpm/release/epel-7-x86_64/

The above command should add a repo to `/etc/yum.repos.d/` that will be named somet

```
ls /etc/yum.repos.d/  
docker-ce.repo redhat.repo releases.ansible.com_ansible_rpm_release_epel-7-  
x86_64_.repo rhel.repo
```

Yum will not access the public Ansible repository unless you **install** its **public** key

```
[releases.ansible.com_ansible_rpm_release_epel-7-x86_64_]  
name=added from: https://releases.ansible.com/ansible/rpm/release/epel-7-x86\_64/  
baseurl=https://releases.ansible.com/ansible/rpm/release/epel-7-x86\_64/  
gpgcheck=0  
enabled=1
```

Configuring Ansible

This section describes some very basic steps required **to** **get** Ansible configured **to**

The Ansible documentation **is** very good. Use it **to find** answers **to** your questions.

Some things **to** decide when using Ansible:

- The machine (**or** machines) **to be an** Ansible "**control machine**", e.g., administrator
- What user **to set up on** the control machine(s) **and all** of the managed nodes that w

The primary configuration tasks:

- Make sure each managed node **has** the Ansible user ID defined. The Ansible user wil
- Create **an** `ssh` key **for** the Ansible user **on** the control machine.
- Use `ssh-copy-id` (**or** something that does the equivalent) **to get** the public key (
- Set **up** the Ansible `hosts` (in `/etc/ansible/` by default) **file to** defined the man

NOTE: Use fully qualified domain names (FQDN) **for** the hosts when using the `ssh-c`

NOTE: A simple way **to** provide the Ansible user with passwordless sudo privileges

Allows people in group wheel to run all commands

%wheel ALL=(ALL) ALL

Same thing without a password

%wheel ALL=(ALL) NOPASSWD: ALL

```
# Kubernetes in a nutshell
```

The following collection of sub-sections is intended to provide enough information

```
## Installing kubectl on the ICP boot-master node
```

Copying kubectl from a container to the host /usr/local/bin

It is very useful to be able to have kubectl available at the shell on the ICP boot

You can get kubectl from the kubernetes container already installed as part of ICP.

```
docker run --net=host -v /usr/local/bin:/data ibmcom/kubernetes:v1.8.3-ee cp /kubectl /data
```

```
## Kubernetes basics
```

The command line tool for working with Kubernetes is kubectl.

Kubernetes [Getting Started](<https://kubernetes.io/docs/reference/generated/kubectl>)

A quick source of kubectl help: [Kubernetes kubectl "cheat sheet"](<https://kubernetes.io/docs/reference/generated/kubectl>)

When working with kubectl you need to be authenticated in your shell. Authentication

To set up authentication in your shell:

1. Log into the ICP console. The client setup command stream is available under the
2. Click on the user name, then click on "Configure client" and in the pop-up window
3. Do a paste in the shell window you are using. At that point, the shell is configured

The authentication token is good for 12 hours. You have to log out of the ICP console

```
### Get a list of pods
```

kubectl get pods


```
### Get info about a pod
```

kubectl describe pod

The **describe** command **is** particularly useful **on any of** the different kinds **of** kubern

```
### Get a list of deployments
```

kubectl get deployments

```
# RHEL 7 network interface overview
```

This section describes **some** basic information **about** networking **for** RHEL **7**.

RHEL **7** networking **is** controlled **by the** Network Manager (network service). Normally

The usual `systemctl` commands are used **for** dealing **with the** network service.

systemctl status|start|stop|restart network

For example:

systemctl status network

network.service - LSB: Bring up/down networking

Loaded: loaded (/etc/rc.d/init.d/network; bad; vendor preset: disabled)

Active: active (exited) since Thu 2017-06-29 16:51:39 EDT; 2 days ago

Docs: man:systemd-sysv-generator(8)

...

Even though **the** network service may be **running**, **that does not** mean a particular net

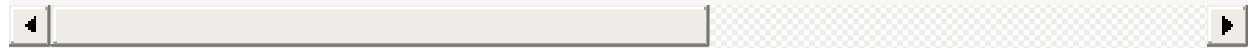
To view **the** status **of the** network interfaces **on** a machine (**and to get the names of**

ifconfig -a

The commands for starting and stopping a network interface are `ifup` (**start**), `ifdown`

For example, suppose the **name of the interface** is `ens33`.

To **stop** the `ens33` **interface**:



```
ifdown ens33
```

```
Device 'ens33' successfully disconnected.
```

To **start** the `ens33` **interface**:

```
ifup ens33
```

```
Connection successfully activated (D-Bus active path:
```

```
/org/freedesktop/NetworkManager/ActiveConnection/5)
```

`` The `ifup` and `ifdown` commands work through the Network Manager (i.e., the network service).

For a lot more details on RHEL 7 networking see the various sections in the [Introduction to Red Hat Enterprise Linux Networking](#) chapter of the RHEL documentation.

RHEL system parameters overview

This section provides some background how system parameters are managed.

RHEL system parameters can be configured in several places with a well-defined precedence. The `/etc/sysctl.conf` file is intended for use by the "local" system administrator. Parameter settings in `/etc/sysctl.conf` have the highest precedence with respect to the value settings. Other locations where system configuration parameters are read in order of precedence are: `/etc/sysctl.d/*.conf`, `/run/sysctl.d/*.conf` and `/usr/lib/sysctl.d/*.conf`. There are other locations for system configuration parameters as well. See the man pages for `sysctl` and `sysctl.d` for more details.

NOTE: It is recommended that configuration parameter file names have a leading 2-digit number followed by a dash in order to clearly indicate the ordering in which the files should be processed at the time the machine is booted. (The files are processed in the lexical ordering of their names.)

The following observations apply to a default RHEL image. The specific VM you are using may

be configured differently.

- On a default RHEL image, the `/etc/sysctl.conf` file has nothing in it, and comments in `/etc/sysctl.conf` refer to using files in `/usr/lib/sysctl.d`.
- On a default RHEL image, `/usr/lib/sysctl.d` has three files:

```
ls /usr/lib/sysctl.d/ 00-system.conf 50-default.conf 60-libvirtd.conf
```
- On a default RHEL image, none of the above files has anything in it having to do with `vm.max_map_count`.
- On a default RHEL image, there is nothing in `/run/sysctl.d/` (The `sysctl.d` directory does not exist.)
The `/etc/sysctl.d` has only the file, `99-sysctl.conf`, but that file has nothing in it except the preface comments.

Installing RHEL 7 - a sample

For those of you who may need to build your own VMs, this section provides a step-by-step look at installing RHEL v7 on a VM created in a vCenter cluster. Obviously, some details may be different due to differences in hypervisor or your specific environment. The intent of this section is to provide enough guidance to get you through the RHEL 7 installation process, particularly if you are not an experienced RHEL system administrator.

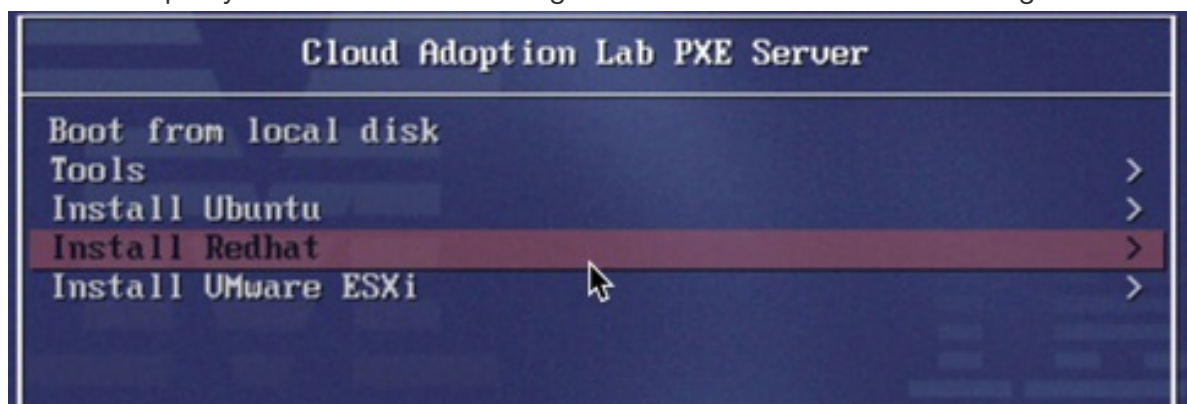
- In a vCenter console, a VM is initially created in the "Hosts and Clusters" navigation pane. You can right-click on a cluster folder and select "New Virtual Machine...". Follow the bouncing ball from there. You will shortly get to a screen that lets you configure the VMs resources. (See figure below.)

Virtual Hardware	VM Options	SDRS Rules
▶ *CPU	2	
▶ *Memory	8192	MB
▼ *New Hard disk	250	GB
Maximum Size	1.64 TB	
VM storage policy	None	
Location	Sandbox	
Disk Provisioning	<input type="radio"/> Thick provision lazy zeroed <input type="radio"/> Thick provision eager zeroed <input checked="" type="radio"/> Thin provision	

NOTE: A more realistic ICP master node resource allocation is 8 core, 32 GB of memory, 250 GB storage. We used smaller core and memory for this sample deployment. See the resource allocation table in section [A production ICP installation in a nutshell](#A production ICP installation in a nutshell) for suggested production deployment resource allocations for the various ICP machine roles.

NOTE: When creating VMs in a "cloud" environment, it is usually appropriate to select "thin provisioning" for the disks as shown in the figure above.

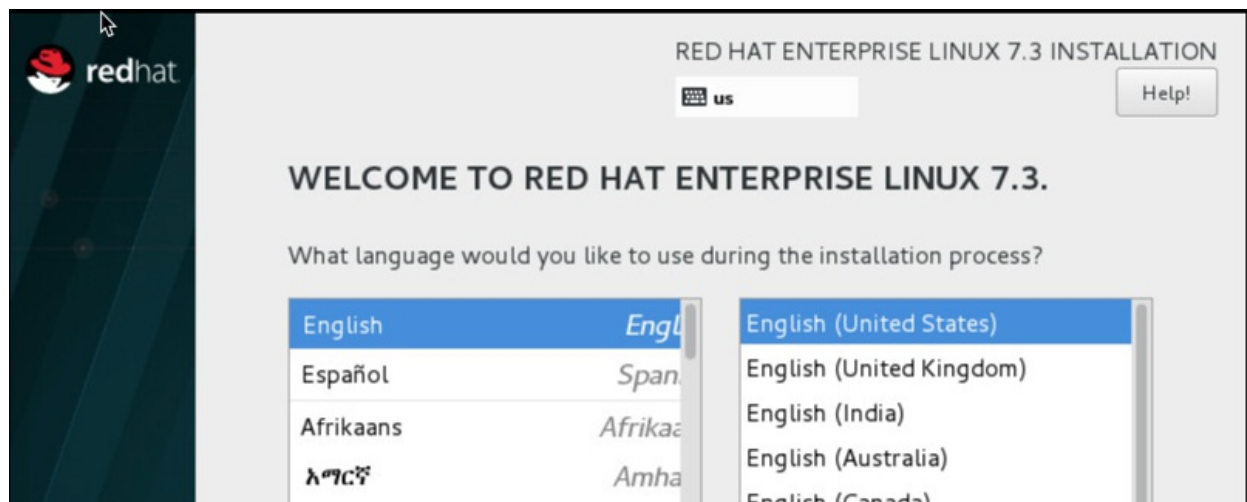
- Continue to follow the bouncing ball until you hit finish.
- Power on the VM and right-click on it and select the "Open console" item. Once the console is open you should see something in the console that looks like the figure below.



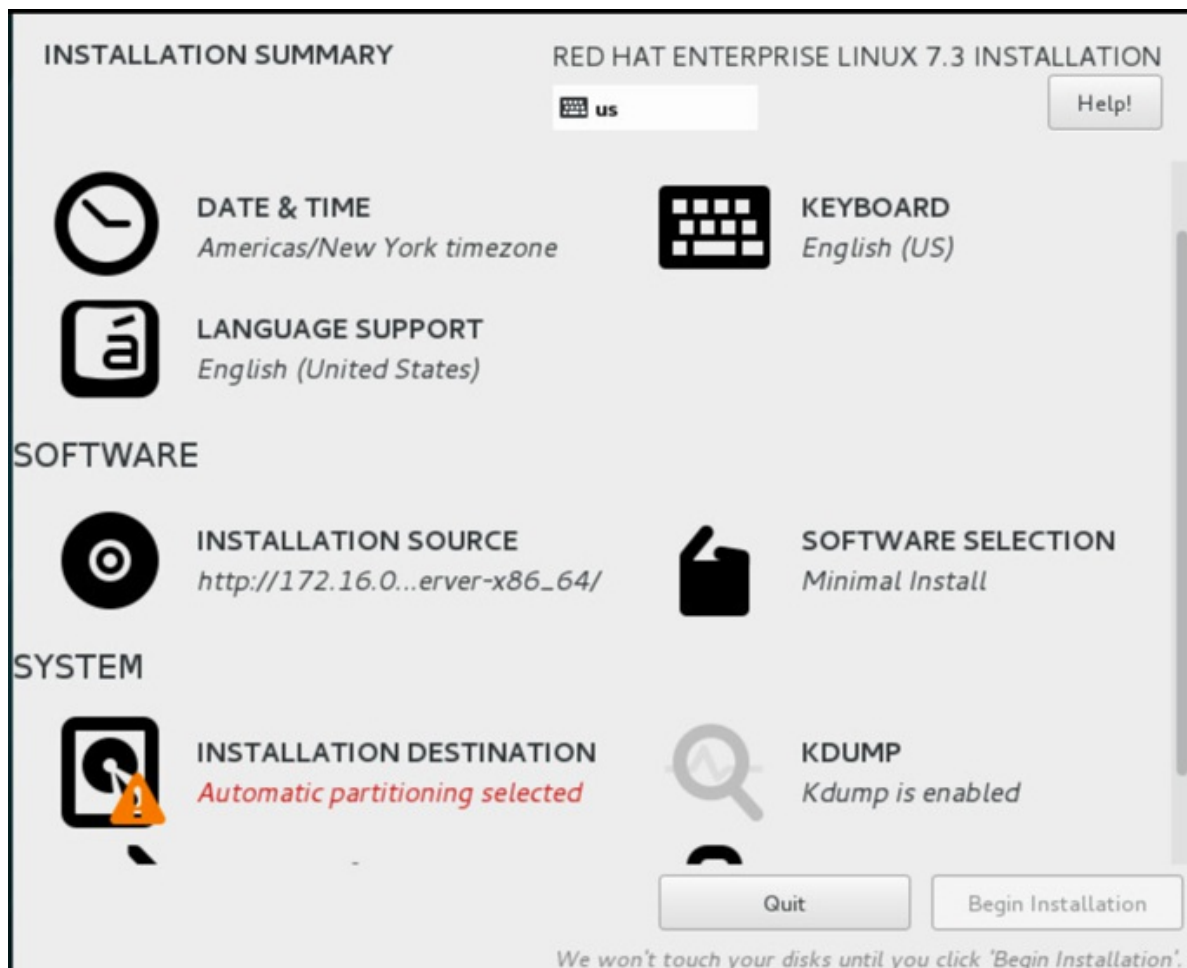
- Choose the most recent version of RHEL available.



- A lot of logging will scroll by on the console as some initial installation steps occur.
- You will be presented with a screen where you can choose the language you want to use. (See figure below.)



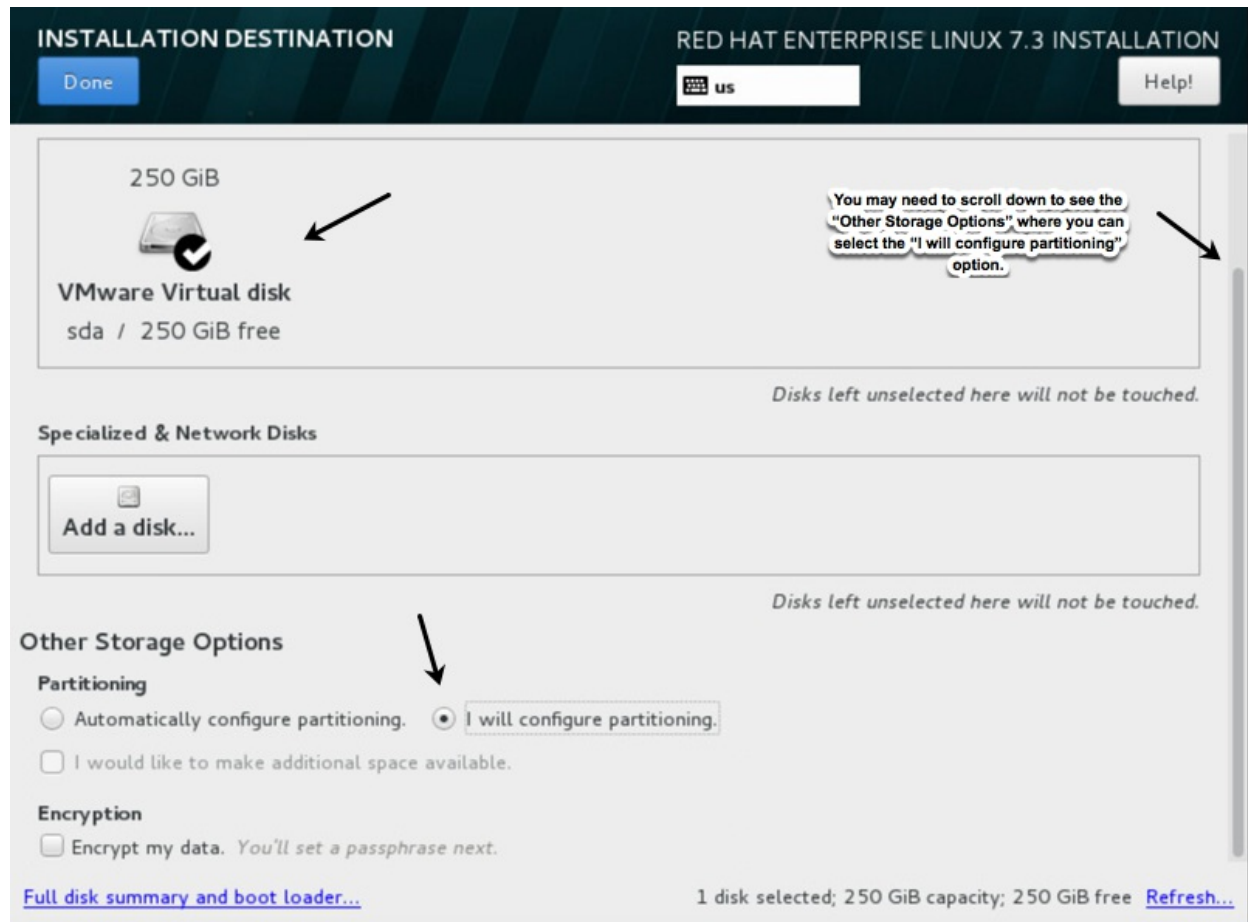
- Once you choose the language, click on the Continue button in the lower right corner.
- The next screen is where all the action is. (See figure below) You will likely need to scroll to see all the installation configuration options.



- Things to take particular note of in this screen are:
 - Software Selection: For an ICP node a "minimal install" is sufficient. Anything else you need can be picked up from a RHEL yum repository. A minimal install keeps the VM small and fast to boot. If you want to look at the other options click on the Software Selection icon.
 - Installation Destination: This is where you configure disk partitions and file system layout. See RHEL 7 documentation section: [Installation Destination](#) for guidance on disk partitioning. Consult your operations team for the standard disk partitioning and file system layout used for RHEL images. See the subsection below for steps for laying out the file system partitions.
 - Security Policy: (In this sample, no security profile is selected. Use of the security profiles is currently beyond the scope of this installation guide.)
 - Network and Host Name: You will likely want to choose a non-default host name. Consult your operations team to find out what host naming conventions are used in your data center. This is also where additional NICs can be configured. For this sample only 1 NIC was configured.

RHEL v7 installation disk partitioning

When you select "Installation Destination" in the installation summary screen, you will see the option, "I will configure partitioning". You may need to scroll down to expose that option. (See figure below.)



- Once you have indicated that you want to customize the disk partitions, click on the blue Done button in the upper left corner.
- The next screen will allow you define multiple mount points and file system size for each mount point. (See figure below.)

MANUAL PARTITIONING

Done

New Red Hat Enterprise Linux 7.3 Installation

You haven't created any mount points for your Red Hat Enterprise Linux 7.3 installation yet. You can:

- [Click here to create them automatically.](#)
- Create new mount points by clicking the '+' button.

New mount points will use the following partitioning scheme:

LVM

+

-

↻

AVAILABLE SPACE

250 GiB

TOTAL SPACE

250 GiB

[1 storage device selected](#)

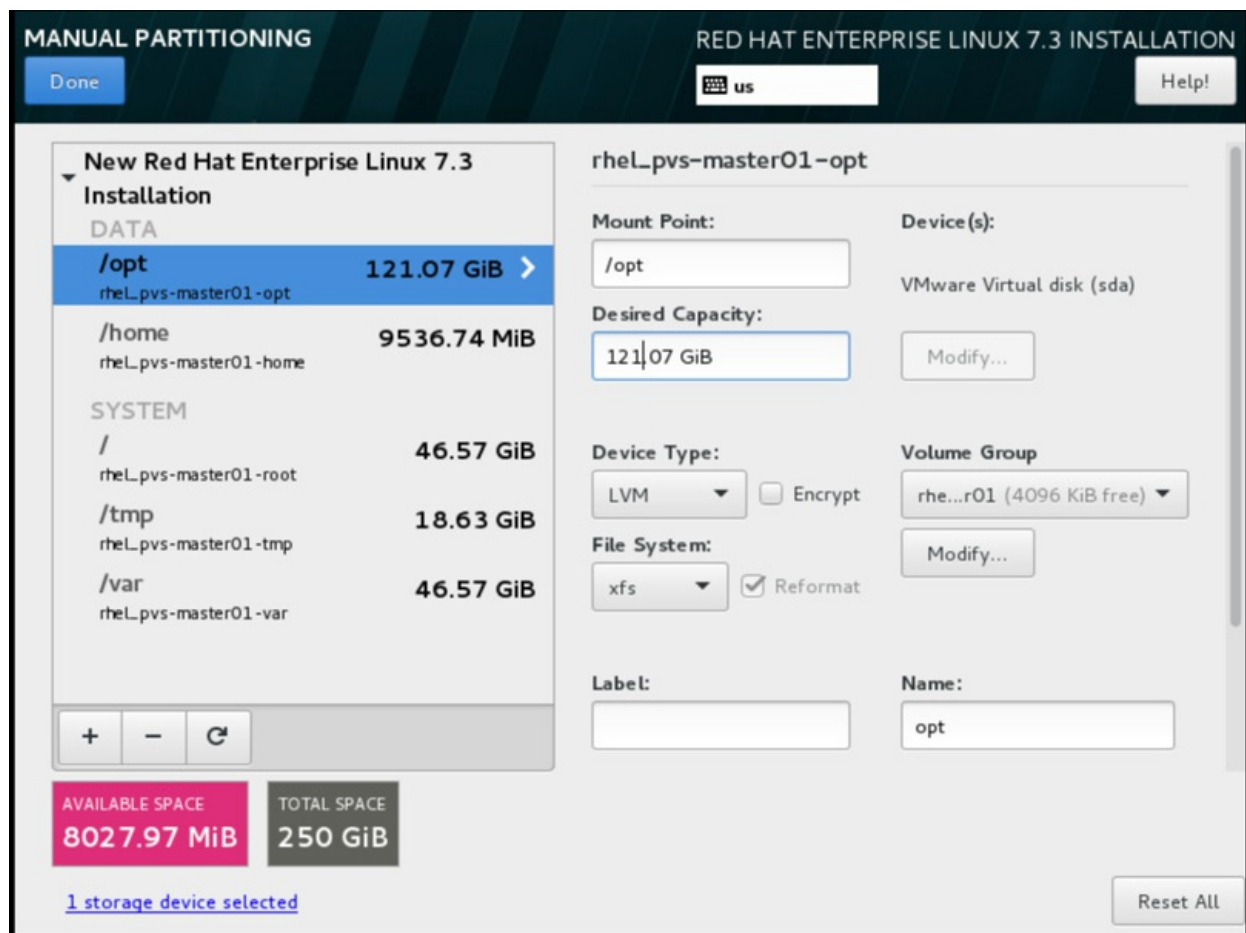
NOTE: The partitions are configured to use LVM except for the boot and swap partitions. The boot partition needs to be a "standard disk" partition. The recommended boot partition size is relatively small: 250 MB. It only needs to hold the boot loader, the kernel and some other low level system files.

TBD: The swap space is also a standard disk partition.

The following table is a summary of the sample disk partitions.

File System Name	Mount Point	Size (GB)
system (aka root)	/	40
boot	/boot	256 MB
swap		8
var	/var	50
tmp	/tmp	20
home	/home	20
opt	/opt	115

The figure below is a screen shot after the disk has been fully partitioned. (The partition sizes in the screen shot are slightly different from those in the table above.)



- Once you finish defining the disk partitions, click on the Done button in the upper left

corner of the screen. You will be returned to the Software Summary screen.

- At this point you can click on "Begin Installation" in the lower right corner of the Software Summary screen.
- In the next screen you need to provide the root password.
- Create a default user. In this sample, a user named "ICP Maestro" with user ID `icpmaestro` is created.

CREATE USER RED HAT ENTERPRISE LINUX 7.3 INSTALLER

[Done](#) us

Full name:

User name:

Tip: Keep your user name shorter than 32 characters and do not use spaces.

☐ Make this user administrator

☒ Require a password to use this account

Password:

Weak

Confirm password:

[Advanced...](#)

- Once the installation has completed, a reboot button will appear in the lower right corner of the final installation screen. The reboot completes the installation.
- Using the console, log in and get the IP address (`ip addr` or `ifconfig -a`) of the new VM and record it in your machine inventory.
- At this point you should be able to `ssh` to the machine either as root or as the default user you defined.