Zenoss Core Planning Guide

Release 5.1.2

Zenoss, Inc.

www.zenoss.com

Zenoss Core Planning Guide

Part Number: 1031.16.112

Zenoss, Inc.
11305 Four Points Drive
Bldg 1 - Suite 300
Austin, Texas 78726

Contents

About this guide

Zenoss Core Installation Guide provides detailed information about preparing to install Zenoss Core.

Related publications

Title	Description
Zenoss Core Administration Guide	Provides an overview of Zenoss Core architecture and features, as well as procedures and examples to help use the system.
Zenoss Core Configuration Guide	Provides required and optional configuration procedures for Zenoss Core, to prepare your deployment for monitoring in your environment.
Zenoss Core Installation Guide	Provides detailed information and procedures for creating deployments of Control Center and Zenoss Core.
Zenoss Core Planning Guide	Provides both general and specific information for preparing to deploy Zenoss Core.
Zenoss Core Release Notes	Describes known issues, fixed issues, and late-breaking information not already provided in the published documentation set.
Zenoss Core Upgrade Guide	Provides detailed information and procedures for upgrading deployments of Zenoss Core.

Additional information and comments

Zenoss welcomes your comments and suggestions regarding our documentation. To share your comments, please send an email to `docs@zenoss.com`. In the email, include the document title and part number. The part number appears at the end of the list of trademarks, at the front of this guide.

1 Introduction to Control Center

This chapter introduces Control Center, an open-source application service orchestrator based on *Docker*. Starting with release 5.0, Zenoss Core is an application managed by Control Center.

Introduction to Control Center

Control Center is a platform-as-a-service framework that can manage Zenoss Core and any other Docker application, from a simple web application to a multi-tiered stateful application stack. Control Center is based on a service-oriented architecture, which enables applications to run as a set of distributed services spanning hosts, datacenters, and geographic regions.

Control Center includes the following, key features:

- Intuitive HTML5 interface for deploying and managing Zenoss Core
- Integrated backup and restore, and incremental snapshot and rollback support
- Centralized logging, through Logstash and ElasticSearch
- Support for database services and other persistent services
- Encrypted communications among all services and containers

Docker fundamentals

Note This section is a summation of *the architecture description provided by Docker*, customized for Zenoss Core. For additional information, refer to the Docker site.

Docker provides convenient tools that make use of the *cgroups feature of the Linux kernel* to develop, distribute, and run applications. Docker internals include images, registries, and containers.

Docker images

Docker images are read-only templates that are used to create Docker containers. Images are easy to build, and image updates are change layers, not wholesale replacements.

Docker registries

Docker registries hold images. The Zenoss Core appliance includes a private Docker registry that holds the images of the Zenoss Core application.

Docker containers

Docker containers have everything needed to run an instance of an application, and are created from images. The Zenoss Core application includes many different containers, and each container is used to run one or more instances of a specific service.

Control Center terms and concepts

application

A collection of one or more software programs that have been converted into Docker containers. For example, Zenoss Core.

resource pool

A collection of one or more hosts, each with its own compute, network, and storage resources. The name of the default resource pool is `default`.

resource pool host

A host that runs the application services scheduled for the resource pool to which it belongs. A system may be configured as agent and master, or just agent, or just master.

master host

The host that runs the application services scheduler, the Docker registry, the distributed file system, and other internal services, including the server for the Control Center browser interface. A system may be configured as agent and master, or just agent, or just master. Only one system in a Control Center cluster may be the master.

cluster

The collection of hosts in one or more Control Center resource pools.

Docker and Control Center storage drivers

Starting with release 1.1.1, Control Center includes the `devicemapper` storage driver for application data. The Control Center driver is based on *the Docker `devicemapper` storage driver*, which in turn is based on the *device mapper framework of the Linux kernel*.

The key feature of the drivers is their use of thin provisioning, a virtualization method that allocates data blocks only when data is written. (The traditional method is to allocate data blocks when a file system is created, before any data is written.) Thin provisioning enables snapshots, a time-efficient and space-efficient method of copying data, and enables making a device appear to have more physical data blocks than are actually available (as long as some blocks are unfilled). Also, thin-provisioned storage can be extended without having to move data from one physical partition to another.

For Docker and for Control Center, Logical Volume Manager (LVM) tools are used to create thin-provisioned storage, in thin pools. A thin pool includes an area for metadata (a small percentage of the total) and an area for the data itself. To simplify creating thin pools, Control Center includes `serviced-storage`, a utility that calls the LVM tools. The utility may be used to create thin pools for use by Docker's `devicemapper` storage driver as well as for the Control Center `devicemapper` storage driver.

For Docker data storage, the recommended storage layout is a device mapper thin pool on one or more primary partitions. Docker's `devicemapper` storage driver may be used in `loop-lvm` mode on loopback-mounted sparse files. However, `loop-lvm` mode is not recommended for production use, and Zenoss strongly recommends using a device mapper thin pool for Docker storage, in all deployment scenarios.

For Control Center application data storage, the recommended (and default) storage driver is `devicemapper`, and the recommended layout is a device mapper thin pool on one or more primary partitions.

Hardware requirements 2

Control Center and Zenoss Core require real or virtual hosts that implement the 64-bit version of the x86 instruction set, and support Red Hat Enterprise Linux (RHEL) 7.1 or 7.2, or CentOS 7.1 or 7.2. Hardware resource requirements vary by role (master or resource pool host) and by the services assigned to the resosurce pool to which a host belongs.

Master host resource requirements

Control Center requires a real or virtual master host that implements the 64-bit version of the x86 instruction set, and supports RHEL/CentOS 7.1 or 7.2.

For single-host deployments, the Control Center master host runs all Control Center internal services and all Zenoss Core application services, and requires the following resources:

- 4 CPU cores (64-bit only; real or virtual)
- 20GB RAM
- 1 network interface controller (must support TCP/IP and IPv4)
- Local or remote high-performance storage (detailed separately)

Note An under-resourced master host does not function properly. Please do not deploy Control Center and Zenoss Core on a master host that does not meet the minimum requirements.

Master host storage requirements

The Control Center master host requires high-performance local or remote storage to function properly. For most deployments, solid-state disk (SSD) devices that are directly attached to the master host provide an excellent storage solution, and Zenoss recommends them.

Storage-area network (SAN) systems are also supported. The overall response times and configuration of a SAN affect the performance and stability of Control Center, and therefore, Zenoss Core. For example, ZooKeeper (a key internal service of Control Center) is sensitive to storage latency greater than 1000 milliseconds. Zenoss recommends using only high-performance SAN systems, and assigning separate logical unit numbers (LUNs) for each mounted path.

In addition to the storage required for its operating system, a Control Center master host requires the following storage areas:

Docker data storage

Size: 50GB (minimum)

Type: Device mapper thin pool

Mount point: None

The Docker data storage area contains the images and snapshots of the containers it manages.

To prepare for installation, simply create a primary partition. The thin pool is created during the installation process.

Control Center internal services data

Size: 50GB (minimum)

Type: XFS file system

Mount point: `/opt/serviced/var/isvcs`

The storage area for Control Center internal services data contains a variety of run-time data, including the ZooKeeper data. ZooKeeper requires consistently fast storage, so ideally, the primary partition for Control Center internal services is on a separate, high-performance device that has only one primary partition.

To prepare for installation, simply create a primary partition or logical volume. The XFS file system for a local partition is created during the installation process.

Application data

Size: 300GB (minimum)

Type: Device mapper thin pool

Mount point: None

The storage area for Control Center application data provides space for Zenoss Core databases.

Zenoss Core data storage requirements vary by collection rate. For example, collecting 25,000 metrics per second can require 1.5TB of storage in 90 days.

To prepare for installation, simply create a primary partition. The thin pool is created during the installation process.

Application data backups

Size: 150GB (minimum; at least 50% of the application data size)

Type: XFS file system, or any Linux-compatible file system

Mount point: `/opt/serviced/var/backups`

The storage area for Control Center backups can be a local partition or a remote file server. The Control Center browser interface stores backup files in this storage area, and uses a subdirectory for temporary files that are created during a restore. Backup files are very large (in the default configuration, it includes the metrics database) and the restore directory uses an equivalent amount.

This path does not require a dedicated partition or logical volume, but Zenoss recommends providing one. Otherwise, this path is part of the root partition or logical volume, and could overfill it, which would crash the master host.

To prepare for installation, simply create a primary partition or a logical volume, or reserve an area on a remote file server. The XFS file system for a local partition is created during the installation process. Likewise, the installation instructions include steps for mounting a remote file server.

Control Center metadata (high-availability systems only)

Size: 1GB

Type: XFS file system

Mount point: `/opt/serviced/var/volumes`

The storage area for Control Center metadata is required to mirror the data between the master nodes in high-availability deployments only.

To prepare for installation, simply create a primary partition. The XFS file system is created during the installation process.

Resource pool host requirements

Control Center requires real or virtual resource pool hosts that implement the 64-bit version of the x86 instruction set, and and support RHEL/CentOS 7.1 or 7.2.

- 4 CPU cores (64-bit only; real or virtual)
- 20GB RAM
- 1 network interface controller (must support TCP/IP and IPv4)
- Local or remote high-performance storage (detailed separately)

In all cases, the hosts in resource pools need enough RAM, CPU, and storage resources to support the services assigned to the pool. In all cases, the resources of all hosts in a resource pool should be identical.

Resource pool host storage requirements

Like master hosts, Control Center resource pool hosts require high-performance local or remote storage to function properly. For most deployments, solid-state disk (SSD) devices that are directly attached to the host provide an excellent storage solution, and Zenoss recommends them.

Storage-area network (SAN) systems are also supported. The overall response times and configuration of a SAN affect the performance and stability of Control Center, and therefore, Zenoss Core. Zenoss recommends using only high-performance SAN systems, and assigning separate logical unit numbers (LUNs) for each mounted path.

In addition to the storage required for its operating system, the typical Control Center resource pool host only requires storage for Docker data. The characteristics of, and preparation for, this storage area are identical to those of the master host.

For resource pool hosts supporting the recommended multi-host configuration, in which the master host is in its own, separate resource pool, two resource pool hosts require an additional storage area. The area is used for Control Center internal services data, so that the hosts can participate in a ZooKeeper ensemble. The characteristics of, and preparation for, this storage area are identical to those of the master host.

3 Operating system requirements

Control Center and Zenoss Core require the 64-bit version of the following Linux distributions.

- Red Hat Enterprise Linux (RHEL) 7.1 or 7.2
- CentOS 7.1 or 7.2

All versions of the Linux kernel included in these releases, and all subsequent updates, are supported. However, Zenoss encourages you to keep the kernel up-to-date.

Control Center and Zenoss Core are tested on operating system platforms that are installed and configured with standard options.

The RHEL/CentOS 7.x distributions provide a variety of server configurations. Docker and Control Center are tested and supported the Minimal Install configuration, when the NFS and NTP packages are installed as well.

Control Center relies on the system clock to synchronize its actions. The installation procedures include steps to add the Network Time Protocol (NTP) daemon to all hosts. By default, the NTP daemon synchronizes the system clock by communicating with standard time servers available on the internet. You may configure the daemon to use a timeserver in your environment, instead.

Note Because of the reliance on the system clock, pausing a virtual machine that belongs to a Control Center cluster is not supported.

Networking requirements

On startup, Docker creates the `docker0` virtual interface and selects an unused IP address and subnet (typically, 172.17.0.1/16) to assign to the interface. The virtual interface is used as a virtual Ethernet bridge, and automatically forwards packets among real and virtual interfaces attached to it. The host and all of its containers communicate among one another through this virtual bridge.

Docker can only check directly-connected routes, so the subnet it chooses for the virtual bridge may be inappropriate for your environment. To customize the virtual bridge subnet, refer to Docker's *advanced network configuration* article.

The default configurations of firewall utilities such as *FirewallD* include rules that can conflict with Docker, and therefore, Control Center and Zenoss Core. The following interactions illustrate the conflicts:

- The `firewalld` daemon removes the DOCKER chain from `iptables` when it starts or restarts.
- Under `systemd`, `firewalld` is started before Docker. However, if you start or restart `firewalld` while Docker is running, you need to restart Docker.

If you are using a firewall utlity, please ensure that it does not conflict with Docker.

If you are not using a firewall utlity, your firewall settings may still prevent communications over the Docker virtual bridge. This occurs when `iptables` INPUT rules restrict most traffic. To ensure that the bridge works properly, append an INPUT rule to your `iptables` configuration that allows traffic on the bridge subnet. For example, if `docker0` is bound to 172.17.42.1/16, then the following, non-specific command ensures that the bridge works.

```
iptables -A INPUT -d 172.17.0.0/16 -j ACCEPT
```

Note The preceding command is only an example. Please consult your networking specialist before modifying your `iptables` configuration.

Additional requirements

Control Center requires a 16-bit, private IPv4 network for virtual IP addresses. The default network is 10.3/16. If the default network is already in use in your environment, you may select any valid IPv4 16-bit address space during installation.

This release of Control Center relies on Network File System (NFS) for its distributed file system implementation. For this reason, hosts in a Control Center cluster may not run a general-purpose NFS server, and all hosts require NFS.

All hosts in Control Center resource pools must:

- be able to resolve the hostnames of all other resource pool hosts to IPv4 addresses (for example, if the public IP address of your host is 192.0.2.1, then the `hostname -i` command should return 192.0.2.1)
- respond with an IPv4 address other than `127.x.x.x` when `ping` *`Hostname`* is invoked
- return a unique result from the `hostid` command

Security considerations

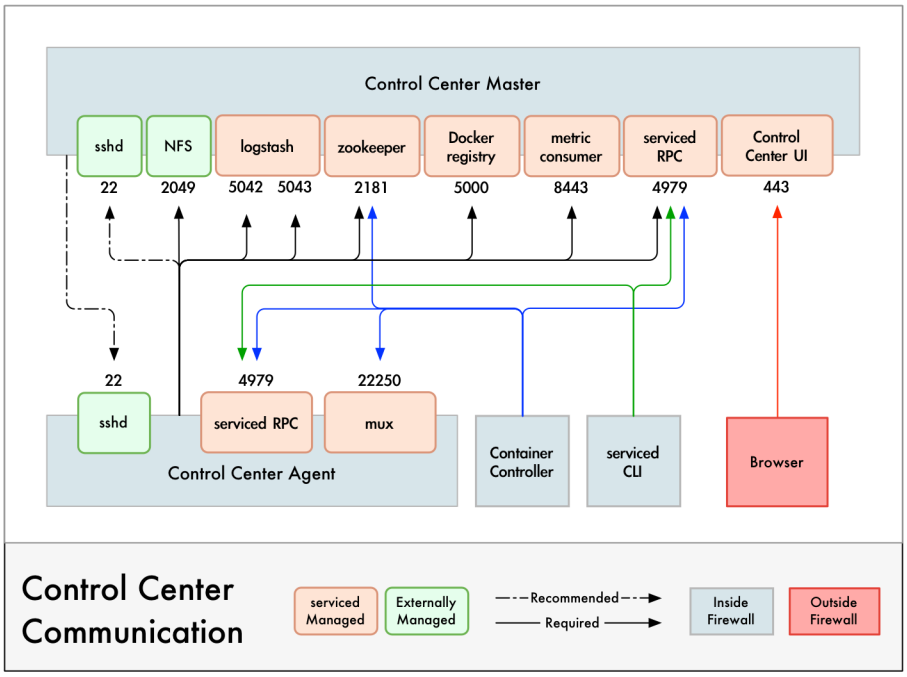
During installation, Control Center has no knowledge of the port requirements of the applications it is to manage, so the installation procedure includes disabling the firewall. After both Control Center and Zenoss Core are installed, you may close unused ports.

Control Center includes a virtual multiplexer (mux), to aggregate the UDP and TCP traffic among the services it manages. The aggregation is opaque to services, and mux traffic is encrypted when it travels among containers on remote hosts. (Traffic among containers on the same host is not encrypted.) The mux, along with the distributed file system, enables Control Center to deploy services to any pool host, rapidly. The mux also reduces the number of open ports required on a Control Center host to a predictable set.

The following illustration identifies the ports that Control Center requires for its operations. All of the ports except 4979 are configurable. All traffic is TCP.

Note Control Center relies on the system clock to synchronize its actions, and indirectly, NTP, to synchronize clocks among multiple hosts. In the default configuration of `ntpd`, the firewalls of master and resource pool hosts must support an incoming UDP connection on port 123.

Figure 1: Port requirements for Control Center hosts



Additional considerations

- To gain access to the Control Center browser interface, users must have login accounts on the Control Center master host. (Pluggable Authentication Modules (PAM) is supported.) By default, the users must be members of the `wheel` group.
- The `serviced` startup script sets the hard and soft open files limit to 1048576, but does not modify the `/etc/sysconfig/limits.conf` file.
- Control Center does not support *Security Enhanced Linux* in enforcing mode. The installation procedures include steps to set the mode to disabled.
- The `firewalld` service can conflict with Docker, and therefore, Control Center and Zenoss Core. For more information, see *Networking requirements* on page 10.

4 Packaging and interface access

This chapter describes how Control Center and Zenoss Core are packaged and distribued, and specifies the supported browsers for the Control Center and Zenoss Core browser interfaces.

Packaging considerations

Control Center is designed to support any application that includes one or more services that are built into Docker containers. A service definition template contains the specifications of application services, in JSON format. The definition of each service includes the IDs of the Docker images needed to run the service.

Control Center, and the service definition templates for Zenoss Core, are distributed as Redhat (`yum/rpm`) packages. The packages are available at public repositories maintained by Zenoss. The Docker images that Zenoss Core requires are available at the Zenoss *Docker Hub* repository. So, the default installation process requires internet access. However, all of the repositories may be mirrored, so offline installations are supported as well.

Note The Docker images for Zenoss Core are available in a public Zenoss repository at Docker Hub. No special permissions are required to pull the images.

The Docker, Control Center, and Zenoss Core packages require approximately 5GB of storage space.

Supported clients and browsers

The client operating systems and web browser combinations supported in this release.

- All browsers must have Adobe® Flash® Player 11 installed, or a more recent version.
- Compatibility mode is not supported in Internet Explorer.

Client OS	Supported Browsers
Windows 7 and 8.1	Internet Explorer 11 (enterprise mode is supported)
	Internet Explorer 10
	Firefox 30 and above
	Chrome 30 and above
Windows Server 2012 R2	Firefox 30
	Chrome 36
Macintosh OS/X 10.9	Firefox 30 and above

Client OS	Supported Browsers
	Chrome 36 and above
Ubuntu 14.04 LTS	Firefox 30 and above
	Chrome 37 and above
Red Hat Enterprise Linux 6.5, CentOS 6.5	Firefox 30 and above
	Chrome 37 and above

Storage management on Linux hosts A

This appendix includes basic procedures for managing storage on a Linux host.

Identifying storage devices and their configuration

Perform this procedure to identify the block storage devices attached to a host, and learn how the devices are configured.

1 Log in to the target host as `root`, or as a user with superuser privileges through a terminal session.
2 Display the block storage devices attached to the host and their configuration.

```
lsblk --output=NAME,SIZE,TYPE,FSTYPE,MOUNTPOINT
```

Example result:

```
NAME                      SIZE TYPE FSTYPE      MOUNTPOINT
sda                       128G disk
|-sda1                    500M part xfs         /boot
|-sda2                  127.5G part LVM2_member
  |-centos_c15246-swap   24.8G lvm  swap
  |-centos_c15246-root     50G lvm  xfs         /
  |-centos_c15246-home   52.6G lvm  xfs         /home
sdb                       768G disk
sdc                       768G disk
sr0                      1024M rom
```

The preceding result shows three disks (`sda`, `sdb`, and `sdc`) and one CD/DVD drive (`sr0`). There are no primary partitions on `sdb` or `sdc`. There are two primary partitions on `sda`: `sda1` and `sda2`.

- `sda1` is devoted to `/boot`, and formatted with the *XFS* file system.
- Partition `sda2` includes the following logical volumes, which are managed by LVM:
  - a swap volume, formatted as such
  - a volume for root (`/`), formatted as XFS
  - a volume for `/home`, formatted as XFS

For more information about `lsblk`, enter `man lsblk`.

Creating primary partitions

To perform this procedure, you need:

- The password of the `root` user account on a Linux host, or of a user account that belongs to the `wheel` group.
- A Linux host with at least one local or remote disk.

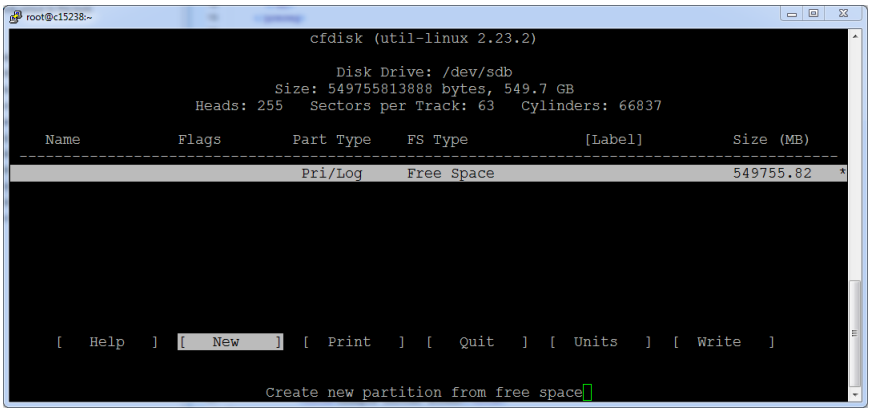
This procedure demonstrates how to create primary partitions on a disk. Each primary partition may be formatted as a file system or swap space, used in a device mapper thin pool, or reserved for future use. Each disk must have one primary partition, and may have four. If you are uncertain whether a disk is partitioned, see the preceding topic.

Note Data present on the disk you select is destroyed by this procedure. Please ensure that data present on the disk is backed up elsewhere, or no longer needed, before proceeding.

1. Log in to the target host as `root`, or as a user with superuser privileges.
2. Start the partition table editor for the target disk.
   In this example, the taget disk is `/dev/sdb`, and it has no entries in its partition table.

```
cfdisk /dev/sdb
```

Figure 2: Initial screen



The `cfdisk` command provides a text user interface (TUI) for editing the partition table. The following list describes how to navigate through the interface:

- To select an entry in the table, use the up and down arrow keys. The current entry is highlighted.
- To select a command from the menu at the bottom of the interface, use the left and right arrow keys, or **Tab** and **Shift-Tab**. The current command is highlighted.
- To choose a command, press the **Enter** key.
- To return to the previous level of the menu, press the **Esc** key.
- To exit the interface, select **Quit** from the menu, and then press the **Enter** key.

For more information about `cfdisk`, enter `man cfdisk`.

3. Create a new partition.

   Repeat the following substeps for each primary partition to create. You may create four primary partitions on a disk.

   a Select the table entry with the value **Free Space** in the **FS Type** column.
   b Select **[New]**, and then press the **Enter** key.
   c Select **[Primary]**, and then press the **Enter** key.

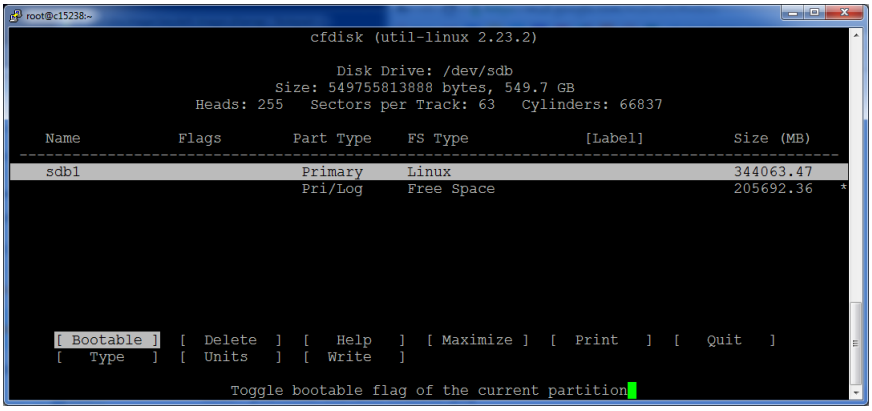
d At the **Size (in MB)** prompt, enter the size of the partition to create in megabytes, and then press the **Enter** key.

To accept the default value, which is all of the free space on the disk, just press the **Enter** key.

e Note If you created a single partition that uses all of the available disk space, skip this substep.

Optional: Select **[Beginning]**, and then press the **Enter** key.

Figure 3: One primary partition



4 Write the partition table to disk, and then exit the partition table editor.

a Select **[Write]**, and then press the **Enter** key.

b At the **Are you sure...** prompt, enter `yes`, and then press the **Enter** key.

You can ignore the warning about a bootable partition.

c Select **[Quit]**, and then press the **Enter** key.

Creating a swap partition

To perform this procedure, you need:

- A host with one or more local disks, with at least one unused primary partition.
- The password of the `root` account on the host, or of a user that is a member of the `sudo` or `wheel` group.

Perform this procedure to configure a primary partition on a local disk as swap space. Typically, configuring one swap partition or swap file on each local disk maximizes swap space performance.

Note This procedure does not use Logical Volume Manager (LVM) tools to create a swap space. For more information about LVM, refer to your operating system documentation.

1 Log in to the target host as `root`, or as a user with superuser privileges.

2 Identify one or more primary partitions for use as swap space.

```
lsblk -p --output=NAME,SIZE,TYPE,FSTYPE,MOUNTPOINT
```

3 Create and enable swap space on each target primary partition.

a Disable swapping on all swap devices.

```
swapoff -a
```

b Create swap space.

Repeat the following command for each primary partition to use as swap space.

Replace *Device* with the path of a primary partition:

```
mkswap Device
```

c Update the file system table.

Repeat the following command for each swap partition created in the previous substep.

Replace *Device* with the path of a swap partition:

```
echo "Device  swap  swap  defaults  0  0" >> /etc/fstab
```

d Enable swapping on all swap devices.

```
swapon -a
```