Zenoss Community Edition (Core) Planning Guide

Release 6.2.1

Zenoss, Inc.

www.zenoss.com

Zenoss Community Edition (Core) Planning Guide

Part Number: 1631.18.248.40

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

Contents

About this guide

Zenoss Community Edition (Core) Planning Guide provides information about planning a Zenoss Community Edition (Core) (short name: Zenoss Core) deployment. You can create a deployment by installing virtual appliances on a hypervisor host or by adding Zenoss Core to a Control Center deployment.

This guide includes information about planning a Control Center deployment, which has host requirements that are independent of Zenoss Core requirements.

Zenoss Core publications

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

Control Center publications

Title	Description
Control Center Release Notes	Describes known issues, fixed issues, and late-breaking information not included in other publications.
Control Center Installation Guide	Provides detailed procedures for installing and configuring Control Center.
Control Center Installation Guide for High-Availability Deployments	Provides detailed procedures for installing and configuring Control Center in a high-availability deployment.
Control Center Reference Guide	Provides information and procedures for managing Control Center. This information is also available as online help in the Control Center browser interface.
Control Center Upgrade Guide	Provides detailed procedures for updating a Control Center deployment to the latest release.
Control Center Upgrade Guide for High-Availability Deployments	Provides detailed procedures for updating a high-availability deployment of Control Center to the latest release.

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 (*Zenoss Community Edition (Core) Planning Guide*) and part number (1631.18.248.40).

Change history

The following list associates document part numbers and the important changes to this guide since the previous release. Some of the changes involve features or content, but others do not. For information about new or changed features, refer to the *Zenoss Community Edition (Core) Release Notes*.

1631.18.248.40 (6.2.1)

Update release numbers.

1631.18.162.37 (6.2.0)

Add the contents of the former *Control Center Planning Guide* to this guide.

Update release numbers.

1631.18.081.40 (6.1.2)

Update release numbers.

1631.18.039.21 (6.1.1)

Update release numbers.

1631.18.009 (6.1.0)

Update release numbers.

1631.18.009 (6.1.0)

Update release numbers.

1631.17.320 (6.0.1)

Controlled availability release.

1631.17.311 (6.0.0)

Controlled availability release.

1031.17.313 (5.3.3)

Update release numbers.

1031.17.268 (5.3.2)

Update release numbers.

1031.16.242 (5.3.1)

Update release numbers.

1031.17.229 (5.3.0)

Update release numbers.

1031.17.191

Update release numbers.

About 5.2.5

Version 5.2.5 was withdrawn.

1031.17.122 (5.2.4)

Update release numbers.

1031.17.100 (5.2.3)

Update release numbers.

1031.17.058 (5.2.2)

Update release numbers.

1031.17.044 (5.2.1)

Update release numbers.

1031.16.335 (5.2.0)

Add this history section to this guide.

Add a tested environments section to this guide.

Remove Control Center planning and storage management information. The information is now in the *Control Center Planning Guide*.

1 Welcome to Zenoss!

This chapter provides an overview of Zenoss software, including introductions to Control Center and Zenoss Core, and a description of the installation options.

Introduction to Control Center

Control Center is an open-source application service orchestrator based on *Docker Community Edition* (Docker CE, or just Docker).

Control Center can manage any 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 relies on declarations of application requirements to integrate 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 includes the following key features:

- Intuitive HTML5 interface for deploying and managing applications
- Integrated backup and restore, and incremental snapshots and rollbacks
- Centralized logging through Logstash and Elasticsearch
- Integration with database services and other persistent services
- Encrypted communications among all services and containers
- Delegate host authentication to prevent unauthorized system access
- Storage monitoring and emergency shutdown of services to minimize the risk of data corruption
- Rolling restart of services to reduce downtime of multi-instance services
- Audit logging, including application audit logging

Terminology, internal services, and concepts

This section defines Control Center terminology, internal services that enable Control Center to function, and concepts that are used in this guide and other documentation.

application

One or more software services packaged in Docker containers.

delegate host

A host that runs the application services scheduled for the resource pool to which it belongs. A system can be configured as delegate or master.

Docker Community Edition (Docker CE, or just Docker)

An open-source application for building, shipping, and running distributed applications.

Elasticsearch

(Control Center internal service) A distributed, real-time search and analytics engine. Control Center uses it to index log files and store service definitions.

Kibana

(Control Center internal service) A browser-based user interface that enables the display and search of Elasticsearch databases, including the log files that Control Center monitors.

Logstash

(Control Center internal service) A log file collector and aggregator that forwards parsed log file entries to Elasticsearch.

master host

The host that runs the application services scheduler, the Docker registry, the distributed file system, and other internal services, including the HTTP server for the Control Center browser interface and application browser interface. A system can be configured as delegate or master. Only one Control Center host can be the master.

OpenTSDB

(Control Center internal service) A time series database that Control Center uses to store its service performance metrics.

resource pool

A collection of one or more hosts, each with its own compute, network, and storage resources. All of the hosts in a resource pool must have identical hardware resources, and must be located in the same data center and on the same subnet. If a resource pool host is a hypervisor guest system, all of the hosts in the resource pool must be guests of the same hypervisor host system.

service

A process and its supporting files that Control Center runs in a single container to provide specific functionality as part of an application.

serviced

The name of the Control Center service and a command-line client for interacting with the service.

tenant

An application that Control Center manages.

ZooKeeper (*Apache ZooKeeper*)

(Control Center internal service) A centralized service that Control Center uses for configuration maintenance, naming, distributed synchronization, and providing group services.

Docker fundamentals

This section summarizes *the architecture description provided by Docker* as customized for Control Center. For additional information, refer to the Docker site.

Docker provides convenient tools that make use of the *control groups 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. Control Center uses a private Docker registry for its own images and Zenoss application images.

Docker containers

Docker containers have everything needed to run a service instance, and are created from images. Control Center launches each service instance in its own Docker container.

Docker storage

Docker and Control Center data are stored in customized LVM thin pools that are created from one or more block devices or partitions, or from one or more LVM volume groups.

ZooKeeper and Control Center

Control Center relies on *Apache ZooKeeper* to distribute and manage application services. ZooKeeper maintains the definitions of each service and the list of services assigned to each host. The scheduler, which runs on the master host, determines assignments and sends them to the ZooKeeper node that is serving as the ensemble leader. The leader replicates the assignments to the other ensemble nodes, so that the other nodes can assume the role of leader if the leader node fails.

All Control Center hosts retrieve assignments and service definitions from the ZooKeeper ensemble leader and then start services in Docker containers as required. So, the Control Center configuration files of all Control Center hosts must include a definition for the *SERVICED_ZK* variable, which specifies the ZooKeeper endpoints of the ensemble nodes. Additional variables are required on ensemble nodes.

A ZooKeeper ensemble requires a minimum of three nodes, which is sufficient for most environments. An odd number of nodes is recommended and an even number of nodes is strongly discouraged. A five-node ensemble improves failover protection during maintenance windows but larger ensembles yield no benefits.

The Control Center master host is always an ensemble node. All ensemble nodes should be on the same subnet.

Control Center application data storage

Control Center uses a dedicated LVM thin pool on the master host to store application data and snapshots of application data.

- The distributed file system (DFS) of each tenant application that `serviced` manages is stored in separate virtual devices. The initial size of each tenant device is copied from the base device, which is created during the initial startup of `serviced`.
- Snapshots of tenant data, used as temporary restore points, are stored in separate virtual devices, outside of tenant virtual devices. The size of a snapshot depends on the size of the tenant device, and grows over time.

The Control Center master host requires high-performance, persistent storage. Storage can be local or remote.

- For local storage, solid-state disk (SSD) devices are recommended.
- For remote storage, storage-area network (SAN) systems have been tested. High-performance SAN systems are recommended, as is assigning separate logical unit numbers (LUNs) for each mounted path.

The overall response times of master host storage affect the performance and stability of Control Center internal services and the applications it manages. For example, ZooKeeper (a key internal service) is sensitive to storage latency greater than 1000 milliseconds.

Note The physical devices associated with the application data thin pool must be persistent. If removable or re-connectable storage such as a SAN based on iSCSI is used, then the Device-Mapper Multipath feature of RHEL/CentOS must be configured and enabled.

Control Center includes the `serviced-storage` utility for creating and managing its thin pool. The `serviced-storage` utility can:

- use physical devices or partitions, or LVM volume groups, to create a new LVM thin pool for application data
- add space to a tenant device at any time
- identify and clean up orphaned snapshots
- create an LVM thin pool for Docker data

Introduction to Zenoss Core

Starting with release 5.0, Zenoss Core is deployed into a distributed architecture using Control Center. That is, Zenoss Core is an application whose installation and deployment configuration is managed by Control Center.

Control Center and Zenoss Core are independent and unaware of each other, although Control Center is designed for the unique requirements of Zenoss Core. In particular, Zenoss Core and Control Center have a different version numbering scheme and different release schedules. Some features of Zenoss Core rely on specific capabilities of Control Center, so this guide includes a table of the tested combinations, *Tested operating environments* on page 10. A separate series of guides describes how to use Control Center.

Note In the Zenoss Core context, a resource is a device to monitor. Devices include compute, storage, network, converged infrastructure, applications, and unified communications systems. In the Control Center context, a resource is a physical or virtual host in a Control Center deployment.

Key Zenoss Core concepts

master pool

A Control Center resource pool that contains only the Control Center master host. By isolating the master host in its own pool and configuring a ZooKeeper ensemble, Zenoss Core services are not affected when Control Center internal services are temporarily unavailable. The recommended name for this pool is `master`.

Zenoss Core pool

A Control Center resource pool that runs Zenoss Core infrastructure services and other key services. This resource pool must be on the same subnet as the master pool. A Zenoss Core pool includes a minimum of two Control Center hosts. The recommended name is `Zenoss Core`.

Tested operating environments

Zenoss Core, Control Center, and operating systems

The following table identifies the tested combinations of Zenoss Core, Control Center, and operating system releases.

Zenoss Core release	Control Center	
	Minimum release	Host OS
6.2.1	1.5.1	RHEL/CentOS 7.2, 7.3, 7.4, or 7.5 (64-bit)
6.0.1, 6.1.0, 6.1.1, 6.1.2, 6.2.0**	1.5.0, 1.5.1	RHEL/CentOS 7.2, 7.3, or 7.4 (64-bit)

** Version 6.0.0 - controlled availability

Zenoss Core release	Control Center	
	Minimum release	Host OS
5.3.0, 5.3.1, 5.3.2, 5.3.3	1.3.0, 1.3.1, 1.3.2, 1.3.3, 1.3.4, 1.4.0, 1.4.1	RHEL/CentOS 7.1, 7.2, or 7.3 (64-bit)
5.2.0, 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.6*	1.2.0, 1.2.1, 1.2.2, 1.2.3, 1.3.0, 1.3.1, 1.3.2, 1.3.3, 1.3.4, 1.4.0, 1.4.1	RHEL/CentOS 7.1, 7.2, or 7.3 (64-bit)
5.1.9, 5.1.10	1.1.9, 1.2.0	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.8	1.1.5, 1.1.6, 1.1.7. 1.1.8, 1.1.9	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.7	1.1.5, 1.1.6, 1.1.7, 1.1.8	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.6 (internal release only)	(none)	(none)
5.1.4, 5.1.5	1.1.5, 1.1.6, 1.1.7	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.3	1.1.2, 1.1.3, 1.1.5	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.2	1.1.2, 1.1.3	RHEL/CentOS 7.1 or 7.2 (64-bit)
5.1.1	1.1.1, 1.1.2	RHEL/CentOS 7.1 or 7.2 (64-bit)

Supported clients and browsers

The following table identifies the supported combinations of client operating systems and web browsers.

Client OS	Supported browsers
Windows 7, 10	Internet Explorer 11*
	Firefox 56 and later
	Chrome 61 and later
macOS 10.12.3, 10.13	Firefox 56 and later
	Chrome 61 and later
Ubuntu 14.04 LTS	Firefox 56 and later
	Chrome 61 and later

Installation options

You can create an on-premise deployment of Zenoss Core by following one of two mutually-exclusive installation paths:

- Install Zenoss Core virtual appliances as guest systems on a VMWare vSphere or Microsoft Hyper-V hypervisor. The guest systems include Control Center and require relatively little configuration. This is the recommended deployment option.
- Install Control Center on Red Hat Linux or CentOS hosts and add Zenoss Core. This the most flexible installation path and it includes support for high-availability configurations.

* Version 5.2.5 - withdrawn
* Enterprise mode only; compatibility mode is not tested.

The remainder of this guide describes the resource requirements of the preceding installation paths.

2 Zenoss Core virtual appliances

Zenoss Core is available as a pair of virtual appliances, one for a Control Center master host, and another for Control Center delegate hosts. Control Center and Zenoss Core are installed, the required Docker images are loaded into the local registry, and the latest supported version of CentOS is installed with all of the packages needed to begin using Zenoss Core as quickly as possible.

Virtual appliances are packaged as Open Virtual Appliance (OVA) and ISO disk image files, and their resource requirements vary by hypervisor (VMware vSphere or Microsoft Hyper-V) and role (Control Center master or delegate).

Resource requirements for multi-host deployments

Zenoss strongly recommends a minimum of 3 hosts for all production deployments of Zenoss Service Dynamics. Development or testing deployments can be single-host deployments.

CPU and memory requirements

The following table provides CPU and memory requirements for hosts in the types of Control Center resource pools that a typical Zenoss Core deployment uses. For more information about pool types, see *Key Zenoss Core concepts* on page 10.

Resource pool type	Host requirements		
	Count	Resources	Comments
Master pool	1	4 CPU cores 16GB main memory	The amount of storage required for application data varies greatly.
Zenoss Core pool	2+n	4 CPU cores 16GB main memory	More than two hosts is preferred, to support failover.

Control Center master host storage

The Control Center master host virtual machine requires a total of 7 virtual hard disks. The following table identifies the purpose and size of each disk.

Purpose		Size
1	Root (/)	30GB

Purpose		Size
2	Swap	16GB
3	Temporary (`/tmp`)	16GB
4	Docker data	50GB
5	Control Center internal services data	50GB
6	Application data	200GB
7	Application data backups	150GB

On vSphere hypervisors, the disks are created when the OVA is installed. On Hyper-V hypervisors, the disks must be created manually.

Control Center delegate host storage

Control Center delegate host virtual machines requires a total of 4 virtual hard disks. The following table identifies the purpose and size of each disk.

Purpose		Size
1	Root (`/`)	30GB
2	Swap	16GB
3	Temporary (`/tmp`)	16GB
4	Docker data	50GB

On vSphere hypervisors, the disks are created when the OVA is installed. On Hyper-V hypervisors, the disks must be created manually.

Resource requirements for single-host deployments

A single-host deployment of the Zenoss Core virtual appliance requires a virtual machine with a minimum of 4 CPU cores and 24GB of main memory.

The disk requirements are identical to the master host disk requirements for multi-host deployments. On vSphere hypervisors, the disks are created when the OVA is installed. On Hyper-V hypervisors, the disks must be created manually.

Control Center resource requirements 3

This chapter describes the CPU, RAM, storage, and operating system requirements for creating a Control Center deployment for Zenoss Core. If you are planning a high-availability configuration, review to the *Control Center Installation Guide for High-Availability Deployments* for information about additional requirements.

Compute and storage requirements

Control Center requires real or virtual master and delegate hosts that

- implement the 64-bit version of the x86 instruction set
- support Red Hat Enterprise Linux (RHEL) 7.x or CentOS 7.x
- support Advanced Encryption Standard (AES)
- include a network interface controller that supports TCP/IP and IPv4

Hardware resource and storage requirements for Control Center vary by role (master or delegate host) and by the services assigned to the resource pool to which a host belongs. This chapter provides minimum requirements for master and delegate hosts.

Master host CPU and RAM resources

You can create a multi-host or single-host deployment of Control Center, on real or virtual hosts.

Zenoss recommends that all production deployments include one Control Center master host and two or more delegate hosts. In this configuration, the master host runs in its own, separate resource pool, and runs only Control Center internal services. Delegate hosts run application services in other resource pools.

- For multi-host deployments, the master host typically requires 4 real or virtual CPU cores and 16GB RAM. Very large multi-host deployments may require 8 CPU cores, but no additional RAM.
- For single-host deployments, the master host requires a minimum of 4 real or virtual CPU cores and 24GB RAM. Actual CPU and RAM requirements depend on application load.

Note An under-resourced master host cannot function properly. Deploy Control Center and Zenoss Core on a master host that meets or exceeds the minimum requirements.

Master host storage areas

A Control Center master host requires the following storage areas:

Root

Size: Depends on configuration

Type: XFS file system (typically)

Mount point: `/`

Resource: One or more block devices or partitions, or one or more LVM physical volumes.

Preparation: None. The file system is created when the operating system is installed.

Docker temporary

Size: 10GB

Type: XFS file system (typically)

Mount point: `/tmp`

Resource: One or more block devices or partitions, or one or more LVM physical volumes.

Preparation: Depends on configuration. Typically, additional space is allocated for the root filesystem, but a separate real or logical partition may be used. The *Control Center Installation Guide* includes instructions to link the Docker temporary directory to `/tmp`.

Swap

Size: 12GB to 16GB

Type: swap

Mount point: None

Resource: One or more block devices or partitions, one or more LVM physical volumes, or a special file on the root filesystem.

Preparation: Depends on configuration.

Docker data

Size: 50GB

Type: LVM thin pool

Mount point: None

Resource: One or more block devices or partitions, or one or more LVM physical volumes.

Preparation: None. The installation procedures include steps for creating the thin pool.

Control Center internal services data

Size: 50GB

Type: XFS file system

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

Resource: One or more block devices or partitions, or one or more LVM physical volumes.

Preparation: None. The installation procedures include steps for formatting the resource.

Control Center audit logging

Size: 10GB (default)

Type: XFS file system

Mount point: `/var/log/serviced`

Resource: One or more block devices or partitions, one or more LVM physical volumes, or a remote file server that is compatible with Linux.

Preparation: Depends on configuration. Typically, additional space is allocated for the root filesystem. For more information about audit logging, refer to the *Control Center Reference Guide*.

Application data

Size: 200GB suggested. The size should be twice as large as the base device, which determines the size of tenant virtual devices.

Type: LVM thin pool

Mount point: None

Resource: One or more block devices or partitions, or one or more LVM physical volumes.

Preparation: None. The installation procedures include steps for creating the thin pool.

Application data backups

Size: 150GB suggested. The size should be at least 150% of the size of the base device.

Note For large environments, this size should be much greater. Individual backup files can be 50GB to 100GB each, or more.

Type: XFS file system, or a Linux-compatible file server

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

Resource: One or more block devices or partitions, one or more LVM physical volumes, or a remote file server that is compatible with Linux.

Note When the storage for backups and Control Center internal services data use the same physical device, resource contention can cause failures during backup and restore operations. For optimal results, use separate physical devices.

Preparation: None. The installation procedures include steps for formatting or mounting the resource.

The following example shows the disk configuration of a Control Center master host.

Figure 1: Example master host with 7 disks

```
# lsblk -ap --output=NAME,SIZE,TYPE,FSTYPE,MOUNTPOINT
NAME                                                  SIZE TYPE FSTYPE      MOUNTPOINT
/dev/fd0                                                4K disk
/dev/sda                                             29.3G disk
##/dev/sda1                                           29.3G part xfs         /
/dev/sdb                                             48.8G disk
##/dev/sdb1                                           48.8G part LVM2_member
  ##/dev/mapper/docker-docker--pool_tmeta              900M lvm
  # ##/dev/mapper/docker-docker--pool                 42.2G lvm
  #   ##/dev/mapper/docker-8:1-42048-7a049d85cad6c      45G dm   xfs         /var/lib/docker/devicemapper/mnt/7a049d85cad6c
  #   ##/dev/mapper/docker-8:1-42048-0f93eb8de36bb      45G dm   xfs         /var/lib/docker/devicemapper/mnt/0f93eb8de36bb
  #   ##/dev/mapper/docker-8:1-42048-d84939118b6de      45G dm   xfs         /var/lib/docker/devicemapper/mnt/d84939118b6de
  #   ##/dev/mapper/docker-8:1-42048-2daa75d92947e      45G dm   xfs         /var/lib/docker/devicemapper/mnt/2daa75d92947e
  #   ##/dev/mapper/docker-8:1-42048-e7277d3081955      45G dm   xfs         /var/lib/docker/devicemapper/mnt/e7277d3081955
  #   ##/dev/mapper/docker-8:1-42048-e91db5ccdcf21      45G dm   xfs         /var/lib/docker/devicemapper/mnt/e91db5ccdcf21
  #   ##/dev/mapper/docker-8:1-42048-99104ce4a2d39      45G dm   xfs         /var/lib/docker/devicemapper/mnt/99104ce4a2d39
  #   ##/dev/mapper/docker-8:1-42048-a29f148212846      45G dm   xfs         /var/lib/docker/devicemapper/mnt/a29f148212846
  ##/dev/mapper/docker-docker--pool_tdata             42.2G lvm
    ##/dev/mapper/docker-docker--pool                 42.2G lvm
      ##/dev/mapper/docker-8:1-42048-7a049d85cad6c      45G dm   xfs         /var/lib/docker/devicemapper/mnt/7a049d85cad6c
      ##/dev/mapper/docker-8:1-42048-0f93eb8de36bb      45G dm   xfs         /var/lib/docker/devicemapper/mnt/0f93eb8de36bb
      ##/dev/mapper/docker-8:1-42048-d84939118b6de      45G dm   xfs         /var/lib/docker/devicemapper/mnt/d84939118b6de
      ##/dev/mapper/docker-8:1-42048-2daa75d92947e      45G dm   xfs         /var/lib/docker/devicemapper/mnt/2daa75d92947e
      ##/dev/mapper/docker-8:1-42048-e7277d3081955      45G dm   xfs         /var/lib/docker/devicemapper/mnt/e7277d3081955
      ##/dev/mapper/docker-8:1-42048-e91db5ccdcf21      45G dm   xfs         /var/lib/docker/devicemapper/mnt/e91db5ccdcf21
      ##/dev/mapper/docker-8:1-42048-99104ce4a2d39      45G dm   xfs         /var/lib/docker/devicemapper/mnt/99104ce4a2d39
      ##/dev/mapper/docker-8:1-42048-a29f148212846      45G dm   xfs         /var/lib/docker/devicemapper/mnt/a29f148212846
/dev/sdc                                             15.6G disk
##/dev/sdc1                                          15.6G part swap        [SWAP]
/dev/sdd                                             15.6G disk
##/dev/sdd1                                          15.6G part xfs         /tmp
/dev/sde                                             48.8G disk
##/dev/sde1                                          48.8G part xfs         /opt/serviced/var/isvcs
/dev/sdf                                            146.5G disk
##/dev/sdf1                                         146.5G part xfs         /opt/serviced/var/backups
/dev/sdg                                            195.3G disk
##/dev/sdg1                                         195.3G part LVM2_member
  ##/dev/mapper/serviced-serviced--pool_tmeta          1.8G lvm
  # ##/dev/mapper/serviced-serviced--pool            172.3G lvm
  #   ##/dev/mapper/docker-8:1-33639769-33l4GP         90G dm   ext4        /exports/serviced_volumes_v2/a9hil5o55iy266s5l
  ##/dev/mapper/serviced-serviced--pool_tdata        172.3G lvm
    ##/dev/mapper/serviced-serviced--pool            172.3G lvm
      ##/dev/mapper/docker-8:1-33639769-33l4GP         90G dm   ext4        /exports/serviced_volumes_v2/a9hil5o55iy266s5l
/dev/sr0                                             1024M rom
```

Delegate host CPU and RAM resources

The following table identifies resource requirements for delegate hosts.

Resource pool type	Host requirements		
	Count	Resources	Comments
Zenoss Core pool	2+n	4 CPU cores 16GB main memory	To support failover or additional services, more than two hosts is preferred.

Delegate host storage requirements

Like master hosts, Control Center delegate hosts require high-performance, persistent storage. Storage can be local or remote.

- For local storage, solid-state disk (SSD) devices are recommended.
- For remote storage, storage-area network (SAN) systems have been tested. High-performance SAN systems are recommended, as is assigning separate logical unit numbers (LUNs) for each mounted path.

The following storage configuration is recommended for delegate hosts:

- root filesystem with a minimum of 30GB of storage, formatted with XFS
- dedicated swap area
- one or more block devices or partitions, or one or more LVM physical volumes, with a total of 50GB of space. The installation procedures include steps for using `serviced-storage` to create an LVM thin pool for Docker data.

The following example shows the disk configuration of a Control Center delegate host.

Figure 2: Example delegate host with 4 disks

```
# lsblk -ap --output=NAME,SIZE,TYPE,FSTYPE,MOUNTPOINT
NAME                                                  SIZE TYPE FSTYPE      MOUNTPOINT
/dev/fd0                                                4K disk
/dev/sda                                             29.3G disk
##/dev/sda1                                           29.3G part xfs         /
/dev/sdb                                             48.8G disk
##/dev/sdb1                                           48.8G part LVM2_member
  ##/dev/mapper/docker-docker--pool_tmeta              900M lvm
  # ##/dev/mapper/docker-docker--pool                 42.2G lvm
  #   ##/dev/mapper/docker-8:1-16786822-2a6c2771415     45G dm   xfs         /var/lib/docker/devicemapper/mnt/2a6c2771415
  #   ##/dev/mapper/docker-8:1-16786822-579a2fb3611     45G dm   xfs         /var/lib/docker/devicemapper/mnt/579a2fb3611
  #   ##/dev/mapper/docker-8:1-16786822-c95ca5f04a4     45G dm   xfs         /var/lib/docker/devicemapper/mnt/c95ca5f04a4
  #   ##/dev/mapper/docker-8:1-16786822-e9733ecdda1     45G dm   xfs         /var/lib/docker/devicemapper/mnt/e9733ecdda1
  #   ##/dev/mapper/docker-8:1-16786822-947636812ed     45G dm   xfs         /var/lib/docker/devicemapper/mnt/947636812ed
  #   ##/dev/mapper/docker-8:1-16786822-d65a56e4300     45G dm   xfs         /var/lib/docker/devicemapper/mnt/d65a56e4300
  #   ##/dev/mapper/docker-8:1-16786822-3d1fb13572c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3d1fb13572c
  #   ##/dev/mapper/docker-8:1-16786822-542e3e08ccc     45G dm   xfs         /var/lib/docker/devicemapper/mnt/542e3e08ccc
  #   ##/dev/mapper/docker-8:1-16786822-3beb7120639     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3beb7120639
  #   ##/dev/mapper/docker-8:1-16786822-1c88db1b91c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/1c88db1b91c
  #   ##/dev/mapper/docker-8:1-16786822-3a246ac134c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3a246ac134c
  #   ##/dev/mapper/docker-8:1-16786822-045933b4418     45G dm   xfs         /var/lib/docker/devicemapper/mnt/045933b4418
  #   ##/dev/mapper/docker-8:1-16786822-e420dfc0229     45G dm   xfs         /var/lib/docker/devicemapper/mnt/e420dfc0229
  #   ##/dev/mapper/docker-8:1-16786822-918a6f92734     45G dm   xfs         /var/lib/docker/devicemapper/mnt/918a6f92734
  #   ##/dev/mapper/docker-8:1-16786822-a7efd4f15c3     45G dm   xfs         /var/lib/docker/devicemapper/mnt/a7efd4f15c3
  #   ##/dev/mapper/docker-8:1-16786822-17b847994e2     45G dm   xfs         /var/lib/docker/devicemapper/mnt/17b847994e2
  #   ##/dev/mapper/docker-8:1-16786822-3564e748e14     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3564e748e14
  ##/dev/mapper/docker-docker--pool_tdata              42.2G lvm
    ##/dev/mapper/docker-docker--pool                  42.2G lvm
      ##/dev/mapper/docker-8:1-16786822-2a6c2771415     45G dm   xfs         /var/lib/docker/devicemapper/mnt/2a6c2771415
      ##/dev/mapper/docker-8:1-16786822-579a2fb3611     45G dm   xfs         /var/lib/docker/devicemapper/mnt/579a2fb3611
      ##/dev/mapper/docker-8:1-16786822-c95ca5f04a4     45G dm   xfs         /var/lib/docker/devicemapper/mnt/c95ca5f04a4
      ##/dev/mapper/docker-8:1-16786822-e9733ecdda1     45G dm   xfs         /var/lib/docker/devicemapper/mnt/e9733ecdda1
      ##/dev/mapper/docker-8:1-16786822-947636812ed     45G dm   xfs         /var/lib/docker/devicemapper/mnt/947636812ed
      ##/dev/mapper/docker-8:1-16786822-d65a56e4300     45G dm   xfs         /var/lib/docker/devicemapper/mnt/d65a56e4300
      ##/dev/mapper/docker-8:1-16786822-3d1fb13572c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3d1fb13572c
      ##/dev/mapper/docker-8:1-16786822-542e3e08ccc     45G dm   xfs         /var/lib/docker/devicemapper/mnt/542e3e08ccc
      ##/dev/mapper/docker-8:1-16786822-3beb7120639     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3beb7120639
      ##/dev/mapper/docker-8:1-16786822-1c88db1b91c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/1c88db1b91c
      ##/dev/mapper/docker-8:1-16786822-3a246ac134c     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3a246ac134c
      ##/dev/mapper/docker-8:1-16786822-045933b4418     45G dm   xfs         /var/lib/docker/devicemapper/mnt/045933b4418
      ##/dev/mapper/docker-8:1-16786822-e420dfc0229     45G dm   xfs         /var/lib/docker/devicemapper/mnt/e420dfc0229
      ##/dev/mapper/docker-8:1-16786822-918a6f92734     45G dm   xfs         /var/lib/docker/devicemapper/mnt/918a6f92734
      ##/dev/mapper/docker-8:1-16786822-a7efd4f15c3     45G dm   xfs         /var/lib/docker/devicemapper/mnt/a7efd4f15c3
      ##/dev/mapper/docker-8:1-16786822-17b847994e2     45G dm   xfs         /var/lib/docker/devicemapper/mnt/17b847994e2
      ##/dev/mapper/docker-8:1-16786822-3564e748e14     45G dm   xfs         /var/lib/docker/devicemapper/mnt/3564e748e14
/dev/sdc                                             15.6G disk
##/dev/sdc1                                          15.6G part swap        [SWAP]
/dev/sdd                                             15.6G disk
##/dev/sdd1                                          15.6G part xfs         /tmp
/dev/sr0                                             1024M rom
```

Operating system requirements

Control Center has tested the 64-bit version of the following Linux distributions:

- Red Hat Enterprise Linux (RHEL) 7.2, 7.3, or 7.4
- CentOS 7.2, 7.3, or 7.4

Kernel versions `3.10.0-327.22.2.el7.x86_64` and higher are tested. For best performance, keep the kernel up-to-date.

The RHEL/CentOS 7 distributions provide a variety of server configurations. Control Center is tested on operating system platforms that are installed and configured with standard options. Docker and Control Center are tested on the Minimal Install configuration with the NFS and Network Time Protocol (NTP) packages installed.

Control Center relies on the system clock to synchronize its actions. The installation procedures include steps to add the NTP daemon to all hosts. By default, the NTP daemon synchronizes the system clock by communicating with standard time servers available on the internet. Use the default configuration or configure the daemon to use a time server in your environment.

Note Because Control Center relies on the system clock, while an application is running, do not pause a virtual machine that is running Control Center.

Networking

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 through this virtual bridge.

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

The following list highlights potential communication conflicts:

- If you use a firewall utility, ensure that it does not conflict with Docker. The default configurations of firewall utilities such as *FirewallD* include rules that can conflict with Docker, and therefore Control Center. 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 must restart Docker.
- Even if you do not use a firewall utility, your firewall settings might still prevent communications over the Docker virtual bridge. This issue 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 a command like the following example would ensure that the bridge works.

Note Before modifying your `iptables` configuration, consult your networking specialist.

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

Additional requirements and considerations

Control Center requires a 16-bit, private IPv4 network for virtual IP addresses. The default network is 10.3/16, but during installation you can select any valid IPv4 16-bit address space.

Before installation, add DNS entries for the Control Center master host and all delegate hosts. Verify that all hosts in Control Center resource pools can

- Resolve the hostnames of all other delegate 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.

Control Center relies on Network File System (NFS) for its distributed file system implementation. Therefore, Control Center hosts cannot run a general-purpose NFS server, and all Control Center hosts require NFS.

Note Disabling IPv6 can prevent the NFS server from restarting, due to *an rpcbind issue*. Zenoss recommends leaving IPv6 enabled on the Control Center master host.

Security

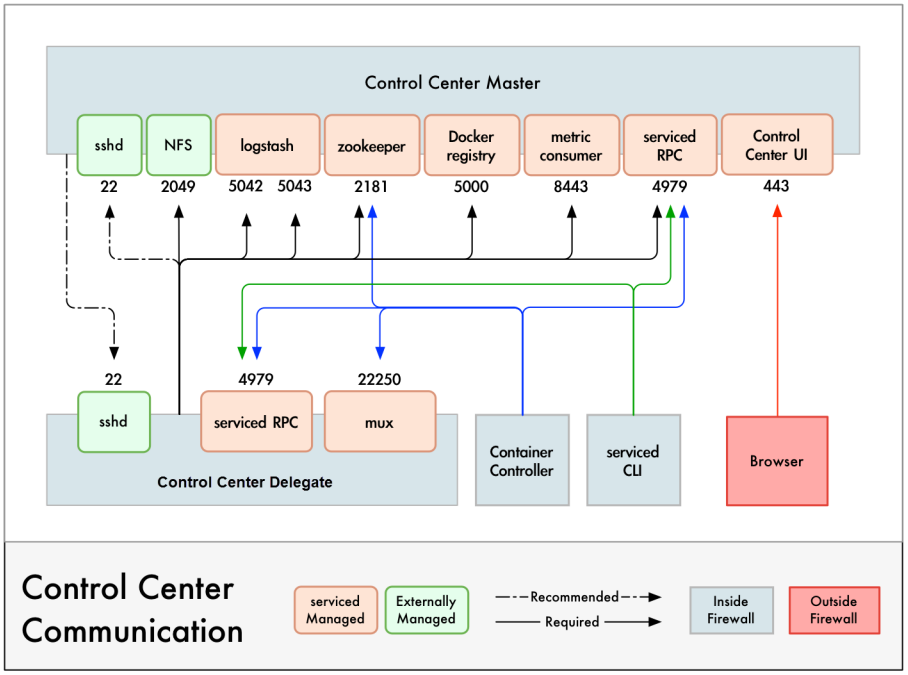
During installation, Control Center has no knowledge of Zenoss Core port requirements, so the installation procedure includes disabling the firewall. After both Control Center and Zenoss Core are installed, you can close unused ports.

Control Center includes a virtual multiplexer (mux) that performs the following functions:

- Aggregates 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.)
- Along with the distributed file system, enables Control Center to quickly deploy services to any pool host.
- Reduces the number of open ports required on a Control Center host to a predictable set.

The following figure identifies the ports that Control Center requires. All traffic is TCP. Except for port 4979, all ports are configurable.

Figure 3: Port requirements for Control Center hosts



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 delegate hosts must support an incoming UDP connection on port 123.

Additional requirements and considerations

- To install Control Center, you must log in as `root`, or as a user with superuser privileges.
- Access to the Control Center browser interface requires a login account on the Control Center master host. Pluggable Authentication Modules (PAM) is tested. By default, users must be members of the `wheel` group. The default group may be changed by setting the `SERVICED_ADMIN_GROUP` variable, and the replacement group does not need superuser privileges.
- The `serviced` startup script sets the hard and soft open files limit to 1048576. The script does not modify the `/etc/sysconfig/limits.conf` file.
- Control Center has been tested with *Security Enhanced Linux* enabled.

Delegate host authentication

Control Center uses RSA key pairs to create the authentication tokens that are required for all delegate communications. When you add a host to a resource pool, the `serviced` instance on the master host creates a private key for the delegate and bundles it with its own public key. The `serviced` instance on the delegate host uses the bundle to sign messages with its unique tokens.

Key bundles are installed by using an SSH connection or a file.

- The command to add a host to a pool can initiate an SSH connection with the delegate and install the key bundle. This option is the most secure because no file is created. However, it requires either public key authentication or password authentication between the master and delegate hosts.
- When no SSH connection is requested, the command to add a host to a pool creates a file that contains the key bundle. You can move the key bundle file to the delegate host with any file transfer method, and then install it on the delegate.