Zenoss Core Installation Guide

Release 5.1.4

Zenoss, Inc.

www.zenoss.com

Zenoss Core Installation Guide

Part Number: 1051.16.181

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

Contents

About this guide

Zenoss Core Installation Guide provides detailed procedures for installing Zenoss Core.

Note Zenoss strongly recommends reviewing the *Zenoss Core Planning Guide* carefully before using this guide.

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.

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 Core Release Notes*.

1051.16.181

Update release numbers.

1051.16.153

Update release numbers.

1051.16.146

Update release numbers.

1051.16.125

Refine the procedure for creating the application data thin pool.

1051.16.118

Add support for Zenoss Core 5.1.2.

Add a substep to create the docker override directory.

1051.16.111

Add this document change history.

Add chapters describing how to install the Zenoss Core appliance.

Chapters are organized into parts.

Docker configuration steps now add the storage driver flag (`-s devicemapper`) to the `/etc/sysconfig/docker` file.

Docker needs a longer startup timeout value, to work around a known Docker issue with the `devicemapper` driver. Docker configuration steps now include adding `TimeoutSec=300`.

Rather than editing `/lib/systemd/system/docker.service`, Docker configuration steps now include adding a `systemd` override file.

Add a symlink to `/tmp` in `/var/lib/docker`.

Update the commands for starting and testing a ZooKeeper ensemble.

Add a procedure for updating the *SERVICED_ZK* value on resource pool hosts that are not members of a ZooKeeper ensemble.

Add a reference topic for the ZooKeeper variables required on hosts in a Control Center cluster.

Add step to install the Nmap Ncat package, which is used to check ZooKeeper ensemble status.

1051.16.060

Planning information is now in the *Zenoss Core Planning Guide*.

Information about how to start and configure Zenoss Core is now in the *Zenoss Core Configuration Guide*.

1 Installing on hosts with internet access

The procedures in this chapter install Control Center and Zenoss Core on one or more Red Hat Enterprise Linux (RHEL) 7.1 or 7.2 hosts, or one or more CentOS 7.1 or 7.2 hosts. To use the procedures in this chapter, all Control Center cluster hosts must have internet access.

You may create a single-host or a multi-host deployment. For production use, Zenoss strongly recommends creating a multi-host deployment that includes a minimum of three real or virtual machines. For more information about deploying Control Center and Zenoss Core, refer to the *Zenoss Core Planning Guide*.

Note For optimal results, review this chapter thoroughly before starting the installation process.

Installing a master host

Perform the procedures in this section to install Control Center and Zenoss Core on a master host.

Verifying candidate host resources

This procedure determines whether a hosts's hardware resources and operating system are sufficient to serve as a Control Center master host.

1. Log in to the candidate host as `root`, or as a user with superuser privileges.
2. Verify that the host implements the 64-bit version of the x86 instruction set.

   ```
   uname -m
   ```

   - If the output is `x86_64`, the architecture is 64-bit. Proceed to the next step
   - If the output is `i386/i486/i586/i686`, the architecture is 32-bit. Stop this procedure and select a different host.
3. Verify that name resolution works on this host.

   ```
   hostname -i
   ```

   If the result is not a valid IPv4 adddress, add an entry for the host to the network nameserver, or to `/etc/hosts`.
4. Verify that the host's numeric identifier is unique.

   Each host in a Control Center cluster must have a unique host identifier.

   ```
   hostid
   ```

5 Determine whether the available, unused storage is sufficient.
   a Display the available storage devices.

```
lsblk --output=NAME,SIZE
```

   b Compare the available storage with the amount required for a Control Center master host.
   For more information, refer to the *Zenoss Core Planning Guide*.
6 Determine whether the available memory and swap is sufficient.
   a Display the available memory.

```
free -h
```

   b Compare the available memory with the amount required for a master host in your deployment.
   For more information, refer to the *Zenoss Core Planning Guide*.
7 Update the operating system, if necessary.
   a Determine which release is installed.

```
cat /etc/redhat-release
```

   If the result includes `7.0`, perform the following substeps.
   b Update the operating system.

```
yum update -y
```

   c Restart the system.

```
reboot
```

Preparing storage for the master host

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

- A local partition for Docker data, configured as a device mapper thin pool.
- A local partition for Control Center internal services data, formatted with the XFS file system.

> **Note** Control Center internal services include ZooKeeper, which requires consistently fast storage. Zenoss recommends using a separate, high-performance storage resource for Control Center internal services. For example, a drive that is configured with only one primary partition, which eliminates contention by other services.

- A local or remote primary partition for Zenoss Core data, configured as a device mapper thin pool.
- A local primary partition, a remote primary partition, or a remote file server, for backups of Zenoss Core data. The local or remote primary partition is formatted with the XFS file system. A remote file server must provide a file system that is compatible with XFS.

> **Note** If you are using a primary partition on a local device for backups, ensure that the primary partition for Control Center internal services data is not on the same device.

For storage sizing information, refer to the *Zenoss Core Planning Guide*.

For device mapper thin pools, no formatting is required—simply create primary partitons, which are configured in subsequent procedures. For more information, refer to the *Zenoss Core Planning Guide*.

To create the required storage, perform the following procedures.

Note Data present on the primary partitions you select are destroyed in these procedure. Please ensure that data is backed up elsewhere, or no longer needed, before proceeding.

Creating a file system for internal services

This procedure creates an XFS file system on a primary partition. For more information about primary partitions, refer to the *Zenoss Core Planning Guide*.

Note Control Center internal services include ZooKeeper, which requires consistently fast storage. Zenoss recommends using a separate, high-performance storage resource for Control Center internal services. For example, a drive that is configured with only one primary partition, which eliminates contention by other services.

1. Log in to the target host as `root`, or as a user with superuser privileges.
2. Identify the target primary partition for the file system to create.

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

   For more information about the output of the `lsblk` command, and about creating primary partitions, refer to the *Zenoss Core Planning Guide*.
3. Create an XFS file system.
   Replace *Partition* with the path of the target primary partition:

   ```
   mkfs -t xfs Partition
   ```
4. Add an entry to the `/etc/fstab` file.
   Replace *Partition* with the path of the primary partition used in the previous step:

   ```
   echo "Partition \
     /opt/serviced/var/isvcs xfs defaults 0 0" >> /etc/fstab
   ```
5. Create the mount point for internal services data.

   ```
   mkdir -p /opt/serviced/var/isvcs
   ```
6. Mount the file system, and then verify it mounted correctly.

   ```
   mount -a && mount | grep isvcs
   ```

   Example result:

   ```
   /dev/xvdb1 on /opt/serviced/var/isvcs type xfs
    (rw,relatime,seclabel,attr2,inode64,noquota)
   ```

Creating a file system for backups

To perform this procedure, you need a host with at least one unused primary partition, or a remote file server.

The Control Center master host requires local or remote storage space for backups of Control Center data. This procedure includes steps to create an XFS file system on a primary partition, if necessary, and steps to mount a file system for backups. For more information about primary partitions, refer to the *Zenoss Core Planning Guide*.

Note If you are using a primary partition on a local device for backups, ensure that the primary partition for Control Center internal services data is not on the same device.

1. Log in to the target host as `root`, or as a user with superuser privileges.
2. Optional: Identify the target primary partition for the file system to create, if necessary.

   Skip this step if you are using a remote file server.

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

   For more information about the output of the `lsblk` command, and about creating primary partitions, refer to the *Zenoss Core Planning Guide*.
3. Optional: Create an XFS file system, if necessary.

   Skip this step if you are using a remote file server.

   Replace *Partition* with the path of the target primary partition:

   ```
   mkfs -t xfs Partition
   ```
4. Create an entry in the `/etc/fstab` file.

   Replace *File-System-Specification* with one of the following values:

   - the path of the primary partition used in the previous step
   - the remote server specification

   ```
   echo "File-System-Specification \
     /opt/serviced/var/backups xfs defaults 0 0" >> /etc/fstab
   ```
5. Create the mount point for backup data.

   ```
   mkdir -p /opt/serviced/var/backups
   ```
6. Mount the file system, and then verify it mounted correctly.

   ```
   mount -a && mount | grep backups
   ```

   Example result:

   ```
   /dev/sdb3 on /opt/serviced/var/backups type xfs
    (rw,relatime,seclabel,attr2,inode64,noquota)
   ```

Preparing the master host operating system

This procedure prepares a RHEL/CentOS 7.1 or 7.2 host as a Control Center master host.

1. Log in to the candidate master host as `root`, or as a user with superuser privileges.
2. Add an entry to `/etc/hosts` for localhost, if necessary.
   a. Determine whether `127.0.0.1` is mapped to `localhost`.

      ```
      grep 127.0.0.1 /etc/hosts | grep localhost
      ```

      If the preceding commands return no result, perform the following substep.
   b. Add an entry to `/etc/hosts` for `localhost`.

      ```
      echo "127.0.0.1   localhost" >> /etc/hosts
      ```
3. Disable the firewall, if necessary.

   This step is required for installation but not for deployment. For more information, refer to the *Zenoss Core Planning Guide*.

a. Determine whether the `firewalld` service is enabled.

```
systemctl status firewalld.service
```

- If the result includes `Active: inactive (dead)`, the service is disabled. Proceed to the next step.
- If the result includes `Active: active (running)`, the service is enabled. Perform the following substep.

b. Disable the `firewalld` service.

```
systemctl stop firewalld && systemctl disable firewalld
```

On success, the preceding commands display messages similar to the following example:

```
rm '/etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service'
rm '/etc/systemd/system/basic.target.wants/firewalld.service'
```

4. Optional: Enable persistent storage for log files, if desired.

By default, RHEL/CentOS systems store log data only in memory or in a small ring-buffer in the `/run/log/journal` directory. By performing this step, log data persists and can be saved indefinitely, if you implement log file rotation practices. For more information, refer to your operating system documentation.

```
mkdir -p /var/log/journal && systemctl restart systemd-journald
```

5. Disable Security-Enhanced Linux (SELinux), if installed.

a. Determine whether SELinux is installed.

```
test -f /etc/selinux/config && grep '^SELINUX=' /etc/selinux/config
```

If the preceding commands return a result, SELinux is installed.

b. Set the operating mode to `disabled`.
Open `/etc/selinux/config` in a text editor, and change the value of the *SELINUX* variable to `disabled`.

c. Confirm the new setting.

```
grep '^SELINUX=' /etc/selinux/config
```

6. Enable and start the Dnsmasq package.

```
systemctl enable dnsmasq && systemctl start dnsmasq
```

7. Install the *Nmap Ncat* utility.

The utility is used to verify ZooKeeper ensemble configurations. **If you are installing a single-host deployment, skip this step**.

```
yum install -y nmap-ncat
```

8. Install and configure the NTP package.

a. Install the package.

```
yum install -y ntp
```

b Set the system time.

```
ntpd -gq
```

c Enable the `ntpd` daemon.

```
systemctl enable ntpd
```

d Configure `ntpd` to start when the system starts.

Currently, an unresolved issue associated with NTP prevents `ntpd` from restarting correctly after a reboot. The following commands provide a workaround to ensure that it does.

```
echo "systemctl start ntpd" >> /etc/rc.d/rc.local
chmod +x /etc/rc.d/rc.local
```

9 Install the Zenoss repository package.

a Install the package.

```
rpm -ivh http://get.zenoss.io/yum/zenoss-repo-1-1.x86_64.rpm
```

b Clean out the `yum` cache directory.

```
yum clean all
```

10 Reboot the host.

```
reboot
```

Installing Docker and Control Center

This procedure installs and configures Docker, and installs Control Center.

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

2 Install Docker 1.9.0, and then disable accidental upgrades.

a Add the Docker repository to the host's repository list.

```
cat > /etc/yum.repos.d/docker.repo <<-EOF
[dockerrepo]
name=Docker Repository
baseurl=https://yum.dockerproject.org/repo/main/centos/7
enabled=1
gpgcheck=1
gpgkey=https://yum.dockerproject.org/gpg
EOF
```

b Install Docker 1.9.0.

```
yum clean all && yum makecache fast
yum install -y docker-engine-1.9.0
```

c Open `/etc/yum.repos.d/docker.repo` with a text editor.

d Change the value of the `enabled` key from `1` to `0`.

e Save the file and close the text editor.

3 Create a symbolic link for the Docker temporary directory.

Docker uses its temporary directory to spool images. The default directory is `/var/lib/docker/tmp`. The following command specifies the same directory that Control Center uses, `/tmp`. You can specify any directory that has a minimum of 10GB of unused space.

a Create the `docker` directory in `/var/lib`.

```
mkdir /var/lib/docker
```

b Create the link to `/tmp`.

```
ln -s /tmp /var/lib/docker/tmp
```

4 Create a `systemd` override file for the Docker service definition.

a Create the override directory.

```
mkdir -p /etc/systemd/system/docker.service.d
```

b Create the override file.

```
cat <<EOF > /etc/systemd/system/docker.service.d/docker.conf
[Service]
TimeoutSec=300
EnvironmentFile=-/etc/sysconfig/docker
ExecStart=
ExecStart=/usr/bin/docker daemon \$OPTIONS -H fd://
EOF
```

c Reload the `systemd` manager configuration.

```
systemctl daemon-reload
```

5 Install Control Center.

Control Center includes a utility that simplifies the process of creating a device mapper thin pool.

```
yum clean all && yum makecache fast
yum --enablerepo=zenoss-stable install -y serviced-1.1.6
```

6 Create a device mapper thin pool for Docker data.

a Identify the primary partition for the thin pool to create.

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

b Create the thin pool.

Replace *Path-To-Device* with the path of an unused primary partition:

```
serviced-storage create-thin-pool docker Path-To-Device
```

On success, the result includes the name of the thin pool, which always starts with `/dev/mapper`.

7 Configure and start the Docker service.

a Create variables for adding arguments to the Docker configuration file.

The `--exec-opt` argument is a workaround for *a Docker issue* on RHEL/CentOS 7.x systems.

Replace *Thin-Pool-Device* with the name of the thin pool device created in the previous step:

```
myDriver="-s devicemapper"
myFix="--exec-opt native.cgroupdriver=cgroupfs"
```

```
myFlag="--storage-opt dm.thinpooldev"
myPool="Thin-Pool-Device"
```

b Add the arguments to the Docker configuration file.

```
echo 'OPTIONS="'$myDriver $myFix $myFlag'='$myPool'"' \
  >> /etc/sysconfig/docker
```

c Start or restart Docker.

```
systemctl restart docker
```

The initial startup takes up to a minute, and may fail. If the startup fails, repeat the previous command.

8 Configure name resolution in containers.

Each time it starts, `docker` selects an IPv4 subnet for its virtual Ethernet bridge. The selection can change; this step ensures consistency.

a Identify the IPv4 subnet and netmask `docker` has selected for its virtual Ethernet bridge.

```
ip addr show docker0 | grep inet
```

b Open `/etc/sysconfig/docker` in a text editor.

c Add the following flags to the end of the *OPTIONS* declaration.

Replace *Bridge-Subnet* with the IPv4 subnet `docker` selected for its virtual bridge, and replace *Bridge-Netmask* with the netmask `docker` selected:

```
--dns=Bridge-Subnet --bip=Bridge-Subnet/Bridge-Netmask
```

For example, if the bridge subnet and netmask is 172.17.0.1/16, the flags to add are `--dns=172.17.0.1 --bip=172.17.0.1/16`.

Note Leave a blank space after the end of the thin pool device name, and make sure the double quote character (") is at the end of the line.

---

d Restart the Docker service.

```
systemctl restart docker
```

Installing Zenoss Core

This procedure installs Zenoss Core and configures the NFS server.

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

2 Install Zenoss Core.

```
yum --enablerepo=zenoss-stable install -y zenoss-core-service
```

3 Configure and restart the NFS server.

Currently, *an unresolved issue* prevents the NFS server from starting correctly. The following commands provide a workaround to ensure that it does.

a Open `/lib/systemd/system/nfs-server.service` with a text editor.

b Change `rpcbind.target` to `rpcbind.service` on the following line:

```
Requires= network.target proc-fs-nfsd.mount rpcbind.target
```

c Reload the `systemd` manager configuration.

```
systemctl daemon-reload
```

Configuring Control Center

This procedure creates a thin pool for application data and customizes key configuration variables of Control Center.

1. Log in to the master host as `root`, or as a user with superuser privileges.
2. Configure Control Center to serve as the master and as an agent.

   The following variables configure `serviced` to serve as both master and agent:

   `SERVICED_AGENT`

   Default: `0` (false)

   Determines whether a `serviced` instance performs agent tasks. Agents run application services scheduled for the resource pool to which they belong. The `serviced` instance configured as the master runs the scheduler. A `serviced` instance may be configured as agent and master, or just agent, or just master.

   `SERVICED_MASTER`

   Default: `0` (false)

   Determines whether a `serviced` instance performs master tasks. The master runs the application services scheduler and other internal services, including the server for the Control Center browser interface. A `serviced` instance may be configured as agent and master, or just agent, or just master. Only one `serviced` instance in a Control Center cluster may be the master.

   a Open `/etc/default/serviced` in a text editor.

   b Find the *SERVICED_AGENT* declaration, and then change the value from `0` to `1`.
   The following example shows the line to change:

   ```
   # SERVICED_AGENT=0
   ```

   c Remove the number sign character (#) from the beginning of the line.

   d Find the *SERVICED_MASTER* declaration, and then change the value from `0` to `1`.
   The following example shows the line to change:

   ```
   # SERVICED_MASTER=0
   ```

   e Remove the number sign character (#) from the beginning of the line.

   f Save the file, and then close the editor.

3. Create a thin pool for Zenoss Core data.

   a Identify the primary partition for the thin pool to create, and the amount of space available on the primary partition.

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

   For more information about the output of the `lsblk` command and primary partitions, refer to the *Zenoss Core Planning Guide*.

   b Create a variable for 50% of the space available on the primary partition for the thin pool to create.

   The thin pool stores application data and snapshots of the data. You can add storage to the pool at any time.

Replace *Half-Of-Available-Space* with 50% of the space available in the primary partition, in gigabytes. Include the symbol for gigabytes (G) after the numeric value.

```
myFifty=Half-Of-Available-SpaceG
```

c Create the thin pool.

Replace *Path-To-Device* with the path of the target primary partition:

```
serviced-storage create-thin-pool -o dm.basesize=$myFifty \
  serviced Path-To-Device
```

On success, the result includes the name of the thin pool, which always starts with `/dev/mapper`.

4 Configure Control Center with the name of the thin pool for Zenoss Core data.

The Control Center configuration file is `/etc/default/serviced`. (For more information about `serviced` configuration options, refer to the Control Center online help.)

a Open `/etc/default/serviced` in a text editor.
b Locate the *SERVICED_FS_TYPE* declaration.
c Remove the number sign character (#) from the beginning of the line.
d Add *SERVICED_DM_THINPOOLDEV* immediately after *SERVICED_FS_TYPE*.

Replace *Thin-Pool-Name* with the name of the thin pool created previously:

```
SERVICED_DM_THINPOOLDEV=Thin-Pool-Name
```

e Save the file, and then close the editor.

5 Optional: Specify an alternate private subnet for Control Center, if necessary.

The default private subnet may already be in use in your environment. The following variable configures `serviced` to use an alternate subnet:

SERVICED_VIRTUAL_ADDRESS_SUBNET

Default: `10.3`

The 16-bit private subnet to use for `serviced`'s virtual IPv4 addresses. RFC 1918 restricts private networks to the 10.0/24, 172.16/20, and 192.168/16 address spaces. However, `serviced` accepts any valid, 16-bit, IPv4 address space for its private network.

a Open `/etc/default/serviced` in a text editor.
b Locate the *SERVICED_VIRTUAL_ADDRESS_SUBNET* declaration, and then change the value. The following example shows the line to change:

```
# SERVICED_VIRTUAL_ADDRESS_SUBNET=10.3
```

c Remove the number sign character (#) from the beginning of the line.
d Save the file, and then close the editor.

User access control

Control Center provides a browser interface and a command-line interface.

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.) In addition, users must be members of the Control Center administrative group, which by default is the system group, `wheel`. To enhance security, you may change the administrative group from `wheel` to any non-system group.

To use the Control Center command-line interface, users must have login accounts on the Control Center master host, and be members of the `docker` user group. Members of the `wheel` group, including `root`, are members of the `docker` group.

Adding users to the default administrative group

This procedure adds users to the default administrative group of Control Center, `wheel`. Performing this procedure enables users with superuser privileges to gain access to the Control Center browser interface.

Note Perform this procedure or the next procedure, but not both.

1. Log in to the host as `root`, or as a user with superuser privileges.
2. Add users to the system group, `wheel`.

   Replace *User* with the name of a login account on the master host.

   ```
   usermod -aG wheel User
   ```

   Repeat the preceding command for each user to add.

   Note For information about using Pluggable Authentication Modules (PAM), refer to your operating system documentation.

Configuring a regular group as the Control Center administrative group

This procedure changes the default administrative group of Control Center from `wheel` to a non-system group.

Note Perform this procedure or the previous procedure, but not both.

1. Log in to the Control Center master host as `root`, or as a user with superuser privileges.
2. Create a variable for the group to designate as the administrative group.

   In this example, the name of group to create is `serviced`. You may choose any name or use an existing group.

   ```
   GROUP=serviced
   ```
3. Create a new group, if necessary.

   ```
   groupadd $GROUP
   ```
4. Add one or more existing users to the new administrative group.

   Replace *User* with the name of a login account on the host:

   ```
   usermod -aG $GROUP User
   ```

   Repeat the preceding command for each user to add.
5. Specify the new administrative group in the `serviced` configuration file.

   The following variable specifies the administrative group:

   `SERVICED_ADMIN_GROUP`

   Default: `wheel`

   The name of the Linux group on the Control Center master host whose members are authorized to use the Control Center browser interface. You may replace the default group with a group that does not have superuser privileges.

   a. Open `/etc/default/serviced` in a text editor.

b Find the *SERVICED_ADMIN_GROUP* declaration, and then change the value from `wheel` to the name of the group you chose earlier.
The following example shows the line to change:

```
# SERVICED_ADMIN_GROUP=wheel
```

c Remove the number sign character (#) from the beginning of the line.
d Save the file, and then close the editor.

6 Optional: Prevent `root` users and members of the `wheel` group from gaining access to the Control Center browser interface, if desired.

The following variable controls privileged logins:

`SERVICED_ALLOW_ROOT_LOGIN`

Default: `1` (true)

Determines whether `root`, or members of the `wheel` group, may gain access to the Control Center browser interface.

a Open `/etc/default/serviced` in a text editor.
b Find the *SERVICED_ALLOW_ROOT_LOGIN* declaration, and then change the value from `1` to `0`.
The following example shows the line to change:

```
# SERVICED_ALLOW_ROOT_LOGIN=1
```

c Remove the number sign character (#) from the beginning of the line.
d Save the file, and then close the editor.

Enabling use of the command-line interface

This procedure enables users to perform administrative tasks with the Control Center command-line interface by adding individual users to the `docker` group.

1 Log in to the Control Center master host as `root`, or as a user with superuser privileges.
2 Add users to the Docker group, `docker`.

Replace *User* with the name of a login account on the host.

```
usermod -aG docker User
```

Repeat the preceding command for each user to add.

Starting Control Center

This procedure starts the Control Center service, `serviced`.

1 Log in to the master host as `root`, or as a user with superuser privileges.
2 Start `serviced`.

```
systemctl start serviced
```

To monitor progress, enter the following command:

```
journalctl -flu serviced -o cat
```

The `serviced` daemon invokes `docker` to pull its internal services images from Docker Hub. The Control Center browser and command-line interfaces are unavailable until the images are installed and the services are started. The process takes approximately 5-10 minutes. When the message `Trying to discover my pool` repeats, Control Center is ready for the next steps.

3 Note Perform this step **only if you are installing a single-host deployment**.

Optional: Add the master host to the `default` resource pool.

Replace *Hostname-Or-IP* with the hostname or IP address of the Control Center master host:

```
serviced host add Hostname-Or-IP:4979 default
```

If you enter a hostname, all hosts in your Control Center cluster must be able to resolve the name, either through an entry in `/etc/hosts`, or through a nameserver on your network.

Isolating the master host in a separate resource pool

Note If you are configuring a single-host deployment, skip this procedure.

Control Center enables or just performs rapid recovery from application service failures. When Control Center internal services and application services share a host, application failures can limit recovery options. Zenoss strongly recommends isolating the Control Center master host in a separate resource pool.

This procedure creates a new resource pool for the Control Center master host, and then adds the master host to the pool.

1 Log in to the master host as `root`, or as a user with superuser privileges.
2 Create a new resource pool named `master`.

```
serviced pool add master
```

3 Add the master host to the `master` resource pool.

Replace *Hostname-Or-IP* with the hostname or IP address of the Control Center master host:

```
serviced host add Hostname-Or-IP:4979 master
```

If you enter a hostname, all hosts in your Control Center cluster must be able to resolve the name, either through an entry in `/etc/hosts`, or through a nameserver on your network.

Installing resource pool hosts

Note If you are installing a single-host deployment, skip this section.

Control Center resource pool hosts run the application services scheduled for the resource pool to which they belong, and for which they have sufficent RAM and CPU resources.

Zenoss Core has two broad categories of application services: Infrastructure and collection. The services associated with each category can run in the same resource pool, or can run in separate resource pools.

For improved reliability, two resource pool hosts are configured as nodes in an *Apache ZooKeeper* ensemble. The storage required for ensemble hosts is slightly different than the storage required for all other resource pool hosts: Each ensemble host requires a separate primary partition for Control Center internal services data, in addition to the primary partition for Docker data. Unless the ZooKeeper service on the Control Center master host fails, their roles in the ZooKeeper ensemble do not affect their roles as Control Center resource pool hosts.

Note The hosts for the ZooKeeper ensemble require static IP addresses, because ZooKeeper does not support hostnames in its configurations.

Repeat the procedures in the following sections for each host you wish to add to your Control Center deployment.

Verifying candidate host resources

This procedure determines whether a hosts's hardware resources and operating system are sufficient to serve as a Control Center resource pool host.

Perform this procedure on each resource pool host in your deployment.

1. Log in to the candidate host as `root`, or as a user with superuser privileges.
2. Verify that the host implements the 64-bit version of the x86 instruction set.

   ```
   uname -m
   ```

   - If the output is `x86_64`, the architecture is 64-bit. Proceed to the next step
   - If the output is `i386/i486/i586/i686`, the architecture is 32-bit. Stop this procedure and select a different host.
3. Verify that name resolution works on this host.

   ```
   hostname -i
   ```

   If the result is not a valid IPv4 adddress, add an entry for the host to the network nameserver, or to `/etc/hosts`.
4. Verify that the host's numeric identifier is unique.

   Each host in a Control Center cluster must have a unique host identifier.

   ```
   hostid
   ```
5. Determine whether the available, unused storage is sufficient.
   a. Display the available storage devices.

      ```
      lsblk --output=NAME,SIZE
      ```

   b. Compare the available storage with the amount required for a resource pool host in your deployment.

      In particular, resource pool hosts that are configured as nodes in a ZooKeeper ensemble require an additional primary partition for Control Center internal services data.

      For more information, refer to the *Zenoss Core Planning Guide*.
6. Determine whether the available memory and swap is sufficient.
   a. Display the available memory.

      ```
      free -h
      ```

   b. Compare the available memory with the amount required for a resource pool host in your deployment.

      For more information, refer to the *Zenoss Core Planning Guide*.
7. Update the operating system, if necessary.
   a. Determine which release is installed.

      ```
      cat /etc/redhat-release
      ```

      If the result includes `7.0`, perform the following substeps.
   b. Update the operating system.

      ```
      yum update -y
      ```

c Restart the system.

```
reboot
```

Preparing a resource pool host

This procedure prepares a RHEL/CentOS 7.1 or 7.2 host as a Control Center resource pool host.

1. Log in to the candidate resource pool host as `root`, or as a user with superuser privileges.
2. Add an entry to `/etc/hosts` for localhost, if necessary.
   a Determine whether `127.0.0.1` is mapped to `localhost`.

   ```
   grep 127.0.0.1 /etc/hosts | grep localhost
   ```

   If the preceding commands return no result, perform the following substep.
   b Add an entry to `/etc/hosts` for `localhost`.

   ```
   echo "127.0.0.1   localhost" >> /etc/hosts
   ```
3. Disable the firewall, if necessary.

   This step is required for installation but not for deployment. For more information, refer to the *Zenoss Core Planning Guide*.
   a Determine whether the `firewalld` service is enabled.

   ```
   systemctl status firewalld.service
   ```

   - If the result includes `Active: inactive (dead)`, the service is disabled. Proceed to the next step.
   - If the result includes `Active: active (running)`, the service is enabled. Perform the following substep.

   b Disable the `firewalld` service.

   ```
   systemctl stop firewalld && systemctl disable firewalld
   ```

   On success, the preceding commands display messages similar to the following example:

   ```
   rm '/etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service'
   rm '/etc/systemd/system/basic.target.wants/firewalld.service'
   ```
4. Optional: Enable persistent storage for log files, if desired.

   By default, RHEL/CentOS systems store log data only in memory or in a small ring-buffer in the `/run/log/journal` directory. By performing this step, log data persists and can be saved indefinitely, if you implement log file rotation practices. For more information, refer to your operating system documentation.

   ```
   mkdir -p /var/log/journal && systemctl restart systemd-journald
   ```
5. Disable Security-Enhanced Linux (SELinux), if installed.
   a Determine whether SELinux is installed.

   ```
   test -f /etc/selinux/config && grep '^SELINUX=' /etc/selinux/config
   ```

   If the preceding commands return a result, SELinux is installed.
   b Set the operating mode to `disabled`.

Open `/etc/selinux/config` in a text editor, and change the value of the *SELINUX* variable to `disabled`.

c Confirm the new setting.

```
grep '^SELINUX=' /etc/selinux/config
```

6 Enable and start the Dnsmasq package.

```
systemctl enable dnsmasq && systemctl start dnsmasq
```

7 Install and configure the NTP package.

a Install the package.

```
yum install -y ntp
```

b Set the system time.

```
ntpd -gq
```

c Enable the `ntpd` daemon.

```
systemctl enable ntpd
```

d Configure `ntpd` to start when the system starts.

Currently, an unresolved issue associated with NTP prevents `ntpd` from restarting correctly after a reboot. The following commands provide a workaround to ensure that it does.

```
echo "systemctl start ntpd" >> /etc/rc.d/rc.local
chmod +x /etc/rc.d/rc.local
```

8 Install the *Nmap Ncat* utility.

The utility is used to verify ZooKeeper ensemble configurations. **Perform this step only on the two resource pool hosts that are designated for use in the ZooKeeper ensemble**.

```
yum install -y nmap-ncat
```

9 Install the Zenoss repository package.

a Install the package.

```
rpm -ivh http://get.zenoss.io/yum/zenoss-repo-1-1.x86_64.rpm
```

b Clean out the `yum` cache directory.

```
yum clean all
```

10 Reboot the host.

```
reboot
```

Creating a file system for Control Center internal services

This procedure creates an XFS file system on a primary partition.

Note **Perform this procedure only on the two resource pool hosts that are designated for use in the ZooKeeper ensemble**. No other resource pool hosts run Control Center internal services, so no other pool hosts need a partition for internal services data.

1. Log in to the target host as `root`, or as a user with superuser privileges.
2. Identify the target primary partition for the file system to create.

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

3. Create an XFS file system.
   Replace *Isvcs-Partition* with the path of the target primary partition:

```
mkfs -t xfs Isvcs-Partition
```

4. Create the mount point for Control Center internal services data.

```
mkdir -p /opt/serviced/var/isvcs
```

5. Add an entry to the `/etc/fstab` file.
   Replace *Isvcs-Partition* with the path of the primary partition used in the previous step:

```
echo "Isvcs-Partition \
  /opt/serviced/var/isvcs xfs defaults 0 0" >> /etc/fstab
```

6. Mount the file system, and then verify it mounted correctly.

```
mount -a && mount | grep isvcs
```

   Example result:

```
/dev/xvdb1 on /opt/serviced/var/isvcs type xfs
 (rw,relatime,seclabel,attr2,inode64,noquota)
```

Installing Docker and Control Center

This procedure installs and configures Docker, and installs Control Center.

1. Log in to the resource pool host as `root`, or as a user with superuser privileges.
2. Install Docker 1.9.0, and then disable accidental upgrades.
   a. Add the Docker repository to the host's repository list.

```
cat > /etc/yum.repos.d/docker.repo <<-EOF
[dockerrepo]
name=Docker Repository
baseurl=https://yum.dockerproject.org/repo/main/centos/7
enabled=1
gpgcheck=1
gpgkey=https://yum.dockerproject.org/gpg
EOF
```

   b. Install Docker 1.9.0.

```
yum clean all && yum makecache fast
yum install -y docker-engine-1.9.0
```

   c. Open `/etc/yum.repos.d/docker.repo` with a text editor.

d Change the value of the `enabled` key from `1` to `0`.

e Save the file and close the text editor.

3 Create a symbolic link for the Docker temporary directory.

Docker uses its temporary directory to spool images. The default directory is `/var/lib/docker/tmp`. The following command specifies the same directory that Control Center uses, `/tmp`. You can specify any directory that has a minimum of 10GB of unused space.

a Create the `docker` directory in `/var/lib`.

```
mkdir /var/lib/docker
```

b Create the link to `/tmp`.

```
ln -s /tmp /var/lib/docker/tmp
```

4 Create a `systemd` override file for the Docker service definition.

a Create the override directory.

```
mkdir -p /etc/systemd/system/docker.service.d
```

b Create the override file.

```
cat <<EOF > /etc/systemd/system/docker.service.d/docker.conf
[Service]
TimeoutSec=300
EnvironmentFile=-/etc/sysconfig/docker
ExecStart=
ExecStart=/usr/bin/docker daemon \$OPTIONS -H fd://
EOF
```

c Reload the `systemd` manager configuration.

```
systemctl daemon-reload
```

5 Install Control Center.

Control Center includes a utility that simplifies the process of creating a device mapper thin pool.

```
yum clean all && yum makecache fast
yum --enablerepo=zenoss-stable install -y serviced-1.1.6
```

6 Create a device mapper thin pool for Docker data.

a Identify the primary partition for the thin pool to create.

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

b Create the thin pool.

Replace *Path-To-Device* with the path of an unused primary partition:

```
serviced-storage create-thin-pool docker Path-To-Device
```

On success, the result includes the name of the thin pool, which always starts with `/dev/mapper`.

7 Configure and start the Docker service.

a Create variables for adding arguments to the Docker configuration file.

The `--exec-opt` argument is a workaround for *a Docker issue* on RHEL/CentOS 7.x systems.

Replace *Thin-Pool-Device* with the name of the thin pool device created in the previous step:

```
myDriver="-s devicemapper"
myFix="--exec-opt native.cgroupdriver=cgroupfs"
myFlag="--storage-opt dm.thinpooldev"
myPool="Thin-Pool-Device"
```

b Add the arguments to the Docker configuration file.

```
echo 'OPTIONS="'$myDriver $myFix $myFlag'='$myPool'"' \
  >> /etc/sysconfig/docker
```

c Start or restart Docker.

```
systemctl restart docker
```

The initial startup takes up to a minute, and may fail. If the startup fails, repeat the previous command.

8 Configure name resolution in containers.

Each time it starts, `docker` selects an IPv4 subnet for its virtual Ethernet bridge. The selection can change; this step ensures consistency.

a Identify the IPv4 subnet and netmask `docker` has selected for its virtual Ethernet bridge.

```
ip addr show docker0 | grep inet
```

b Open `/etc/sysconfig/docker` in a text editor.

c Add the following flags to the end of the *OPTIONS* declaration.

Replace *Bridge-Subnet* with the IPv4 subnet `docker` selected for its virtual bridge, and replace *Bridge-Netmask* with the netmask `docker` selected:

```
--dns=Bridge-Subnet --bip=Bridge-Subnet/Bridge-Netmask
```

For example, if the bridge subnet and netmask is 172.17.0.1/16, the flags to add are `--dns=172.17.0.1 --bip=172.17.0.1/16`.

Note Leave a blank space after the end of the thin pool device name, and make sure the double quote character (") is at the end of the line.

d Restart the Docker service.

```
systemctl restart docker
```

Configuring and starting Control Center

This procedure customizes key configuration variables of Control Center.

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

2 Configure Control Center as an agent of the master host.

The following variable configures `serviced` to serve as agent:

`SERVICED_AGENT`

Default: `0` (false)

Determines whether a `serviced` instance performs agent tasks. Agents run application services scheduled for the resource pool to which they belong. The `serviced` instance configured as the master

runs the scheduler. A `serviced` instance may be configured as agent and master, or just agent, or just master.

`SERVICED_MASTER`

Default: `0` (false)

Determines whether a `serviced` instance performs master tasks. The master runs the application services scheduler and other internal services, including the server for the Control Center browser interface. A `serviced` instance may be configured as agent and master, or just agent, or just master. Only one `serviced` instance in a Control Center cluster may be the master.

In addition, the following lines need to be edited, to replace `{{SERVICED_MASTER_IP}}` with the IP address of the master host:

```
# SERVICED_ZK={{SERVICED_MASTER_IP}}:2181
# SERVICED_DOCKER_REGISTRY={{SERVICED_MASTER_IP}}:5000
# SERVICED_ENDPOINT={{SERVICED_MASTER_IP}}:4979
# SERVICED_LOG_ADDRESS={{SERVICED_MASTER_IP}}:5042
# SERVICED_LOGSTASH_ES={{SERVICED_MASTER_IP}}:9100
# SERVICED_STATS_PORT={{SERVICED_MASTER_IP}}:8443
```

a Open `/etc/default/serviced` in a text editor.

b Find the *SERVICED_AGENT* declaration, and then change the value from `0` to `1`.
The following example shows the line to change:

```
# SERVICED_AGENT=0
```

c Remove the number sign character (#) from the beginning of the line.

d Find the *SERVICED_MASTER* declaration, and then remove the number sign character (#) from the beginning of the line.

e Globally replace `{{SERVICED_MASTER_IP}}` with the IP address of the master host.

Note Remove the number sign character (#) from the beginning of each variable declaration that includes the master IP address.

---

f Save the file, and then close the editor.

3 Optional: Specify an alternate private subnet for Control Center, if necessary.

The default private subnet may already be in use in your environment. The following variable configures `serviced` to use an alternate subnet:

`SERVICED_VIRTUAL_ADDRESS_SUBNET`

Default: `10.3`

The 16-bit private subnet to use for `serviced`'s virtual IPv4 addresses. RFC 1918 restricts private networks to the 10.0/24, 172.16/20, and 192.168/16 address spaces. However, `serviced` accepts any valid, 16-bit, IPv4 address space for its private network.

a Open `/etc/default/serviced` in a text editor.

b Locate the *SERVICED_VIRTUAL_ADDRESS_SUBNET* declaration, and then change the value.
The following example shows the line to change:

```
# SERVICED_VIRTUAL_ADDRESS_SUBNET=10.3
```

c Remove the number sign character (#) from the beginning of the line.

d Save the file, and then close the editor.

4 Start the Control Center service (`serviced`).

```
systemctl start serviced
```

To monitor progress, enter the following command:

```
journalctl -flu serviced -o cat
```

To install additional resource pool hosts, return to *Verifying candidate host resources* on page 19.

ZooKeeper ensemble configuration

Note If you are installing a single-host deployment, or if your deployment includes fewer than two resource pool hosts, skip this section.

Control Center relies on *Apache ZooKeeper* to coordinate its services. The procedures in this section create a ZooKeeper ensemble of 3 nodes. To perform these procedures, you need a Control Center master host and a minimum of two resource pool hosts. Each resource pool host requires a separate primary partition for Control Center internal services, and each should have a static IP address. For more information about storage requirements, refer to the *Zenoss Core Planning Guide*.

Note Zenoss strongly recommends configuring a ZooKeeper ensemble for all production deployments.

A ZooKeeper ensemble requires a minimum of 3 nodes, and 3 nodes is sufficient for most deployments. A 5-node configuration improves failover protection during maintenance windows. Ensembles larger than 5 nodes are not necessary. An odd number of nodes is recommended, and an even number of nodes is strongly discouraged.

Note The Control Center ZooKeeper service requires consistently fast storage. Ideally, the primary partition for Control Center internal services is on a separate, high-performance device that has only one primary partition.

Control Center variables for ZooKeeper

This tables in this section associates the ZooKeeper-related Control Center variables to set in `/etc/default/serviced` with the roles that hosts play in a Control Center cluster.

Table 1: Control Center master host

SERVICED_ISVCS_ZOOKEEPER_ID The unique identifier of a ZooKeeper ensemble node. Value: 1
SERVICED_ISVCS_ZOOKEEPER_QUORUM The ZooKeeper node ID, IP address, peer communications port, and leader communications port of each host in an ensemble. Each quorum definition must be unique, so the IP address of the "current" host is 0.0.0.0. Value: *`ZooKeeper-ID@IP-Address`*`:2888:3888,...`
SERVICED_ZK The list of endpoints in the Control Center ZooKeeper ensemble, separated by the comma character (`,`). Each endpoint includes the IP address of the ensemble node, and the port that Control Center uses to communicate with it. Value: *`IP-Address`*`:2181,...`

Table 2: Control Center resource pool host and ZooKeeper ensemble node

SERVICED_ISVCS_ZOOKEEPER_ID The unique identifier of a ZooKeeper ensemble node. Value: 2 or 3
SERVICED_ISVCS_ZOOKEEPER_QUORUM The ZooKeeper node ID, IP address, peer communications port, and leader communications port of each host in an ensemble. Each quorum definition must be unique, so the IP address of the "current" host is 0.0.0.0. Value: `ZooKeeper-ID@IP-Address:2888:3888,...`
SERVICED_ISVCS_START The list of Control Center internal services to start and run on hosts other than the master host. Value: `zookeeper`
SERVICED_ZK The list of endpoints in the Control Center ZooKeeper ensemble, separated by the comma character (`,`). Each endpoint includes the IP address of the ensemble node, and the port that Control Center uses to communicate with it. Value: `IP-Address:2181,...`

Table 3: Control Center resource pool host only

SERVICED_ZK The list of endpoints in the Control Center ZooKeeper ensemble, separated by the comma character (`,`). Each endpoint includes the IP address of the ensemble node, and the port that Control Center uses to communicate with it. Value: `IP-Address:2181,...`

Configuring the master host as a ZooKeeper node

This procedure configures the Control Center master host as a member of the ZooKeeper ensemble.

Note For accuracy, this procedure constructs Control Center configuration variables in the shell and appends them to `/etc/default/serviced`. The last step is to move the variables from the end of the file to more appropriate locations.

1. Log in to the master host as `root`, or as a user with superuser privileges.
2. Create a variable for each Control Center host to include in the ZooKeeper ensemble.

   The variables are used in subsequent steps.

   Note Define the variables identically on the master host and on each resource pool host.

   Replace *Master-Host-IP* with the IP address of the Control Center master host, and replace *Pool-Host-A-IP* and *Pool-Host-B-IP* with the IP addresses of the Control Center resource pool hosts to include in the ensemble:

   ```
   node1=Master-Host-IP
   node2=Pool-Host-A-IP
   node3=Pool-Host-B-IP
   ```

Note ZooKeeper requires IP addresses for ensemble configuration.

3 Set the ZooKeeper node ID to `1`.

```
echo "SERVICED_ISVCS_ZOOKEEPER_ID=1" >> /etc/default/serviced
```

4 Specify the nodes in the ZooKeeper ensemble.
You may copy the following text and paste it in your console:

```
echo "SERVICED_ZK=${node1}:2181,${node2}:2181,${node3}:2181" \
  >> /etc/default/serviced
```

5 Specify the nodes in the ZooKeeper quorum.

ZooKeeper requires a unique quorum definition for each node in its ensemble. To achieve this, replace the IP address of the current node with `0.0.0.0`.

You may copy the following of text and paste it in your console:

```
q1="1@0.0.0.0:2888:3888"
q2="2@${node2}:2888:3888"
q3="3@${node3}:2888:3888"
echo "SERVICED_ISVCS_ZOOKEEPER_QUORUM=${q1},${q2},${q3}" \
  >> /etc/default/serviced
```

6 Clean up the Control Center configuration file.

a Open `/etc/default/serviced` with a text editor.

b Navigate to the end of the file, and cut the line that contains the *SERVICED_ZK* variable declaration at that location.

The value of this declaration specifies 3 hosts.

c Locate the *SERVICED_ZK* variable near the beginning of the file, and then delete the line it is on.

The value of this declaration is just the master host.

d Paste the *SERVICED_ZK* variable declaration from the end of the file in the location of the just-deleted declaration.

e Navigate to the end of the file, and cut the line that contains the *SERVICED_ISVCS_ZOOKEEPER_ID* variable declaration at that location.

f Locate the *SERVICED_ISVCS_ZOOKEEPER_ID* variable near the end of the file, and then delete the line it is on.

This declaration is commented out.

g Paste the *SERVICED_ISVCS_ZOOKEEPER_ID* variable declaration from the end of the file in the location of the just-deleted declaration.

h Navigate to the end of the file, and cut the line that contains the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable declaration at that location.

i Locate the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable near the end of the file, and then delete the line it is on.

This declaration is commented out.

j Paste the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable declaration from the end of the file in the location of the just-deleted declaration.

k Save the file, and then close the text editor.

7 Verify the ZooKeeper environment variables.

```
egrep '^[^#]*SERVICED' /etc/default/serviced | egrep '(_ZOO|_ZK)'
```

Configuring a resource pool host as a ZooKeeper node

To perform this procedure, you need a resource pool host with an XFS file system on a separate partition, created previously.

This procedure configures a ZooKeeper ensemble on a resource pool host. Repeat this procedure on each Control Center resource pool host to add to the ZooKeeper ensemble.

1 Log in to the resource pool host as `root`, or as a user with superuser privileges.
2 Create a variable for each Control Center host to include in the ZooKeeper ensemble.

   The variables are used in subsequent steps.

   Note Define the variables identically on the master host and on each resource pool host.

   Replace *Master-Host-IP* with the IP address of the Control Center master host, and replace *Pool-Host-A-IP* and *Pool-Host-B-IP* with the IP addresses of the Control Center resource pool hosts to include in the ensemble:

```
node1=Master-Host-IP
node2=Pool-Host-A-IP
node3=Pool-Host-B-IP
```

   Note ZooKeeper requires IP addresses for ensemble configuration.

3 Set the ID of this node in the ZooKeeper ensemble.

   For *Pool-Host-A-IP* (**node2**), use the following command:

```
echo "SERVICED_ISVCS_ZOOKEEPER_ID=2" >> /etc/default/serviced
```

   For *Pool-Host-B-IP* (**node3**), use the following command:

```
echo "SERVICED_ISVCS_ZOOKEEPER_ID=3" >> /etc/default/serviced
```

4 Specify the nodes in the ZooKeeper ensemble.
   You may copy the following text and paste it in your console:

```
echo "SERVICED_ZK=${node1}:2181,${node2}:2181,${node3}:2181" \
  >> /etc/default/serviced
```

5 Specify the nodes in the ZooKeeper quorum.

   ZooKeeper requires a unique quorum definition for each node in its ensemble. To achieve this, replace the IP address of the current node with `0.0.0.0`.

   For *Pool-Host-A-IP* (**node2**), use the following commands:

```
q1="1@${node1}:2888:3888"
q2="2@0.0.0.0:2888:3888"
q3="3@${node3}:2888:3888"
echo "SERVICED_ISVCS_ZOOKEEPER_QUORUM=${q1},${q2},${q3}" \
  >> /etc/default/serviced
```

   For *Pool-Host-B-IP* (**node3**), use the following commands:

```
q1="1@${node1}:2888:3888"
q2="2@${node2}:2888:3888"
```

```
q3="3@0.0.0.0:2888:3888"
echo "SERVICED_ISVCS_ZOOKEEPER_QUORUM=${q1},${q2},${q3}" \
  >> /etc/default/serviced
```

6 Set the *SERVICED_ISVCS_START* variable, and clean up the Control Center configuration file.
   a Open `/etc/default/serviced` with a text editor.
   b Locate the *SERVICED_ISVCS_START* variable, and then delete all but `zookeeper` from its list of values.
   c Remove the number sign character (#) from the beginning of the line.
   d Navigate to the end of the file, and cut the line that contains the *SERVICED_ZK* variable declaration at that location.

      The value of this declaration specifies 3 hosts.
   e Locate the *SERVICED_ZK* variable near the beginning of the file, and then delete the line it is on.

      The value of this declaration is just the master host.
   f Paste the *SERVICED_ZK* variable declaration from the end of the file in the location of the just-deleted declaration.
   g Navigate to the end of the file, and cut the line that contains the *SERVICED_ISVCS_ZOOKEEPER_ID* variable declaration at that location.
   h Locate the *SERVICED_ISVCS_ZOOKEEPER_ID* variable near the end of the file, and then delete the line it is on.

      This declaration is commented out.
   i Paste the *SERVICED_ISVCS_ZOOKEEPER_ID* variable declaration from the end of the file in the location of the just-deleted declaration.
   j Navigate to the end of the file, and cut the line that contains the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable declaration at that location.
   k Locate the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable near the end of the file, and then delete the line it is on.

      This declaration is commented out.
   l Paste the *SERVICED_ISVCS_ZOOKEEPER_QUORUM* variable declaration from the end of the file in the location of the just-deleted declaration.
   m Save the file, and then close the text editor.
7 Verify the ZooKeeper environment variables.

```
egrep '^[^#]*SERVICED' /etc/default/serviced \
  | egrep '(_ZOO|_ZK|_STA)'
```

8 Pull the required Control Center ZooKeeper image from the master host.
   a Identify the image to pull.

      ```
      serviced version | grep IsvcsImages
      ```

      Example result:

      ```
      IsvcsImages: [zenoss/serviced-isvcs:v40 zenoss/isvcs-zookeeper:v3]
      ```
   b Pull the Control Center ZooKeeper image.

      Replace *Isvcs-ZK-Image* with the name and version number of the ZooKeeper image from the previous substep:

      ```
      docker pull Isvcs-ZK-Image
      ```

Starting a ZooKeeper ensemble

This procedure starts a ZooKeeper ensemble.

The window of time for starting a ZooKeeper ensemble is relatively short. The goal of this procedure is to restart Control Center on each ensemble node at about the same time, so that each node can participate in electing the leader.

1. Log in to the Control Center master host as `root`, or as a user with superuser privileges.
2. In a separate window, log in to the second node of the ZooKeeper ensemble (*Pool-Host-A-IP*).
3. In another separate window, log in to the third node of the ZooKeeper ensemble (*Pool-Host-B-IP*).
4. On all ensemble hosts, stop and start `serviced`.

```
systemctl stop serviced && systemctl start serviced
```

5. On the master host, check the status of the ZooKeeper ensemble.

```
{ echo stats; sleep 1; } | nc localhost 2181 | grep Mode
{ echo stats; sleep 1; } | nc Pool-Host-A-IP 2181  | grep Mode
{ echo stats; sleep 1; } | nc Pool-Host-B-IP 2181  | grep Mode
```

   If `nc` is not available, you can use `telnet` with *interactive ZooKeeper commands*.
6. Optional: Log in to the Control Center browser interface, and then start Zenoss Core and related applications, if desired.

   The next procedure requires stopping Zenoss Core.

Updating resource pool hosts

The default configuration of resource pool hosts sets the value of the *SERVICED_ZK* variable to the master host only. This procedure updates the setting to include the full ZooKeeper ensemble.

Perform this procedure on each resource pool host in your Control Center cluster.

1. Log in to the resource pool host as `root`, or as a user with superuser privileges.
2. Update the variable.
   a. Open `/etc/default/serviced` in a text editor.
   b. Locate the *SERVICED_ZK* declaration, and then replace its value with the same value used in the ZooKeeper ensemble nodes.
   c. Save the file, and then close the editor.
3. Restart Control Center.

```
systemctl restart serviced
```

Adding hosts to the default resource pool

Note If you are installing a single-host deployment, skip this section.

This procedure adds one or more resource pool hosts to the `default` resource pool.

1. Log in to the Control Center master host as `root`, or as a user with superuser privileges.
2. Add a resource pool host.

Replace *Hostname-Or-IP* with the hostname or IP address of the resource pool host to add:

```
serviced host add Hostname-Or-IP:4979 default
```

If you enter a hostname, all hosts in your Control Center cluster must be able to resolve the name, either through an entry in `/etc/hosts`, or through a nameserver on your network.

3 Repeat the preceding command for each resource pool host in your Control Center cluster.

Deploying Zenoss Core

This procedure adds the Zenoss Core application to the list of applications that Control Center manages, and pulls application images from Docker Hub.

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

2 Add the `Zenoss.core` application to Control Center.

```
myPath=/opt/serviced/templates
serviced template add $myPath/zenoss-core-*.json
```

On success, the `serviced` command returns the template identifier.

3 Deploy the application.

Replace *Template-ID* with the template identifier returned in the previous step, and replace *Deployment-ID* with a name for this deployment (for example, `Dev` or `Test`):

```
serviced template deploy Template-ID default Deployment-ID
```

Control Center pulls Zenoss Core images into the local registry. To monitor progress, enter the following command:

```
journalctl -flu serviced -o cat
```

Control Center and Zenoss Core are now installed, and Zenoss Core is ready to be configured for your environment. For more information, refer to the *Zenoss Core Configuration Guide*.