Installing Control Center

This page lists the procedures for installing multi-host and single-host deployments of Control Center. For planning information, see Planning a Resource Manager deployment.

Keep this page open, and open new tabs or windows for each procedure.

For optimum results, review the installation procedures before installing Control Center.

All deployments

1. Verify hosts
    a. Download the zenoss-installer script
    b. Verify candidate host resources
2. Download and stage required files
    a. Download files
    b. Install the repository mirror
    c. Stage Docker image files:
        - On the master host
        - On ZooKeeper ensemble nodes

Multi-host deployments

For production use, Zenoss recommends installing a multi-host deployment with a minimum of 3 hosts.

One: Install the master host

Every deployment requires one master host.

1. Install the master host:
    a. Prepare the host
    b. Install Docker and Control Center
    c. Configure Docker
    d. Load image files
    e. Create the application data thin pool
2. Configure the master host
    a. Set up user access
    b. Configure tenant data storage
    c. Set the host role to master
    d. Optional: Change the Docker registry endpoint
    e. Optional: Configure offline use
    f. Optional: Configure additional features
        - Master host configuration variables
        - Universal configuration variables
3. Start Control Center
4. Create resource pools
5. Add the host to a resource pool

Two: Install delegate hosts

Repeat this procedure for each delegate host.

1. Install a delegate host
    a. Prepare the host
    b. Install Docker and Control Center
    c. Configure NFS
    d. Configure Docker
2. Configure a delegate host
    a. Enable CLI access
    b. Set host role to delegate
    c. Set the Docker registry endpoint
    d. Set internal services endpoints
    e. Optional: Configure offline use
    f. Optional: Configure additional features
        - Delegate host configuration variables
        - Universal configuration variables
3. Start Control Center

4. Authenticate the host

Three: Configure the ZooKeeper ensemble

Use the following procedures to configure a ZooKeeper ensemble (cluster) for a multi-host Control Center deployment that includes a minimum of three hosts.

1. Understand ZooKeeper and Control Center
2. Understand the configuration process
    - ZooKeeper variables
    - Example multi-host ZooKeeper configuration
3. Configure the master host
4. Configure delegate host A
5. Configure delegate host B
6. Import the Docker image for ZooKeeper
7. Start the ensemble
8. Update and start other hosts

Single-host deployments

Single-host deployments are recommended for testing and development use only.

1. Install the master host:
    a. Prepare the host
    b. Install Docker and Control Center
    c. Configure Docker
    d. Load image files
    e. Create the application data thin pool
2. Configure the master host
    a. Set up user access
    b. Configure tenant data storage
    c. Set the host role to master
    d. Optional: Change the Docker registry endpoint
    e. Optional: Configure offline use
    f. Optional: Configure additional features
        - Master host configuration variables
        - Universal configuration variables
3. Start Control Center
4. Add the host to a resource pool

Verifying hosts

This section describes how to use the `zenoss-installer` script to verify hosts for their roles in a Control Center deployment. In addition, this section includes procedures for preparing required filesystems on the master host.

The `verify` action of the `zenoss-installer` script performs read-only tests of the compute, memory, operating system, and storage resources of a host. The `verify` action is intended for iterative use; run the script, correct an error, and then run the script again. You can perform the `verify` action as many times as you wish without affecting the host.

The `zenoss-installer` script is updated regularly. Please download the latest version before creating a new deployment of Control Center. The script is not needed on Zenoss Resource Manager or Zenoss Community Edition (Core) virtual appliances.

Downloading the zenoss-installer script

To perform this procedure, you need:

- A workstation with internet access.
- Zenoss Resource Manager users: Permission to download files from delivery.zenoss.com. Customers can request permission by filing a ticket at t he Zenoss Support site.
- Zenoss Community Edition (Core) users: An account on the Zenoss Community site.
- A secure network copy program.

Use this procedure to download the `zenoss-installer` script:

1. In a web browser, navigate to the download site, and then log in.
    - Zenoss Resource Manager users: delivery.zenoss.com
    - Zenoss Community Edition (Core) users: Zenoss Community
2. Download the zenoss-installer script.
3. Use a secure copy program to copy the script to Control Center candidate hosts.

Verifying candidate host resources

Use this procedure to run the `zenoss-installer` script on a candidate host.

1. Log in to the host as root, or as a user with superuser privileges.
2. Add execute permissions to the script.
   The following example assumes the script is located in /tmp; adjust the path, if necessary.
   ```
   chmod +x /tmp/zenoss-installer
   ```
3. Run the script with the arguments that match the role the host will play in your Control Center deployment.

Role	Deployment	Invocation
Master	Single-host	`zenoss-installer -a verify -d single -r master`
Master	Multi-host	`zenoss-installer -a verify -d multi -r master`
Delegate	Multi-host	`zenoss-installer -a verify -r delegate`
Collector	Multi-host	`zenoss-installer -a verify -r collector`

Hardware errors (VHW)

Error	Issue	Solution
VHW01	Control Center supports only the x86_64 processor architecture.	Select a different host.
VHW02	The number of available CPU cores does not meet the minimum required for the specified host role.	Increase the number of cores assigned to the host or select a different host.
VHW03	One or more CPU cores does not support the AES instruction set, which speeds encryption and decryption processing.	If the candidate host is a virtual machine, the managing hypervisor may be configured in Hyper-V compatibility mode. Check the setting and disable it or select a different host.
VHW04	The amount of available main memory does not support the specified host role. †	Increase the amount of memory assigned to the host or select a different host.

† Memory is measured in kibibytes (1024 per byte).

Software errors (VSW)

Error	Issue	Solution
VSW01	Control Center supports only the x86_64 kernel architecture.	Upgrade the operating system or select a different host.
VSW02	The installed kernel version is less than the required minimum version.	Upgrade the kernel or select a different host.
VSW03	The installed kernel patch is less than the required minimum patch.	Upgrade the kernel or select a different host.
VSW04	The installed operating system is not RHEL or CentOS.	Install a supported operating system or select a different host.
VSW05	The installed operating system release is not supported.	Upgrade the operating system or select a different host.

Network errors (VNW)

Error	Issue	Solution
VNW01	The hostname resolves to 127.0.0.1 only or does not resolve to a recognizable IPv4 address.	Add an entry for the host to the network nameserver, or to /etc/hosts.

VNW02	The /etc/hosts file does not include an entry for 127.0.0.1.	Add an entry to /etc/hosts that maps 127.0.0.1 to localhost.
VNW03	The /etc/hosts file does not include an entry that maps localhost to 127.0.0.1.	Add an entry to /etc/hosts that maps 127.0.0.1 to localhost.

Storage errors (VST)

Storage space is computed as kilobytes (1000 per byte), not kibibytes (1024 per byte).

Error	Issue	Solution
VST01	The amount of available swap space is less than the required minimum.	Add space to the swap device or partition.
VST02	Swap is not mounted on a block device or partition.	Add a separate device or partition for swap.
VST03	(Master hosts only) One or both of the following paths do not exist: • /opt/serviced/var/backups • /opt/serviced/var/isvcs	Perform one or both of the following procedures: • Creating a filesystem for application data backups • Creating a filesystem for Control Center internal services
VST04	One or more required mount points do not have the required minimum space. This test does not examine space for thin pools.	Add storage as recommended in Recommended storage layout.
VST05	One or more required mount points are mounted on the same device or partition.	Add devices or partitions for each required mount point.
VST06	The amount of space available on unused block storage devices is not enough for the thin pools the specified role requires.	On master hosts, separate devices or partitions are required for the thin pool for Docker data and the thin pool for application data. On delegate and collector hosts, a separate device is required for the Docker data thin pool. For more information, see Recommended storage layout. Best practice is to dedicate block devices to thin pools, so this test only examines top-level block devices. Partitions and logical volumes are not considered.

Recommended storage layout

The tests of available block storage are based on best practice recommendations for master hosts and delegate or collector hosts. Enter the following command to display information about available block storage:

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

The suggested minimum sizes for application data and application data backups should be replaced with sizes that meet your application requirements. To calculate the appropriate sizes for these storage areas, use the following guidelines:

- Application data storage includes space for both data and snapshots. The default base size for data is 100GB, and the recommended space for snapshots is 100% of the base size. Adding the two yields the suggested minimum size of 200GB.
- For application data backups, the recommended minimum space is 150% of the base size for data, or a minimum of 400GB, whichever is greater.

For large environments, the space for application data backups should be much greater. Individual backup files can be 50GB to 100GB each, or more.

Control Center master host storage

A Control Center master host should have a total of 7 separate block storage devices or partitions. The following table identifies the purpose and recommended minimum size of each.

	Purpose	Minimum size
1	Root (`/`)	30GB
2	Swap	16GB
3	Temporary (`/tmp`)	16GB
4	Docker data	50GB
5	Control Center internal services data (`/opt/serviced/var/isvcs`)	50GB
6	Application data	200GB
7	Application data backups (`/opt/serviced/var/backups`)	150GB

Control Center delegate host storage

A Control Center delegate or collector host should have a total of 4 separate block storage devices or partitions. The following table identifies the purpose and recommended minimum size of each.

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

Creating a filesystem for application data backups

This procedure requires one unused device or partition, or a remote file server that is compatible with XFS.
Use this procedure create an XFS filesystem on a device or partition, or to mount a remote filesystem, for application data backups.
If you are using a partition on a local device for backups, ensure that the storage 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 device or partition for the filesystem to create, if necessary.
   Skip this step if you are using a remote file server.
   ```
   lsblk -ap --output=NAME,SIZE,TYPE,FSTYPE,MOUNTPOINT
   ```
3. Optional: Create an XFS filesystem, if necessary.
   Skip this step if you are using a remote file server.
   Replace Storage with the path of the target device or partition:
   ```
   mkfs.xfs Storage
   ```
4. Create an entry in the /etc/fstab file.
   Replace File-System-Specification with one of the following values:
   - the path of the device or 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 filesystem, 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)
   ```

Creating a filesystem for Control Center internal services

This procedure requires one unused device or partition.
Use this procedure to create an XFS filesystem on an unused device or partition.

1. Log in to the target host as root, or as a user with superuser privileges.
2. Identify the target device or partition for the filesystem to create.

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

3. Create an XFS filesystem.
   Replace Storage with the path of the target device or partition:

   ```
   mkfs.xfs Storage
   ```

4. Enter the following command to add an entry to the /etc/fstab file.
   Replace Storage with the path of the device or partition used in the previous step:

   ```
   echo "Storage /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 filesystem, and then verify it mounted correctly.

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

   Example result:

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

Downloading and staging required files

This section describes how to download and install or stage Control Center software and its operating system dependencies. The procedures in this section are required to perform an installation.

The following table identifies where to perform each procedure in this section.

Procedure	Where to perform
Downloading Control Center files	A workstation with internet access
Installing the repository mirror	All Control Center hosts
Staging Docker image files on the master host	The Control Center master host
Staging a Docker image file on ZooKeeper ensemble nodes	Delegate hosts that are ZooKeeper ensemble nodes

Downloading Control Center files

To perform this procedure, you need:

- A workstation with internet access.
- Zenoss Resource Manager users: Permission to download files from delivery.zenoss.com. Customers can request permission by filing a ticket at the Zenoss Support site.
- Zenoss Community Edition (Core) users: An account on the Zenoss Community site.
- A secure network copy program.

Use this procedure to

- download the required files to a workstation
- copy the files to the hosts that need them

Perform these steps:

1. In a web browser, navigate to the download site, and then log in.
   - Zenoss Resource Manager users: delivery.zenoss.com
   - Zenoss Community Edition (Core) users: Zenoss Community
2. Download the self-installing Docker image files.
   - `install-zenoss-serviced-isvcs-vLATEST.run`
   - `install-zenoss-isvcs-zookeeper-vLATEST.run`
3. Download the Control Center RPM file.
   - `serviced-VERSION-1.x86_64.rpm`
4. Identify the operating system release on Control Center hosts.
   Enter the following command on each Control Center host in your deployment, if necessary. All Control Center hosts should be running the same operating system release and kernel.

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

5. Download the RHEL/CentOS repository mirror file for your deployment.
   The download site provides a repository mirror file for each tested release of RHEL/CentOS. Each mirror file contains the release-specific packages that Control Center requires.
   - `yum-mirror-centos7.2-1511-serviced-1.6.5.x86_64.rpm`
   - `yum-mirror-centos7.3-1611-serviced-1.6.5.x86_64.rpm`
   - `yum-mirror-centos7.4-1708-serviced-1.6.5.x86_64.rpm`
   - `yum-mirror-centos7.5-1804-serviced-1.6.5.x86_64.rpm`
   - `yum-mirror-centos7.6-1810-serviced-1.6.5.x86_64.rpm`
6. Optional: Download the Zenoss GNU Privacy Guard (GPG) key, if desired.
   You can use the Zenoss GPG key to verify Zenoss RPM files and the yum metadata of the repository mirror.
   a. Download the key.

      ```
      curl --location -o /tmp/tmp.html 'https://pgpkeys.hu/pks/lookup?op=get&search=0xED0A5FD2AA5A1AD7'
      ```

   b. Determine whether the download succeeded.

      ```
      grep -Ec '^\-\-\-\-\-BEGIN PGP' /tmp/tmp.html
      ```

      - If the result is 0, return to the previous substep.
      - If the result is 1, proceed to the next substep.

   c. Extract the key.

      ```
      awk '/^-----BEGIN PGP.*$/,/^-----END PGP.*$/' /tmp/tmp.html > ./RPM-GPG-KEY-Zenoss
      ```

7. Use a secure copy program to copy the files to Control Center hosts.
   - Copy all files to the master host.
   - Copy the RHEL/CentOS RPM file, the Control Center RPM file, and the Zenoss GPG key file to all delegate hosts.
   - Copy the Docker image file for ZooKeeper to delegate hosts that are ZooKeeper ensemble nodes.

Installing the repository mirror

Use this procedure to install the Zenoss repository mirror on a Control Center host. The mirror contains packages that are required on all Control Center hosts. Repeat this procedure on each host in your deployment.

1. Log in to the target host as root, or as a user with superuser privileges.
2. Move the RPM files and the Zenoss GPG key file to /tmp.
3. Upgrades only: Remove the existing repository mirror, if necessary.
    a. Search for the mirror.
    ```
    yum list --disablerepo=* | awk '/^yum-mirror/ { print $1}'
    ```
    b. Remove the mirror.
    Replace Old-Mirror with the name of the Zenoss repository mirror returned in the previous substep:
    ```
    yum remove Old-Mirror
    ```
4. Install the repository mirror.
    ```
    yum install /tmp/yum-mirror-*.rpm
    ```
    The yum command copies the contents of the RPM file to /opt/zenoss-repo-mirror.
5. Optional**:** Install the Zenoss GPG key, and then test the package files, if desired.
    a. Move the Zenoss GPG key to the mirror directory.
    ```
    mv /tmp/RPM-GPG-KEY-Zenoss /opt/zenoss-repo-mirror
    ```
    b. Install the key.
    ```
    rpm --import /opt/zenoss-repo-mirror/RPM-GPG-KEY-Zenoss
    ```
    c. Test the repository mirror package file.
    ```
    rpm -K /tmp/yum-mirror-*.rpm
    ```
    On success, the result includes the file name and the following information:
    ```
    (sha1) dsa sha1 md5 gpg OK
    ```
    d. Test the Control Center package file.
    ```
    rpm -K /tmp/serviced-VERSION-1.x86_64.rpm
    ```
6. Optional**:** Update the configuration file of the Zenoss repository mirror to enable GPG key verification, if desired.
    a. Open the repository mirror configuration file (/etc/yum.repos.d/zenoss-mirror.repo) with a text editor, and then add the following lines to the end of the file.
    ```
    repo_gpgcheck=1
    gpgkey=file:///opt/zenoss-repo-mirror/RPM-GPG-KEY-Zenoss
    ```
    b. Save the file, and then close the editor.
    c. Update the yum metadata cache.
    ```
    yum makecache fast
    ```
    The cache update process includes the following prompt:
    ```
    Retrieving key from file:///opt/zenoss-repo-mirror/RPM-GPG-KEY-Zenoss
    Importing GPG key 0xAA5A1AD7:
     Userid     : "Zenoss, Inc. <dev@zenoss.com>"
     Fingerprint: f31f fd84 6a23 b3d5 981d a728 ed0a 5fd2 aa5a 1ad7
     From       : /opt/zenoss-repo-mirror/RPM-GPG-KEY-Zenoss
    Is this ok [y/N]:
    ```
    Enter y.

7. Move the Control Center package file to the mirror directory.

```
mv /tmp/serviced-VERSION-1.x86_64.rpm /opt/zenoss-repo-mirror
```

8. Optional**:** Delete the mirror package file, if desired.

```
rm /tmp/yum-mirror-*.rpm
```

Staging Docker image files on the master host

Before performing this procedure, verify that approximately 640MB of temporary space is available on the file system where /root is located.

Use this procedure to copy Docker image files to the Control Center master host. The files are used when Docker is fully configured.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Copy or move the archive files to /root.
3. Add execute permission to the files.

```
chmod +x /root/*.run
```

Staging a Docker image file on ZooKeeper ensemble nodes

Before performing this procedure, verify that approximately 170MB of temporary space is available on the file system where /root is located.

Use this procedure to add a Docker image file to the Control Center delegate hosts that are ZooKeeper ensemble nodes. Delegate hosts that are not ZooKeeper ensemble nodes do not need the file.

1. Log in to a delegate host as root, or as a user with superuser privileges.
2. Copy or move the install-zenoss-isvcs-zookeeper-v11.run file to /root.
3. Add execute permission to the file.

```
chmod +x /root/*.run
```

Installing a master host

This section describes how to install Control Center on a Red Hat Enterprise Linux (RHEL) or CentOS host. The candidate host must have the CPU, RAM, and storage resources required to serve as the Control Center master host.

For more information about master host requirements, see Planning a Resource Manager deployment.

Preparing the master host operating system

Perform the steps in Downloading and staging required files, before performing this procedure.
Use this procedure to prepare a RHEL/CentOS 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. Ensure the host has a persistent numeric ID.
   Skip this step if you are installing a single-host deployment.
   Each Control Center host must have a unique host ID, and the ID must be persistent (not change when the host reboots).

   ```
   test -f /etc/hostid || genhostid ; hostid
   ```

   Record the ID for comparison with other Control Center hosts.
3. Disable the firewall, if necessary.
   This step is required for installation but not for deployment. For more information, see Planning a Resource Manager deployment.
   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 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. Note: The following commands are safe when performed during an installation, before Docker or Control Center are installed or running. To enable persistent log files after installation, stop Control Center, stop Docker, and then enter the following commands.

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

5. Enable and start the Dnsmasq package.
   The package facilitates networking among Docker containers.

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

   Most deployments do not need specific configuration for dnsmasq, however if name resolution in your environment relies solely on entries in /etc /hosts, configure dsnmasq so that containers can use the file:
   a. Open /etc/dnsmasq.conf with a text editor.
   b. Add the following lines to the file:

      ```
      domain-needed
      bogus-priv
      local=/local/
      domain=local
      interface=docker0
      ```

   c. Save the file, and then close the text editor.
   d. Restart the dnsmasq service.

      ```
      systemctl restart dnsmasq
      ```

6. Install and configure the NTP package.
   Note: This procedure assumes the host has internet access. To install and configure NTP on a host that does not have internet access, see Configuring a private master NTP server.
   a. Stop and disable chronyd, if present.

      ```
      test "$(systemctl is-active chronyd)" = "active" \
        && systemctl stop chronyd && systemctl disable chronyd
      ```

b. Install the NTP package.

```
yum makecache fast && yum install ntp
```

c. Set the system time.

```
ntpd -gq
```

d. Enable and start the ntpd service.

```
systemctl enable ntpd && systemctl start ntpd
```

Installing Docker and Control Center

Perform the steps in Downloading and staging required files before performing the procedures on this page.

Install Docker and Control Center

1. Log in to the host as root, or as a user with superuser privileges.
2. Install Docker CE 18.09.6 from the local repository mirror.
    a. Install Docker CE.

    ```
    yum install --enablerepo=zenoss-mirror docker-ce-18.09.6-3.el7
    ```

    If yum returns an error due to dependency issues, see Resolving package dependency conflicts for potential resolutions.
    b. Enable automatic startup.

    ```
    systemctl enable docker
    ```

3. Install Control Center from the local repository mirror.
    a. Install Control Center.

    ```
    yum install --enablerepo=zenoss-mirror /opt/zenoss-repo-mirror/serviced-VERSION-1.x86_64.rpm
    ```

    If yum returns an error due to dependency issues, see Resolving package dependency conflicts for potential resolutions.
    b. Enable automatic startup.

    ```
    systemctl enable serviced
    ```

4. Make a backup copy of the Control Center configuration file.
    a. Make a copy of /etc/default/serviced.

    ```
    cp /etc/default/serviced /etc/default/serviced-VERSION-orig
    ```

    b. Set the backup file permissions to read-only.

    ```
    chmod 0440 /etc/default/serviced-VERSION-orig
    ```

Remove unused maintenance scripts

For more information about maintenance scripts, see Control Center maintenance scripts.

Master host (multi-host deployment only)

```
rm -f /etc/cron.d/cron_zenossdbpack
```

Resource Manager pool host

Enter this command on **all but one** host in the pool:

```
rm -f /etc/cron.hourly/serviced \
  /etc/cron.weekly/serviced-fstrim
\
  /etc/cron.d/cron_zenossdbpack
```

Enter the following command **on one** host in the pool:

```
rm -f /etc/cron.hourly/serviced \
  /etc/cron.weekly/serviced-fstrim
```

Collector pool host

```
rm -f /etc/cron.hourly/serviced \
  /etc/cron.weekly/serviced-fstrim
\
  /etc/cron.d/cron_zenossdbpack
```

Configuring Docker on a master host

Use this procedure to configure Docker.

1. Log in to the master host as root, or as a user with superuser privileges.
2. 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
   ```
3. Create a systemd drop-in file for Docker.
   a. Create the override directory.
   ```
   mkdir -p /etc/systemd/system/docker.service.d
   ```
   b. Create the unit drop-in file.
   ```
   cat <<EOF > /etc/systemd/system/docker.service.d/docker.conf
   [Service]
   TimeoutSec=300
   EnvironmentFile=-/etc/sysconfig/docker
   ExecStart=
   ExecStart=/usr/bin/dockerd \$OPTIONS
   TasksMax=infinity
   EOF
   ```
   c. Reload the systemd manager configuration.
   ```
   systemctl daemon-reload
   ```
4. Create an LVM thin pool for Docker data.
   For more information about the serviced-storage command, see serviced-storage.
   To use an entire block device or partition for the thin pool, replace Device-Path with the device path:
   ```
   serviced-storage create-thin-pool docker Device-Path
   ```
   On success, the result is the device mapper name of the thin pool, which always starts with /dev/mapper.
5. Configure and start the Docker service.
   a. Create a variable for the name of the Docker thin pool.
   Replace Thin-Pool-Device with the name of the thin pool device created in the previous step:
   ```
   myPool="Thin-Pool-Device"
   ```
   b. 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.
   ```
   myDriver="--storage-driver devicemapper"
   myLog="--log-level=error"
   myFix="--exec-opt native.cgroupdriver=cgroupfs"
   myMount="--storage-opt dm.mountopt=discard"
   myFlag="--storage-opt dm.thinpooldev=$myPool"
   ```
   c. Add the arguments to the Docker configuration file.
   ```
   echo 'OPTIONS="'$myLog $myDriver $myFix $myMount $myFlag'"' \
     >> /etc/sysconfig/docker
   ```
   d. Start or restart Docker.

```
systemctl restart docker
```

The startup may take up to a minute, and may fail. If startup fails, repeat the restart command.

6. 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 address 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 Virtual-Bridge with the IPv4 address docker selected for its virtual bridge, followed by a slash (*/*) the subnet mask:

```
--dns=Virtual-Bridge --bip=Virtual-Bridge/16
```

      For example, if the bridge address is 172.17.0.1, add the following flags:

```
--dns=172.17.0.1 --bip=172.17.0.1/16
```

      Note: Use a space character ( ) to separate flags, and make sure the double quote character (") delimits the declaration of OPTIONS.
   d. Save the file, and then close the editor.
   e. Restart the Docker service.

```
systemctl restart docker
```

Loading image files

Use this procedure to load images into the local Docker registry on a host.

1. Log in to the host as root, or as a user with superuser privileges.
2. Change directory to /root.

   ```
   cd /root
   ```

3. Load the images.

   ```
   for image in install-zenoss-*.run
   do
     /bin/echo -en "\nLoading $image..."
     yes | ./$image
   done
   ```

4. List the images in the registry.

   ```
   docker images
   ```

   The result should show one image for each archive file.
5. Optional: Delete the archive files, if desired.

   ```
   rm -i ./install-zenoss-*.run
   ```

Creating the application data thin pool

Use this procedure to create a thin pool for application data storage.
This procedure does not include a specific value for the size of the thin pool. For more information about sizing this resource, see Master host storage areas. Or, use the suggested minimum value, 200GB. You can add storage to an LVM thin pool at any time.
Perform these steps:

1. Log in to the master host as root, or as a user with superuser privileges.
2. Create an LVM thin pool for application data.
   For more information, see serviced-storage.
   To use an entire block device or partition for the thin pool, replace Device-Path with the device path:

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

   On success, the result is the device mapper name of the thin pool, which always starts with /dev/mapper. Record the name for use in the next step.
3. Edit storage variables in the Control Center configuration file.
   a. Open /etc/default/serviced in a text editor.
   b. Locate the line for the SERVICED_FS_TYPE variable, and then make a copy of the line, immediately below the original.
   c. Remove the number sign character (#) from the beginning of the line.
   d. Locate the line for the SERVICED_DM_THINPOOLDEV variable, and then make a copy of the line, immediately below the original.
   e. Remove the number sign character (#) from the beginning of the line.
   f. Set the value to the device mapper name of the thin pool for application data.
   g. Save the file, and then close the editor.

Proceed to the next section and configure the host.

Configuring and starting the master host

This section includes the procedures for configuring Control Center on the master host, describes the configuration options that apply to the master host, and includes steps for starting the master host for the first time. Before installing Control Center on delegate hosts, perform the procedures in this section.

Many configuration choices depend on application requirements. Before configuring Control Center on the master host, review your application documentation.

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. In addition, users must be members of the Control Center browser interface access group, which by default is the system group, `wheel`. To enhance security, you may change the browser interface access group from `wheel` to any other group.

To use the Control Center command-line interface (CLI) on a Control Center host, a user must have login account on the host, and the account must be a member of the `serviced` group. The `serviced` group is created when the Control Center (`serviced`) RPM package is installed.

You can use two different groups to control access to the browser interface and the CLI. You can enable access to both interfaces for the same users by choosing the `serviced` group as the browser interface access group.

Pluggable Authentication Modules (PAM) has been tested and is recommended for enabling access to both the browser interface and the command-line interface. However, the PAM configuration must include the `sudo` service. Control Center relies on the host's `sudo` configuration, and if no configuration is present, PAM defaults to the configuration for `other`, which is typically too restrictive for Control Center users. For more information about configuring PAM, refer to your operating system documentation.

Adding users to the default browser interface access group

Use this procedure to add users to the default browser interface access group of Control Center, wheel.

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 a user to the wheel group.
   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.

Configuring a regular group as the Control Center browser interface access group

Use this procedure to change the default browser interface access group of Control Center from wheel to a non-system group.
Perform this procedure or the previous procedure, but not both.

1. Log in to the 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 group is ccuser. You may choose a different group, or choose the serviced group. (Choosing the serviced group allows all browser interface users to use the CLI.)

   ```
   myGROUP=ccuser
   ```

3. Create a new group, if necessary.

   ```
   groupadd $myGROUP
   ```

4. Add one or more existing users to the group.
   Replace User with the name of a login account on the host:

   ```
   usermod -aG $myGROUP User
   ```

   Repeat the preceding command for each user to add.
5. Specify the new administrative group in the serviced configuration file.
   a. Open /etc/default/serviced in a text editor.
   b. Locate the line for the SERVICED_ADMIN_GROUP variable, and then make a copy of the line, immediately below the original.
   c. Remove the number sign character (#) from the beginning of the line.
   d. Change the value from wheel to the name of the group you chose earlier.
   e. Save the file, and then close the editor.
6. Optional: Prevent the root user from gaining access to the Control Center browser interface, if desired.
   a. Open /etc/default/serviced in a text editor.
   b. Locate the line for the SERVICED_ALLOW_ROOT_LOGIN variable, and then make a copy of the line, immediately below the original.
   c. Remove the number sign character (#) from the beginning of the line.
   d. Change the value from 1 to 0.
   e. Save the file, and then close the editor.

Enabling use of the command-line interface

Use this procedure to enable a user to perform administrative tasks with the Control Center command-line interface.

1. Log in to the host as root, or as a user with superuser privileges.
2. Add a user to the serviced group.
   Replace User with the name of a login account on the host.

   ```
   usermod -aG serviced User
   ```

   Repeat the preceding command for each user to add.

Configuring the base size device for tenant data storage

Use this procedure to configure the base size of virtual storage devices for tenants in the application data thin pool. The base size is used each time a tenant device is created. In particular, the first time serviced starts, it creates the base size device and then creates a tenant device from the base size device.

Perform these steps:

1. Log in to the master host as root, or as a user with superuser privileges.
2. Identify the size of the thin pool for application data.
   The size is required to set an accurate value for the SERVICED_DM_BASESIZE variable.
   ```
   lvs --options=lv_name,lv_size | grep serviced-pool
   ```
3. Edit storage variables in the Control Center configuration file.
   a. Open /etc/default/serviced in a text editor.
   b. Locate the line for the SERVICED_DM_BASESIZE variable, and then make a copy of the line, immediately below the original.
   c. Remove the number sign character (#) from the beginning of the line.
   d. Change the value, if necessary.
      Replace Fifty-Percent with the value that is less than or equal to 50% of the size of the thin pool for application data. Include the symbol for gigabytes, G:
      ```
      SERVICED_DM_BASESIZE=Fifty-PercentG
      ```
   e. Save the file, and then close the editor.
4. Verify the settings in the serviced configuration file.
   ```
   grep -E '^\b*[A-Z_]+' /etc/default/serviced
   ```

Setting the host role to master

Use this procedure to configure a host as the master host.
Perform these steps:

1. Log in to the host as root, or as a user with superuser privileges.
2. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. Locate the line for the SERVICED_MASTER variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Save the file, and then close the editor.
3. Verify the settings in the serviced configuration file.

```
grep -E '^\b*[A-Z_]+' /etc/default/serviced
```

Optional: Changing the local Docker registry endpoint

Use this procedure to configure the master host with the endpoint of an alternative local Docker registry. Control Center includes a local Docker registry, but you may use an existing registry in your environment, if desired. For more information about configuring a local Docker registry, please refer to Docker documentation. Note: Changing the local Docker registry endpoint is rare. Perform this procedure only if you are sure it is necessary and the alternative local Docker registry is already available in your environment.

Perform these steps:

1. Log in to the master host as root, or as a user with superuser privileges.
2. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. Locate the line for the SERVICED_DOCKER_REGISTRY variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Replace localhost:5000 with the endpoint of the local Docker registry.
    Use the IP address or fully-qualified domain name of the host and the port number.
    e. Save the file, and then close the editor.
3. Verify the settings in the serviced configuration file.

    ```
    grep -E '^\b*[A-Z_]+' /etc/default/serviced
    ```

4. Add the insecure registry flag to the Docker configuration file.
    a. Open /etc/sysconfig/docker in a text editor.
    b. Add the following flag to the end of the OPTIONS declaration.
    Replace Registry-Endpoint with the same value used for the SERVICED_DOCKER_REGISTRY variable:

    ```
    --insecure-registry=Registry-Endpoint
    ```

    Note: Use a space character ( ) to separate flags, and make sure the double quote character (") delimits the declaration of OPTIONS.
    c. Save the file, and then close the editor.
5. Restart the Docker service.

    ```
    systemctl restart docker
    ```

Optional: Configuring offline use

Use this procedure to configure a host to operate without internet access.
Perform these steps:

1. Log in to the host as root, or as a user with superuser privileges.
2. Identify the IPv4 address of the host.

```
hostname -i
```

3. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. Locate the line for the SERVICED_OUTBOUND_IP variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Change the value to the IPv4 address identified in the previous step.
    e. Save the file, and then close the editor.
4. Verify the settings in the serviced configuration file.

```
grep -E '^\b*[A-Z_]+' /etc/default/serviced
```

Starting Control Center for the first time

Use this procedure to start the Control Center service (`serviced`) on a master host after installing and configuring it. This procedure is valid for single-host and multi-host deployments, whether the deployment has internet access or not, and is only performed once.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Verify the settings in the serviced configuration file.

   ```
   grep -E '^\b*[A-Z_]+' /etc/default/serviced
   ```

3. Start the Control Center service.

   ```
   systemctl start serviced
   ```

   To monitor progress, enter the following command:

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

   The `serviced` daemon tags images in the local Docker registry and starts its internal services. The Control Center browser and command-line interfaces are unavailable for about 3 minutes.

   When the message Host Master successfully started is displayed, Control Center is ready for the next procedure.

Until the master host is added to a pool, all `serviced` CLI commands that use the RPC server must be run as root.

Adding the master host to a resource pool

Use this procedure to add the master host to the `default` resource pool.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Add the master host to the pool.Replace Hostname-Or-IP with the hostname or IP address of the Control Center master host:

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

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

Creating resource pools for multi-host deployments

Use this procedure to create resource pools for a multi-host deployment.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Create a resource pool for Resource Manager.
    a. Create the pool.
    You can use any name for the pool; this example uses `resmgr`:

    ```
    serviced pool add resmgr
    ```

    b. Assign administrative and distributed file system (DFS) permissions to the new resource pool.

    ```
    serviced pool set-permission --admin --dfs resmgr
    ```

3. Optional: Create a resource pool for a collector.
    You can use any name for the pool; typically, the name reflects the physical location of the resources the pool will monitor:

    ```
    serviced pool add LOCATION-NAME
    ```

    Collector pools **do not** get administrative or DFS permissions.
4. Repeat step 3 for each collector in your environment.

Installing delegate hosts

This section describes how to install Control Center on a Red Hat Enterprise Linux (RHEL) or CentOS host. The candidate host must have the CPU, RAM, and storage resources required to serve as a Control Center delegate host.

For more information about delegate host requirements, see Planning a Resource Manager deployment.

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

Repeat the procedures in this section on each host to add to your deployment.

Preparing a delegate host

Perform the steps in Downloading and staging required files, before performing this procedure.
Use this procedure to prepare a RHEL/CentOS host as a Control Center delegate host.

1. Log in to the candidate delegate host as root, or as a user with superuser privileges.
2. Ensure the host has a persistent numeric ID.
   Each Control Center host must have a unique host ID, and the ID must be persistent (not change when the host reboots).

   ```
   test -f /etc/hostid || genhostid ; hostid
   ```

   Record the ID for comparison with other Control Center hosts.
3. Disable the firewall, if necessary.
   This step is required for installation but not for deployment. For more information, refer to the Resource Manager Planning Guide or the Zenoss Community Edition (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 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. Note: The following commands are safe when performed during an installation, before Docker or Control Center are installed or running. To enable persistent log files after installation, stop Control Center, stop Docker, and then enter the following commands.

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

5. Enable and start the Dnsmasq package.
   The package facilitates networking among Docker containers.

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

   Most deployments do not need specific configuration for dnsmasq, however if name resolution in your environment relies solely on entries in /etc /hosts, configure dsnmasq so that containers can use the file:
   a. Open /etc/dnsmasq.conf with a text editor.
   b. Add the following lines to the file:

      ```
      domain-needed
      bogus-priv
      local=/local/
      domain=local
      interface=docker0
      ```

   c. Save the file, and then close the text editor.
   d. Restart the dnsmasq service.

      ```
      systemctl restart dnsmasq
      ```

6. Install and configure the NTP package.
   Note: This procedure assumes the host has internet access. To install and configure NTP on a host that does not have internet access, see Configuring a private master NTP server.
   a. Stop and disable chronyd, if present.

      ```
      test "$(systemctl is-active chronyd)" = "active" && systemctl stop chronyd && systemctl disable
      chronyd
      ```

b. Install the NTP package.

```
yum makecache fast && yum install ntp
```

c. Set the system time.

```
ntpd -gq
```

d. Enable and start the ntpd service.

```
systemctl enable ntpd && systemctl start ntpd
```

Installing required software

Perform the procedure described in the following topic: Installing Docker and Control Center

Configuring NFS 4.0

Use this procedure to configure NFS 4.0 on delegate hosts if the operating system release is 7.4 or a more recent release. There may be a file locking defect in NFS 4.1 with RHEL/CentOS 7.4.

1. Log in to the host as root, or as a user with superuser privileges.
2. Determine which release is installed.

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

   - If the result is 7.4 or greater, perform the remaining steps of this procedure.
   - If the result includes 7.2 or 7.3, continue to the next procedure.
3. Change the NFS configuration file.
   a. Open /etc/nfsmount.conf with a text editor.
   b. Locate the Defaultvers directive.
   c. Remove the number sign character (#) from the beginning of the line.
   d. Change the value from 4 to 4.0.
      The line should appear as follows:

      ```
      Defaultvers=4.0
      ```

   e. Save the file, and then close the editor.
4. Restart the NFS server.

   ```
   systemctl restart nfs-server
   ```

Configuring Docker on delegate hosts

Use this procedure to configure Docker.

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. 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
   ```
3. Create a systemd drop-in file for Docker.
   a. Create the override directory.
   ```
   mkdir -p /etc/systemd/system/docker.service.d
   ```
   b. Create the unit drop-in file.
   ```
   cat <<EOF > /etc/systemd/system/docker.service.d/docker.conf
   [Service]
   TimeoutSec=300
   EnvironmentFile=-/etc/sysconfig/docker
   ExecStart=
   ExecStart=/usr/bin/dockerd \$OPTIONS
   TasksMax=infinity
   EOF
   ```
   c. Reload the systemd manager configuration.
   ```
   systemctl daemon-reload
   ```
4. Create an LVM thin pool for Docker data.
   For more information about the serviced-storage command, see serviced-storage.
   To use an entire block device or partition for the thin pool, replace Device-Path with the device path:
   ```
   serviced-storage create-thin-pool docker Device-Path
   ```
   On success, the result is the device mapper name of the thin pool, which always starts with /dev/mapper.
5. Configure and start the Docker service.
   a. Create a variable for the name of the Docker thin pool.
   Replace Thin-Pool-Device with the name of the thin pool device created in the previous step:
   ```
   myPool="Thin-Pool-Device"
   ```
   b. 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.
   ```
   myDriver="--storage-driver devicemapper"
   myLog="--log-level=error"
   myFix="--exec-opt native.cgroupdriver=cgroupfs"
   myMount="--storage-opt dm.mountopt=discard"
   myFlag="--storage-opt dm.thinpooldev=$myPool"
   ```
   c. Add the arguments to the Docker configuration file.
   ```
   echo 'OPTIONS="'$myLog $myDriver $myFix $myMount $myFlag'"' >> /etc/sysconfig/docker
   ```
   d. Start or restart Docker.
   ```
   systemctl restart docker
   ```

The startup may take up to a minute, and may fail. If startup fails, repeat the restart command.

6. 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 address and netmask docker has selected for its virtual 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 Virtual-Bridge with the IPv4 address docker selected for its virtual bridge, followed by a slash (*/*) the subnet mask:

      ```
      --dns=Virtual-Bridge --bip=Virtual-Bridge/16
      ```

      For example, if the bridge address is 172.17.0.1, add the following flags:

      ```
      --dns=172.17.0.1 --bip=172.17.0.1/16
      ```

      Note: Use a space character ( ) to separate flags, and make sure the double quote character (") delimits the declaration of OPTIONS.
   d. Save the file, and then close the editor.
   e. Restart the Docker service.

      ```
      systemctl restart docker
      ```

Proceed to the next section and configure the host.

Configuring and starting delegate hosts

This section provides the procedures that are required to configure a delegate host, describes the configuration options that apply to delegate hosts, and includes steps for starting a delegate for the first time. Perform the procedures in this section on each Control Center delegate host.

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

Additional configuration procedures

The following procedures are important for delegate hosts:

- Enabling use of the command-line interface
- Optional: Configuring offline use

Setting the host role to delegate

Use this procedure to configure a host as a delegate host.
Perform these steps:

1. Log in to the host as root, or as a user with superuser privileges.
2. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. Locate the line for the SERVICED_MASTER variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Change the value from 1 to 0.
    e. Save the file, and then close the editor.
3. Verify the settings in the serviced configuration file.

```
grep -E '^\b*[A-Z_]+' /etc/default/serviced
```

Setting the Docker registry endpoint

Use this procedure to configure the delegate host with the endpoint of the Docker registry for Control Center. Unless the master host is configured with an alternative local Docker registry, which is rare, the endpoint is the master host's hostname or IP address and port 5000.
Perform these steps:

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. Locate the line for the SERVICED_DOCKER_REGISTRY variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Replace localhost:5000 with the endpoint of the local Docker registry.
    If the master host is configured with an alternative local Docker registry, use the same endpoint here. Otherwise, just replace localhost with the IP address or fully-qualified domain name of the Control Center master host.
    e. Save the file, and then close the editor.
3. Verify the settings in the serviced configuration file.

```
grep -E '^\b*[A-Z_]+' /etc/default/serviced
```

4. Add the insecure registry flag to the Docker configuration file.
    a. Open /etc/sysconfig/docker in a text editor.
    b. Add the local Docker registry endpoint to the end of the OPTIONS declaration.
    Replace Registry-Endpoint with the same value used for the SERVICED_DOCKER_REGISTRY variable:

```
--insecure-registry=Registry-Endpoint
```

    Note: Use a space character ( ) to separate flags, and make sure the double quote character (") delimits the declaration of OPTIONS.
    c. Save the file, and then close the editor.
5. Restart the Docker service.

```
systemctl restart docker
```

Setting internal services endpoints

Use this procedure to configure a delegate host with the endpoints of the Control Center internal services.

The following variables identify the internal services endpoints:

- SERVICED_ZK
- SERVICED_ENDPOINT
- SERVICED_LOG_ADDRESS
- SERVICED_LOGSTASH_ES
- SERVICED_STATS_PORT

Perform these steps:

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Edit the Control Center configuration file.
    a. Open /etc/default/serviced in a text editor.
    b. For each endpoint variable, locate the line that sets the variable, and then make a copy of the line, immediately below the original.
    c. Remove the number sign character (#) from the beginning of the line.
    d. Replace localhost or {{SERVICED_MASTER_IP}} with the IP address or hostname of the master host.
    e. Save the file, and then close the editor.
3. Verify the settings in the serviced configuration file.

```
grep -E '^\b*[A-Z_]+' /etc/default/serviced
```

Starting Control Center on delegate hosts

Use this procedure to start serviced on a delegate host for the first time.

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Verify the settings in the serviced configuration file.

   ```
   grep -E '^\b*[A-Z_]+' /etc/default/serviced
   ```

3. Start the Control Center service (serviced).

   ```
   systemctl start serviced
   ```

   To monitor progress, enter the following command:

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

Authenticating delegate hosts

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 containing 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.

The following procedures demonstrate how to add a host to a resource pool and install its key bundle.

- Adding a delegate host to a pool through SSH
- Adding a delegate to a pool with a file

Adding a delegate host to a pool through SSH

To succeed, the following statements about the login account used to perform this procedure must be true:

- The account exists on both the master host and on the delegate host.
- The account has `serviced` CLI privileges.
- The account has either public key authentication or password authentication enabled on the master host and on the delegate host.

Use this procedure to add a delegate host to a resource pool through an SSH connection.

1. Log in to the Control Center master host as a user with serviced CLI privileges.
2. Add a delegate host to a resource pool.

   If the master and delegate host are configured for key-based access, the following command does not prompt you to add the delegate to the list of known hosts or to provide the password of the remote user account.

   Use a hostname or IP address to identify the delegate. If you use a hostname, all Control Center hosts must be able to resolve it, either through an entry in `/etc/hosts` or through a nameserver on the network.

   In the following example, replace *Hostname-Or-IP* with the hostname or IP address of a delegate host, and replace *Resource-Pool* with the name of a resource pool.

   If the host is behind a router or firewall for network address translation (NAT), include the `--nat-address` option to specify the NAT device's hostname or IP address and port of the delegate host.

   ```
   serviced host add --register Hostname-Or-IP:4979 Resource-Pool [--nat-address==NAT-Hostname-Or-IP:NAT-Port]
   ```

Adding a delegate to a pool with a file

Use this procedure to add a delegate host to a resource pool with a key bundle file.

1. Log in to the Control Center master host as a user with `serviced` CLI privileges.
2. Add a delegate host to a resource pool.

   Use a hostname or IP address to identify the delegate. If you use a hostname, all Control Center hosts must be able to resolve it, either through an entry in `/etc/hosts` or through a nameserver on the network.

   In the following example, replace *Hostname-Or-IP* with the hostname or IP address of a delegate host, and replace *Resource-Pool* with the name of a resource pool.

   If the host is behind a router or firewall for network address translation (NAT), include the `--nat-address` option to specify the NAT device's hostname or IP address and port of the delegate host.

   ```
   serviced host add Hostname-Or-IP:4979 Resource-Pool [--nat-address==NAT-Hostname-Or-IP:NAT-Port]
   ```

   The command creates a unique key bundle file in the local directory.
3. Use a file transfer utility such as `scp` to copy the key bundle file to the delegate host.
   Once copied to the delegate host, the key bundle file is not needed on the master host and can be deleted.
4. Log in to the Control Center delegate host as a user with `serviced` CLI privileges.
5. Install the key bundle.
   Replace Key-Bundle-Path with the pathname of the key bundle file:

   ```
   serviced host register Key-Bundle-Path
   ```

6. Delete the key bundle file.
   The file is no longer needed on the delegate host.
   Replace Key-Bundle-Path with the pathname of the key bundle file:

   ```
   rm Key-Bundle-Path
   ```

Setting the connection timeout of a resource pool

Use this procedure to set the length of time the services scheduler waits for a disconnected delegate host to reconnect before moving its services to a different host in the resource pool. This affects all multi-host resource pools. Zenoss recommends using a minimum value of 15 seconds for all multi-host pools. For resource pools that are connected through high-latency, wide-area networks, Zenoss recommends values greater than 15 seconds.

1. Log in to the master host as a user with serviced CLI privileges.
2. Display the list of resource pools and their connection timeout values.
   ```
   serviced pool list -v | grep -E 'ID|ConnectionTimeout'
   ```
3. Set the connection timeout value of a resource pool.
   This command accepts the following units identifiers:
   - ms (milliseconds)
   - s (seconds)
   - m (minutes)
   - h (hours)

   Replace Pool-ID with a resource pool identifier, and replace Timeout+Units with an integer followed by a units identifier:
   ```
   serviced pool set-conn-timeout Pool-ID Timeout+Units
   ```

Configuring a ZooKeeper ensemble

This section describes how to create a ZooKeeper ensemble (cluster) for a multi-host Control Center deployment that includes a minimum of three hosts.

If your deployment includes just one host or two hosts, skip this section.

Understanding the configuration process

The procedures in this section instruct you to create temporary variables that are used as building blocks, to construct Control Center configuration variables accurately. You append the Control Center variables to `/etc/default/serviced`, and then edit the file to move the variables to more appropriate locations.

The most important temporary variables specify the IP address or hostname of each host in the ZooKeeper ensemble. The following table identifies these important variables, the names and values of which must be identical on every Control Center host.

Variable name	Placeholder value	Actual value
node1	Master	The IP address or hostname of the master host.
node2	Delegate-A	The IP address or hostname of delegate host A.
node3	Delegate-B	The IP address or hostname of delegate host B.

All ensemble hosts should be on the same subnet.

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.

ZooKeeper variables

The variables in the following table are set only on ZooKeeper ensemble nodes, except SERVICED_ZK, which must be identical on all Control Center hosts.

Variable	Where to set
SERVICED_ISVCS_START	ZooKeeper ensemble nodes
SERVICED_ISVCS_ZOOKEEPER_ID	ZooKeeper ensemble nodes
SERVICED_ISVCS_ZOOKEEPER_QUORUM	ZooKeeper ensemble nodes
SERVICED_ZK	All Control Center hosts
SERVICED_ZK_SESSION_TIMEOUT	ZooKeeper ensemble nodes

Example multi-host ZooKeeper configuration

This example shows the ZooKeeper variables in the /etc/default/serviced configuration file of each host in a 4-node Control Center deployment. For convenience, the relevant settings for each node or host are also included in subsequent procedures.

Master host and ZooKeeper ensemble node, 198.51.100.135:

```
SERVICED_ISVCS_ZOOKEEPER_ID=1
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@0.0.0.0:2888:3888,2@198.51.100.136:2888:3888,3@198.51.100.137:2888:3888
SERVICED_ZK_SESSION_TIMEOUT=15
```

Delegate host and ZooKeeper ensemble node, 198.51.100.136:

```
SERVICED_ISVCS_START=zookeeper
SERVICED_ISVCS_ZOOKEEPER_ID=2
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@198.51.100.135:2888:3888,2@0.0.0.0:2888:3888,3@198.51.100.137:2888:3888
SERVICED_ZK_SESSION_TIMEOUT=15
```

Delegate host and ZooKeeper ensemble node, 198.51.100.137:

```
SERVICED_ISVCS_START=zookeeper
SERVICED_ISVCS_ZOOKEEPER_ID=3
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@198.51.100.135:2888:3888,2@198.51.100.136:2888:3888,3@0.0.0.0:2888:3888
SERVICED_ZK_SESSION_TIMEOUT=15
```

Delegate host, 198.51.100.138:

```
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
```

Configuring the master host as a ZooKeeper node

This procedure configures the Control Center master host as a node in a ZooKeeper ensemble.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Define the IP address variables for each node in the ZooKeeper ensemble.
   Replace Master with the IP address or hostname of the Control Center master host, and replace Delegate-A and Delegate-B with the IP addresses or hostnames of the delegate hosts to include in the ensemble:

   ```
   node1=Master
   node2=Delegate-A
   node3=Delegate-B
   ```

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 can 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 or hostname of the master host with 0.0.0.0.
   You can copy the following 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. Specify the timeout for inactive connections.
   You can copy the following text and paste it in your console:

   ```
   echo "SERVICED_ZK_SESSION_TIMEOUT=15" >> /etc/default/serviced
   ```

7. Clean up the Control Center configuration file.
   a. Open /etc/default/serviced in 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.
   c. Locate the original SERVICED_ZK variable declaration, and then paste the cut line immediately below it.
   d. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_ZOOKEEPER_ID variable declaration at that location.
   e. Locate the original SERVICED_ISVCS_ZOOKEEPER_ID variable declaration, and then paste the cut line immediately below it.
   f. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration at that location.
   g. Locate the original SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration, and then paste the cut line immediately below it.
   h. Navigate to the end of the file, and cut the line that contains the SERVICED_ZK_SESSION_TIMEOUT variable declaration at that location.
   i. Locate the original SERVICED_ZK_SESSION_TIMEOUT variable declaration, and then paste the cut line immediately below it.
   j. Save the file, and then close the editor.
8. Verify the ZooKeeper environment variables.

   ```
   grep -E '^\b*SERVICED' /etc/default/serviced | grep -E '_Z(OO|K)'
   ```

   The following example shows the environment variables for a master host with IP address 198.51.100.135.

   ```
   SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
   SERVICED_ISVCS_ZOOKEEPER_ID=1
   SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@0.0.0.0:2888:3888,2@198.51.100.136:2888:3888,3@198.51.100.137:2888:3888
   SERVICED_ZK_SESSION_TIMEOUT=15
   ```

Configuring delegate host A as a ZooKeeper node

Use this procedure to configure the delegate host designated as Delegate-A as a ZooKeeper node.

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Define the IP address variables for each node in the ZooKeeper ensemble.
   Replace Master with the IP address or hostname of the Control Center master host, and replace Delegate-A and Delegate-B with the IP addresses or hostnames of the delegate hosts to include in the ensemble:

   ```
   node1=Master
   node2=Delegate-A
   node3=Delegate-B
   ```

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

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

4. Specify the nodes in the ZooKeeper ensemble.
   You can 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 or hostname of delegate host A with 0.0.0.0.
   You can copy the following text and paste it in your console:

   ```
   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
   ```

6. Specify the timeout for inactive connections.
   You can copy the following text and paste it in your console:

   ```
   echo "SERVICED_ZK_SESSION_TIMEOUT=15" >> /etc/default/serviced
   ```

7. Configure Control Center to start the ZooKeeper service.
   You can copy the following text and paste it in your console:

   ```
   echo "SERVICED_ISVCS_START=zookeeper" >> /etc/default/serviced
   ```

8. Clean up the Control Center configuration file.
   a. Open /etc/default/serviced in 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.
   c. Locate the original SERVICED_ZK variable declaration, and then paste the cut line immediately below it.
   d. Comment the original SERVICED_ZK declaration, which references only the master host.
      Insert the number sign character (#) immediately in front of the original SERVICED_ZK variable.
   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 original SERVICED_ISVCS_ZOOKEEPER_ID variable declaration, and then paste the cut line immediately below it.
   g. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration at that location.
   h. Locate the original SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration, and then paste the cut line immediately below it.
   i. Navigate to the end of the file, and cut the line that contains the SERVICED_ZK_SESSION_TIMEOUT variable declaration at that location.
   j. Locate the original SERVICED_ZK_SESSION_TIMEOUT variable declaration, and then paste the cut line immediately below it.
   k. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_START variable declaration at that location.
   l. Locate the original SERVICED_ISVCS_START variable declaration, and then paste the cut line immediately below it.
   m. Save the file, and then close the editor.
9. Verify the ZooKeeper environment variables.

   ```
   grep -E '^\b*SERVICED' /etc/default/serviced | grep -E '(CS_ZO|_ZK|CS_ST)'
   ```

   The following example shows the environment variables for a delegate host with IP address 198.51.100.136.

```
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
SERVICED_ISVCS_START=zookeeper
SERVICED_ISVCS_ZOOKEEPER_ID=2
SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@198.51.100.135:2888:3888,2@0.0.0.0:2888:3888,3@198.51.100.137:2888:3888
SERVICED_ZK_SESSION_TIMEOUT=15
```

Configuring delegate host B as a ZooKeeper node

Use this procedure to configure the delegate host designated as Delegate-B as a ZooKeeper node.

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Define the IP address variables for each node in the ZooKeeper ensemble.
   Replace Master with the IP address or hostname of the Control Center master host, and replace Delegate-A and Delegate-B with the IP addresses or hostnames of the delegate hosts to include in the ensemble:

   ```
   node1=Master
   node2=Delegate-A
   node3=Delegate-B
   ```

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

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

4. Specify the nodes in the ZooKeeper ensemble.
   You can 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 or hostname of delegate host B with 0.0.0.0.
   You can copy the following text and paste it in your console:

   ```
   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. Specify the timeout for inactive connections.
   You can copy the following text and paste it in your console:

   ```
   echo "SERVICED_ZK_SESSION_TIMEOUT=15" >> /etc/default/serviced
   ```

7. Configure Control Center to start the ZooKeeper service.
   You can copy the following text and paste it in your console:

   ```
   echo "SERVICED_ISVCS_START=zookeeper" >> /etc/default/serviced
   ```

8. Clean up the Control Center configuration file.
   a. Open /etc/default/serviced in 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.
   c. Locate the original SERVICED_ZK variable declaration, and then paste the cut line immediately below it.
   d. Comment the original SERVICED_ZK declaration, which references only the master host.
      Insert the number sign character (#) immediately in front of the original SERVICED_ZK variable.
   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 original SERVICED_ISVCS_ZOOKEEPER_ID variable declaration, and then paste the cut line immediately below it.
   g. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration at that location.
   h. Locate the original SERVICED_ISVCS_ZOOKEEPER_QUORUM variable declaration, and then paste the cut line immediately below it.
   i. Navigate to the end of the file, and cut the line that contains the SERVICED_ZK_SESSION_TIMEOUT variable declaration at that location.
   j. Locate the original SERVICED_ZK_SESSION_TIMEOUT variable declaration, and then paste the cut line immediately below it.
   k. Navigate to the end of the file, and cut the line that contains the SERVICED_ISVCS_START variable declaration at that location.
   l. Locate the original SERVICED_ISVCS_START variable declaration, and then paste the cut line immediately below it.
   m. Save the file, and then close the editor.
9. Verify the ZooKeeper environment variables.

   ```
   grep -E '^\b*SERVICED' /etc/default/serviced | grep -E '(CS_ZO|_ZK|CS_ST)'
   ```

   The following example shows the environment variables for a delegate host with IP address 198.51.100.137.

```
SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
SERVICED_ISVCS_START=zookeeper
SERVICED_ISVCS_ZOOKEEPER_ID=3
SERVICED_ISVCS_ZOOKEEPER_QUORUM=1@198.51.100.135:2888:3888,2@198.51.100.136:2888:3888,3@0.0.0.0:2888:3888
SERVICED_ZK_SESSION_TIMEOUT=15
```

Importing the Docker image for ZooKeeper

Perform the steps in Downloading and staging required files before performing this procedure.

Use this procedure to import the Docker image for ZooKeeper on delegate hosts A and B. This procedure is not necessary on the master host.

1. Log in to the host as root, or as a user with superuser privileges.
2. Change directory to /root.

   ```
   cd /root
   ```

3. Extract the ZooKeeper image.

   ```
   yes | ./install-zenoss-isvcs-zookeeper_v*.run
   ```

4. Optional: Delete the archive file, if desired.

   ```
   rm -i ./install-zenoss-isvcs-zookeeper_v*.run
   ```

Starting a ZooKeeper ensemble

Use this procedure to start a ZooKeeper ensemble.
This procedures uses the `nc` utility to query ensemble hosts. If `nc` is not available, you can use `telnet` with interactive ZooKeeper commands.

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 (Delegate-A) as root, or as a user with superuser privileges.
3. In a different window, log in to the third node of the ZooKeeper ensemble (Delegate-B) as root, or as a user with superuser privileges.
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.
   a. Attach to the container of the ZooKeeper service.

      ```
      docker exec -it serviced-isvcs_zookeeper /bin/bash
      ```

   b. Define IP address variables for each node in the ZooKeeper ensemble.
      Replace Master with the IP address or hostname of the Control Center master host, and replace Delegate-A and Delegate-B with the IP addresses or hostnames of the delegate hosts in the ensemble:

      ```
      node1=Master
      node2=Delegate-A
      node3=Delegate-B
      ```

   c. Query the master host and identify its role in the ensemble.

      ```
      { echo stats; sleep 1; } | nc $node1 2181 | grep Mode
      ```

      The result includes leader or follower.
   d. Query delegate host A and identify its role in the ensemble.

      ```
      { echo stats; sleep 1; } | nc $node2 2181 | grep Mode
      ```

   e. Query delegate host B and identify its role in the ensemble.

      ```
      { echo stats; sleep 1; } | nc $node3 2181 | grep Mode
      ```

   f. Detach from the container of the ZooKeeper service.

      ```
      exit
      ```

   If none of the hosts reports that it is the ensemble leader within a few minutes of starting serviced, reboot the hosts.
   Once the ZooKeeper quorum is running, update and start the remaining delegate hosts.

Updating and starting other hosts

The default configuration of delegate hosts sets the value of the SERVICED_ZK variable to the master host only. Use this procedure to update the setting to include all of the hosts in the ZooKeeper ensemble. **Perform this procedure on each host in your Control Center deployment that is not a ZooKeeper ensemble node.**

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Define the IP address variables for each node in the ZooKeeper ensemble.
   Replace Master with the IP address or hostname of the Control Center master host, and replace Delegate-A and Delegate-B with the IP addresses or hostnames of the delegate hosts to include in the ensemble:

   ```
   node1=Master
   node2=Delegate-A
   node3=Delegate-B
   ```

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

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

4. Update the variable.
   a. Open /etc/default/serviced in 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 three endpoints.
   c. Locate the SERVICED_ZK variable near the beginning of the file, and then delete the line it is on.
      The value is just the master host endpoint.
   d. Paste the SERVICED_ZK variable declaration from the end of the file in the location of the just-deleted declaration.
   e. Save the file, and then close the editor.
5. Verify the setting.

   ```
   grep -E '^\b*SERVICED_ZK' /etc/default/serviced
   ```

   The following example shows the environment variable for a delegate host that is not a node in the ZooKeeper ensemble:

   ```
   SERVICED_ZK=198.51.100.135:2181,198.51.100.136:2181,198.51.100.137:2181
   ```

6. Restart Control Center.

   ```
   systemctl restart serviced
   ```

Resolving package dependency conflicts

This section includes procedures for resolving Docker CE and Control Center dependency conflicts that may occur during installation.

Resolving device mapper dependency conflicts

To perform this procedure, you need:

- An RHEL/CentOS system with internet access and the same operating system release and kernel as the Control Center hosts in your deployment.
- A secure network copy program.

Use this procedure to resolve dependency issues in which the installed versions of device mapper libraries are newer than the versions included in the Zen oss mirror. The following example shows a typical yum error of this type:

```
Error: Package: 7:device-mapper-event-1.02.107-5.el7.x86_64 (zenoss-mirror)
Requires: device-mapper = 7:1.02.107-5.el7
Installed: 7:device-mapper-1.02.107-5.el7_2.5.x86_64 (@updates)
device-mapper = 7:1.02.107-5.el7_2.5
```

Follow these steps:

1. Display the version number of the installed device mapper package.

   ```
   rpm -q device-mapper | cut -d - -f 3-
   ```

   Example result:

   ```
   1.02.135-1.el7_3.1.x86_64
   ```

   Record the version number for subsequent use.
2. Log in to a compatible host that is connected to the internet as root, or as a user with superuser privileges.
   The host must have the same operating system (RHEL or CentOS) and release installed as the Control Center hosts in your deployment.
3. Install yum utilities, if necessary.
   a. Determine whether the yum utilities package is installed.

      ```
      rpm -qa | grep yum-utils
      ```

      - If the command returns a result, the package is installed. Proceed to the next step.
      - If the command does not return a result, the package is not installed. Perform the following substep.

   b. Install the yum-utils package.

      ```
      yum install yum-utils
      ```

4. Download the required dependencies, and then create a tar archive of the files.
   a. Create a variable for the dependency version to download.
      Replace Device-Mapper-Version with the version number displayed in a previous step:

      ```
      myVersion=Device-Mapper-Version
      ```

   b. Create a temporary directory for the dependencies.

      ```
      mkdir /tmp/downloads
      ```

   c. Download the dependencies to the temporary directory.

      ```
      yum install --downloadonly --downloaddir=/tmp/downloads device-mapper-event-$myVersion
      ```

      The yum command downloads two package files.
   d. Create a tar archive of the temporary directory.

      ```
      cd /tmp && tar czf ./downloads.tgz ./downloads
      ```

5. Use a secure copy program to copy the archive file to the /tmp directory of the Control Center host or hosts that need the dependencies.
6. Log in to the host as root, or as a user with superuser privileges.
7. Install the device mapper dependencies.
   a. Extract the packages from the tar archive.

      ```
      cd /tmp && tar xzf ./downloads.tgz
      ```

   b. Install the dependencies.

```
yum install $(ls /tmp/downloads/*.rpm)
```

Return to the procedure you were performing before turning to this section and retry the yum install command that failed previously.

Resolving other dependency conflicts

Use this procedure to resolve dependency issues in which the installed versions of one or more dependencies are newer than the versions included in the Zenoss mirror. The following example shows a typical yum error of this type:

```
Error: Package: policycoreutils-python-2.5-9.el7.x86_64 (zenoss-mirror)
Requires: policycoreutils = 2.5-9.el7
Installed: policycoreutils-2.5-11.el7_3.x86_64 (@updates)
```

Follow these steps:

1. Install the older package.
   Replace Package-Name with the name of the package displayed in the error message:
   ```
   rpm -Uvh --oldpackage Package-Name
   ```
2. Clean all yum caches.
   ```
   yum clean all
   ```

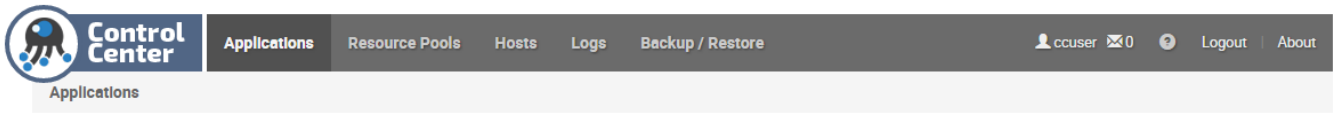
Return to the procedure you were performing before turning to this section and retry the yum install command that failed previously.

Browser interface overview

This section provides high-level reference information about the Control Center browser interface. For context-specific information about elements in the browser interface, hover the pointer over items to open tooltips and information boxes.

Main menu and navigation

The menu bar at the top of the Control Center browser interface provides access to functional areas of the application, user and application information, and documentation.



Applications page

The Applications page lists managed applications, the services that support Control Center, application templates, and a graph showing thin pool usage.

In the Applications area, to display a details page for an application, click its name. To display the Internal Services page, click Internal Services. You can also search for, start or stop, and add or delete an application. In the Resource Pools column, click the resource pool name to access a details page. In the Public Endpoint column, click the link to open an application.

In the Application Templates area, you can search for and add or delete an application template.

The graphs area shows thin pool usage information. You can change range, aggregator, and refresh settings. For more information about thin pool requirements, see Control Center application data storage requirements.

Internal Services details

The Internal Services page provides summary information about the services upon which Control Center relies. To view graph data associated with a service, click on its name.

Application details

The application details page includes controls for and information about the application.

For the Zenoss application, you can view service logs, edit services, edit variables, and start or stop the application. Details include public endpoints, IP assignments, configuration files, services associated with the application, and instances of the application.

In the Public Endpoints area, you can search for and add public endpoints. The table lists public endpoint information for the services that are associated with the application. You can start, stop, and delete service public endpoints.

The Instances area lists information about each instance of the application. You can search for an instance, access the host, view logs, and restart an instance.

Graphs visually represent the following information about the application:

- CPU usage by the system and users
- Memory usage (total RSS and cache)
- DFS usage
- Network usage
- Open connection information

For more information, see Emergency shutdown of services and Rolling restart of services.

IP Assignments

The IP Assignments area lists the services that have external (public) IP addresses. You can change the external IP address for a service.

Configuration Files table

The Configuration Files table lists any configuration files that are associated with the application. You can search for and edit configuration files.

To edit a configuration file, click Edit in the Actions column. Changes are not shared with other instances. For more information about the files, see the configuration documentation for your application.

Services

The Services table contains a nested list of services and subservices that are associated with an application.

- To expand or collapse the list of subservices, click the arrow beside a service name. Services at the same level are peers. An indented service is a subservice of the preceding service.
- To access a service details page, click a service name.
- The Status column shows the start or stop status of a service.
- The Health column indicates health-check results for started services. If multiple instances are running, the number is noted. Hover the mouse over the icon for additional details.
- The Actions column contains the controls for starting and stopping a service. Stop applies to all instances of the service and discards the containers in which the instances were running. Start creates new containers for the required number of instances. Action controls for the entire application are in the upper right corner of the window.

Because the data series used by the service graphs is tagged by serviceid and hostid, when a single instance service is restarted and ends up on a different host, there are two data series in the database for that service. While both series are present, sum and average will show different values. For data after the last transition, sum and average will be the same.

Resource Pools page

The Resource Pools page lists the resource pools that Control Center employs, with aggregate information about the hosts in each pool.

The Resource Pools table shows the total resources available in each pool. You can search for and add resource pools. To view a details page for a pool, click the pool name.

The resource pool details page displays pool summary information. You can add a virtual IP for the pool, view hosts, and edit the following settings for the resource pool:

- Name of the resource pool.
- Length of time that the scheduler waits for a disconnected delegate host to rejoin its pool before moving the services scheduled for the delegate to a different host in the pool. This feature is useful for remote resource pools that are connected through a high-latency, wide-area network.
- Permissions, which requires a restart of the delegate.

Hosts page

The Hosts page displays information about the individual hosts that are included in all resource pools that Control Center employs.

The Hosts table lists each machine that is assigned to a resource pool. Summary information about each host includes whether it is active, its resource pool, memory, RAM limit and usage (for the Docker containers only), CPU cores, kernel version, and the version of Control Center that the host is running, and actions that you can perform on the host.

You can search for, add, and delete hosts. To display details about a host, click the host name.

The memory values in the table on the Hosts page differ from those in the Memory Usage graph on the host details page. The table shows memory usage for the Docker containers only. The graph shows total system memory usage for the Docker containers and all services that are running on the system.

Host details

The hosts details page displays information about a specific machine that is assigned to a resource pool.

You can edit the host's RAM limit, and reset the host's authentication keys and generate a new key pair. When you reset keys, services that are running on the host cannot connect until you register the new keys by using the serviced host register command.

The graphs show dynamic information about the host's CPU usage, memory usage (total for Docker containers and all services that are running on the system), load average, and other details. You can filter the information that is displayed in a graph. For example, in the Memory Usage graph, click Used to hide used memory, and then click it again to display used memory. You can edit the graph settings to change the time range, whether the graph shows the sum or the average, and the refresh frequency.

Below the graphs is IP and service instance information. You can search each table for specific IP or service information.

Logs page

The Logs page displays the log information that Control Center collects.

The log information is presented in the Kibana JavaScript interface, which provides a variety of methods for sorting, querying, and analyzing log data.

For more information about Kibana, see the Kibana Documentation.

Backup/Restore page

The Backup/Restore page displays a list of backup files for the system.

You can search for and create a backup and restore from a backup file.

For more information about backing up and restoring the system, see Backing up and restoring.

Command-line interface reference

This section contains information about and procedures for using the Control Center command-line interface (CLI). The CLI uses the same application programming interface (API) that the browser interface uses. To perform many actions, you can use either interface.

Invoking serviced

To use `serviced`, you need a login account on hosts in Control Center pools. The account must be a member of the `docker` group.

You can invoke `serviced` from the Control Center master host or a delegate host. For example, to list the services running on the master host, enter the following command:

```
serviced service list
```

To invoke `serviced` on a delegate host, you can specify the master host and port by using the `--endpoint` global option. Replace *Master-Host* with the hostname or IP address of the Control Center master host.

Example:

```
serviced --endpoint Master-Host:4979 service list
```

To commit a container, you must run a `serviced` CLI session on the Control Center master host.

serviced

NAME

`serviced` - A container-based management system.

SYNTAX

```
serviced [global options] command [command options] [arguments...]
```

DESCRIPTION

`serviced` is an open-source, application service orchestrator based on Docker Community Edition.

GLOBAL OPTIONS

`--docker-registry` *Master-Hostname*`:5000`

The local Docker registry to use. See SERVICED_DOCKER_REGISTRY.

`--static-ip` *IP-Address* `[--static-ip` *IP-Address*`] ...`

One or more static IP addresses for a `serviced` instance to advertise. See SERVICED_STATIC_IPS.

`--endpoint Host:Port`

The serviced RPC endpoint. See SERVICED_ENDPOINT.

`--outbound` *IP-Address*

See SERVICED_OUTBOUND_IP.

`--uiport:`*Port*

The port on which the HTTP server listens for requests. The default value is 443, unless SERVICED_UI_PORT is set.

`--nfs-client` *Value*

Determines whether a `serviced` delegate mounts the DFS. The default value is 1 (enable) unless SERVICED_NFS_CLIENT is set.

Before changing the default, ensure that no stateful services can run on the host. Disabling the DFS can destroy application data.

To disable mounting, set the value to 0.

`--listen:`*Port*

The `serviced` RPC endpoint on the local host. The default value of *Port* is 4979.

`--docker-dns` *Option* `[--docker-dns` *Option*`] ...`

One or more DNS configuration flags for Docker to use when starting containers.

`--master`

Run the application services scheduler and other internal services.

`--agent`

Run application services scheduled by the master.

`--mux` *Port*

The port used for traffic among Docker containers. The default value is 22250, unless SERVICED_MUX_PORT is set.

`--mux-disable-tls` *Value*

Determines whether inter-host traffic among Docker containers is encrypted with TLS. Intra-host traffic among Docker containers is not encrypted.

The default value is 0 (enabled) unless SERVICED_MUX_DISABLE_TLS is set to 1 (disable encryption).

`--mux-tls-ciphers` *Option* `[--mux-tls-ciphers` *Option*`] ...`

The list TLS ciphers serviced can use for mux traffic. The default may be set in by SERVICED_MUX_TLS_CIPHERS.

`--mux-tls-min-version`

The minimum version of TLS that serviced accepts for mux traffic. See SERVICED_MUX_TLS_MIN_VERSION.

`--var` *Path*

This option has been deprecated.

`--volumes-path` *Path*

The location of `serviced` application data. The default path is `/opt/serviced/var/volumes` unless SERVICED_VOLUMES_PATH is set.

`--isvcs-path` *Path*

The location of `serviced` internal services data. The default path is `/opt/serviced/var/isvcs` unless SERVICED_ISVCS_PATH is set.

`--backups-path` *Path*

The location of `serviced` backup files. The default path is `/opt/serviced/var/backups` unless SERVICED_BACKUPS_PATH is set.

`--etc-path` *Path*

The location of `serviced` configuration files. The default path is `/opt/serviced/etc` unless SERVICED_ETC_PATH is set.

`--keyfile` *Path*

The path of a digital certificate key file. See SERVICED_KEY_FILE.

`--certfile` *Path*

The path of a digital certificate file. See SERVICED_CERT_FILE.

`--zk` *Host*`:`*Port* `[--zk` *Host*`:`*Port*`] ...`

One or more ZooKeeper endpoints. See SERVICED_ZK.

`--mount` *Option* `[--mount` *Option*`] ...`

One or more bind mounts for a container. The syntax for *Option* is `DOCKER_IMAGE,HOST_PATH[,CONTAINER_PATH]`.

`--fstype` *Driver*

The driver to manage application data storage on the `serviced` master host. The default is `devicemapper` unless SERVICED_FS_TYPE is set.

`--alias` *Alias* `[--alias` *Alias*`] ...`

One or more DNS aliases to associate with a container.

`--es-startup-timeout` *Duration*

The number of seconds to wait for Elasticsearch to complete its startup. The default value is 600 seconds (10 minutes).

`--max-container-age` *Duration*

The number of seconds `serviced` waits before removing a stopped container. The default value is 86400 seconds (24 hours), unless SERVICED_MAX_CONTAINER_AGE is set in the configuration file.

`--max-dfs-timeout` *Duration*

The number of seconds to wait for a snapshot to complete. The default value is 300 seconds (5 minutes).

`--virtual-address-subnet` *Subnet*

The private subnet for containers that use virtual IP addresses on a host. The default value is 10.3.0.0/16, unless SERVICED_VIRTUAL_ADDRESS_SUBNET is set in the configuration file.

`--master-pool-id` *Pool-ID*

The name of the resource pool to which the `serviced` instance configured as master belongs. The default value of *Pool-ID* is `default`.

`--admin-group` *Group*

The name of the Linux group on the `serviced` master host whose members are authorized to use the `serviced` browser interface. The default is `wheel` unless SERVICED_ADMIN_GROUP is set in the configuration file.

`--storage-opts` *Option* `[--storage-opts` *Option*`] ...`

Storage arguments to initialize the filesystem.

`--isvcs-start` *Option* `[--isvcs-start` *Option*`] ...`

Enables one or more internal services to run on a delegate host. Currently, only `zookeeper` has been tested. If SERVICED_ISVCS_START is set in the configuration file, its value is used.

`--isvcs-zk-id` *Identifier*

The unique identifier (a positive integer) of a ZooKeeper ensemble node. If SERVICED_ISVCS_ZOOKEEPER_ID is set in the configuration file, its value is used.

`--isvcs-zk-quorum` *Option* `[--isvcs-zk-quorum` *Option*`] ...`

The list of nodes in a ZooKeeper ensemble. If SERVICED_ISVCS_ZOOKEEPER_QUORUM is set in the configuration file, its value is used.

`--tls-ciphers` *Option* `[--tls-ciphers` *Option*`] ...`

The list TLS ciphers that `serviced` accepts for HTTP traffic. If SERVICED_TLS_CIPHERS is set in the configuration file, its value is used.

`--tls-min-version` *Version*

The minimum version of TLS that `serviced` accepts for HTTP traffic. Valid values include the default, `VersionTLS11`, and `VersionTLS12`. If SERVICED_TLS_MIN_VERSION is set in the configuration file, its value is used.

`--report-stats`

Enable reporting statistics in a container.

`--host-stats` *Host*:*Port*

The endpoint of the `serviced` metrics consumer service. The default value of Host is the IP address of the master host, and the default value of Port is 8443. If SERVICED_STATS_PORT is set in the configuration file, its value is used instead of the default endpoint.

`--stats-period` *Duration*

The frequency, in seconds, at which delegates gather metrics to send to the `serviced` metrics consumer service on the master host. The default value of Duration is 10, unless SERVICED_STATS_PERIOD is set in the configuration file.

`--mc-username` *User*

The username of the OpenTSDB account that MetricConsumer uses gain access to data stored by `serviced`.

`--mc-password` *Password*

The password of the OpenTSDB account that MetricConsumer uses gain access to data stored by `serviced`.

`--cpuprofile`

Instructs a container to write its CPU profile to a file.

`--isvcs-env` *Option* `[--isvcs-env` *Option*`] ...`

Startup arguments to pass to internal services. The default value is no arguments, unless SERVICED_ISVCS_ENV_[0-9]+ is set in the configuration file.

`--debug-port` *Port*

The port on which `serviced` listens for HTTP requests for the Go profiler. The default value of Port is 6006, unless SERVICED_DEBUG_PORT is set in the configuration file. To stop listening for requests, set the value to -1.

`--max-rpc-clients` *Count*

The preferred maximum number of simultaneous connections a `serviced` delegate uses for RPC requests. The value is used to create a pool of sockets, which are reused as needed. Increasing the value increases the number of open sockets and the use of socket-related operating system resources.

When the demand for connections exceeds the supply of open sockets, `serviced` opens more sockets. When demand eases, `serviced` reduces the number of open sockets to the preferred maximum.

The default value is 3, unless SERVICED_MAX_RPC_CLIENTS is set in the configuration file. For more information, see SERVICED_MAX_RPC_CLIENTS.

`--rpc-dial-timeout` *Duration*

The number of seconds `serviced` waits before giving up on attempts to connect to the RPC endpoint on the master host.

`--rpc-cert-verify` *Value*

Determines whether `serviced` is enabled to perform TLS certificate verification for RPC connections. The default value is false (disabled) unless SERVICED_RPC_CERT_VERIFY is set in the configuration file.

`--rpc-disable-tls` *Value*

Determines whether `serviced` enabled to encrypt RPC traffic with TLS. The default value is false (disabled) unless SERVICED_RPC_DISABLE_TLS is set in the configuration file.

`--rpc-tls-ciphers` *Option* `[--rpc-tls-ciphers` *Option*`] ...`

The list of TLS ciphers `serviced` prefers for RPC connections. If SERVICED_RPC_TLS_CIPHERS is set in the configuration file, its value is used.

`--rpc-tls-min-version` *Version*

The minimum version of TLS `serviced` accepts for RPC connections. Valid values include the default, `VersionTLS11`, and `VersionTLS12`. The default value is `VersionTLS10` unless SERVICED_RPC_TLS_MIN_VERSION is set in the configuration file.

`--snapshot-ttl` *Duration*

The number of hours an application data snapshot is retained before removal. The default value is 12 unless SERVICED_SNAPSHOT_TTL is set in the configuration file.

`--snapshot-space-percent` *Value*

The amount of free space in the thin pool, expressed as a percentage the total size. This value is used to determine whether the thin pool can hold a new snapshot. The default value is 20 unless SERVICED_SNAPSHOT_USE_PERCENT is set in the configuration file.

`--controller-binary` *Path*

The path to the container controller binary. The default is `/opt/serviced/bin/serviced-controller`.

`--log-driver` *File*

The log driver for all Docker container logs, including containers for Control Center internal services.

`--log-config` *Option* `[--log-config` *Option*`] ...`

A list of Docker `--log-opt` options as key=value pairs.

`--ui-poll-frequency` *Duration*

The number of seconds between polls from browser interface clients. The value is included in a JavaScript library that is sent to the clients. The default value is 3 unless SERVICED_UI_POLL_FREQUENCY is set in the configuration file.

`--storage-stats-update-interval` *Duration*

The frequency in seconds that the thin pool usage is analyzed. The default value is 300 (five minutes) unless SERVICED_STORAGE_STATS_UPDATE_INTERVAL is set in the configuration file.

`--zk-session-timeout` *Duration*

The number of seconds the ZooKeeper leader waits before flushing an inactive connection. The default value is 15 unless SERVICED_ZK_SESSION_TIMEOUT is set in the configuration file.

`--auth-token-expiry` *Value*

The expiration time, in seconds, of delegate authentication tokens. The default value is 3600 (one hour) unless SERVICED_AUTH_TOKEN_EXPIRATION is set in the configuration file.

`--logtostderr`

Write log messages to STDERR instead of the system log.

`--alsologtostderr`

Write log messages to STDERR as well as the system log.

`--logstashurl` *Host*:*Port*

The endpoint of the logstash service. The default value of Host is the IP address or hostname of the `serviced` master host and the default value of Port is 5042. If SERVICED_LOG_ADDRESS is set in the configuration file, its value is used instead of the default endpoint.

`--logstash-es` *Host*:*Port*

The endpoint of the logstash Elasticsearch service. The default value of *Host* is the IP address of the master host, and the default value of *Port* is 9100. If SERVICED_LOGSTASH_ES is set in the configuration file, its value is used instead of the default endpoint.

`--logstash-max-days` *Duration*

The maximum number of days to keep application logs in the logstash database before purging them. The default value of *Duration* is 14, unless SERVICED_LOGSTASH_MAX_DAYS is set in the configuration file. When this argument and `--logstash-max-size` are used at the same time, both conditions are evaluated and enforced.

`--logstash-max-size` *Quantity*

The maximum size of the logstash database, in gigabytes. When this argument and `--logstash-max-days` are used at the same time, both conditions are evaluated and enforced. The default value of Quantity is 10, unless SERVICED_LOGSTASH_MAX_SIZE is set in the configuration file.

`--logstash-cycle-time` *Duration*

The amount of time between logstash purges, in hours. The default value is 6 unless SERVICED_LOGSTASH_CYCLE_TIME is set in the configuration file.

`--v` *Level*

The log level `serviced` uses when writing to the system log. Valid values are 0 (normal) and 2 (debug). The default value is 0, unless SERVICED_LOG_LEVEL is set in the configuration file.

`--stderrthreshold` *Level*

Write log messages at or above Level to STDERR, in addition to the system log. The value of Level may be 0 (INFO), 1 (WARNING), 2 (ERROR), or 3 (FATAL). The default value is 2.

`--vmodule`

Module-specific logging. For more information, refer to the Google Logging documentation.

`--log_backtrace_at` *File*`:`*Line*

Emit a stack trace when logging hits the specified line and file.

`--config-file` *Path*

The path of the configuration file. The default is `/etc/default/serviced`.

`--allow-loop-back` *Value*

Determines whether loop-back files can be used with the `devicemapper` storage driver. This option is not supported for production use.

`--version`

Display minimal version information about the `serviced` binary. To display additional information, use the `serviced` version command.

`[--help|-h]`

Display help information.

COMMANDS

`backup`

Copy all templates, services, and application data into a compressed tar archive file.

`config`

Report on the `serviced` configuration.

`debug`

Manage debugging.

`docker`

Docker administration commands.

`healthcheck`

Report on the health of `serviced`.

`[help|h]`

Display a global or command-specific help message.

`host`

Administer hosts.

`key`

Display the host's public key.

`log`

Administer logs.

`metric`

Administer metrics.

`pool`

Administer resource pool data.

`restore`

Reconstruct templates, services, and application data from a compressed `tar` archive file created with `serviced backup`.

`script`

Verify or perform the commands in a script file.

`service`

Administer services.

`snapshot`

Administer snapshots.

`template`

Administer templates.

`version`

Display `serviced` version information.

`volume`

Administer volume data.

INVOCATION

Service (daemon) control commands include start, stop and reload. The reload command sends SIGHUP to the daemon, which restarts all internal services except ZooKeeper.

```
systemctl [start|stop|reload] serviced
```

MISCELLANEOUS

Sending SIGUSR1 to the `serviced` process toggles the log level between 0 and 2.

To attach to a container running on a remote host, log in to the container from the `serviced` master host. If you are running a Linux shell on a delegate host, you can specify the `--endpoint` option in the `serviced` invocation.

`serviced` relies on Docker, and some administration procedures include `docker` commands. However, commands that manipulate containers directly, such as `docker pause`, should not be used when `serviced` is running.

During installation, `serviced` creates the internal services directory on the master host, so `serviced` commands must be run as `root`, or as a user with superuser privileges. After the master host is added as a delegate, `serviced` commands use the delegate host authorization keys, so `root` is no longer required.

ENVIRONMENT

SERVICED_HOME

The install path of `serviced`. The default value is `/opt/serviced`.

FILES

`/etc/default/serviced`

serviced backup

The `serviced backup` command saves a snapshot of the current state of the system, the state of all services, and application data to a compressed tar archive file (.tgz).

You can back up the entire system or exclude certain directories, such as the directory that contains application performance data.

Before starting a backup, Control Center estimates the size of the backup file and compares it to the amount of free space. If storage space is insufficient, Control Center does not start a backup. Exit code 1 indicates insufficient space. If storage space is insufficient, take action to increase available space, and then try the backup again.

USAGE

`serviced backup [arguments...]` *Backup-Path*

OPTIONS

`--exclude` *Item* `[–exclude` *Item*`] ...`

One or more tenant volume items to exclude from the backup. The following list identifies valid values for *Item*:

- `etl-analytics`
- `hbase-*`
- `mariadb-events`
- `rabbitmq`
- `zeneventserver`
- `var-zenpacks`
- `zenjobs`
- `zencatalogservice`

`--check`

Estimate the backup file size and check available storage space, but do not start the backup.

`--force`

Attempt to perform a backup even if the check for sufficient storage space failed. Use this override option with caution. If the backup exhausts storage space on the device, you must restart `serviced`.

`[--help|–h]`

Show the help message.

EXAMPLE

Create a backup without HBase or the Solr index:

```
serviced backup --exclude hbase-* \
  --exclude zencatalogservice /opt/serviced/var/backups/
```

serviced config

The `serviced config` command displays the active Control Center configuration.

USAGE

```
serviced config
```

serviced debug

The `serviced debug` command allows you to manage debugging of the Control Center application

USAGE

```
serviced debug [global options] command [command options] [arguments...]
```

COMMANDS

`enable-metrics`

Enable debug metrics

`disable-metrics`

Disable debug metrics

`help, h`

Shows a list of commands or help for one command

OPTIONS

`--generate-bash-completion`

`--help, -h`

Show the help message

serviced docker

The `serviced docker` command administers Docker images and the Docker registry.

USAGE

```
serviced docker [global options] command [command options] [arguments...]
```

COMMANDS

`sync`

Asynchronously push all images from the `serviced` Docker registry index into the Docker registry.

`reset-registry`

For upgrades only, download the latest images from the Docker registry and save them into the `serviced` Docker registry index.

`migrate-registry`

Migrate Docker registry data into another remote registry.

`override`

Replace an image in the Docker registry with a new image

`[help|h]`

Show a list of commands or help for a single command.

OPTIONS

`[--help|-h]`

Show the help message.

serviced healthcheck

The `serviced healthcheck` command displays the status of the Control Center internal services health checks.

USAGE

```
serviced healthcheck
```

serviced host

The serviced host command administers host data.

USAGE

`serviced host [global options] command [command options] [arguments...]`

COMMANDS

`list`

List all hosts.

`add` *Hostname-Or-IP*

Add a host.

If the host is behind a router or firewall, include the `--nat-address` option in the invocation.

`[remove|rm]` *Hostname-Or-IP*

Remove a host.

`register`

Set the authentication keys to use for a host. When KEYSFILE is `-`, keys are read from standard input (stdin).

`set-memory` *Hostname-Or-IP*

Set the memory allocation for a specific host.

`[help|h]`

Show a list of commands or help for a single command.

OPTIONS

`--nat-address` *NAT-Hostname-Or-IP*`[:`*Port*`]`

Associate a NAT address with the host. Specify the port if using an IP address instead of a hostname.

`[--help|-h]`

Show the help message.

EXAMPLE

Add host that is behind a NAT:

```
serviced host add Hostname-Or-IP:4979 Resource-Pool \
  --nat-address==NAT-Hostname-Or-IP:NAT-Port
```

serviced key

The `serviced key` command displays or resets a host's public key.

USAGE

serviced key [global options] command [command options] [arguments...]

COMMANDS

`list`

Show the public key for the host.

`reset`

Regenerate the public key for the host.

`[help|h]`

Show a list of commands or the help for a single command.

OPTIONS

`[--help|-h]`

Show the help message.

serviced log export

The `serviced log export` command exports application log files from Logstash for one or more services based on service identifier or service name. By default, the command exports a tar archive file, which contains a separate log file for each unique combination of container and log file name, and an index file summarizing the log files.

USAGE

```
serviced log export [arguments...]
```

OPTIONS

`--from` *Date*

The start date, in yyyy.mm.dd format. The default is yesterday's date.

`--to` *Date*

The end date, in yyyy.mm.dd format. The default is today's date.

`--service` *Service* `[--service` *Service*`] ...`

The name or ID of one or more services. By default, all child services are included.

`--file` *Path* `[--file` *Path*`] ...`

The path or paths of one or more application logs to export.

`--out` *Path*

The location for the output file. The default is the current directory.

`[--debug|-d]`

Display diagnostic messages. In addition, you may need to enable INFO messages in the `serviced` log configuration file. For more information, see Enabling serviced debug messages.

`--group-by` *Group*

Group the results. The value of *Group* can be `container`, `service`, or `day`.

`[--no-children|-n]`

Do not export child services.

`[--help|-h]`

Display the help message.

EXAMPLE

```
serviced log export --service zope --service zenhub --group-by day \
  --from 2017.05.01 --to 2017.05.31 --out /tmp
```

serviced metric

The `serviced metric` command allows you to interact with metrics

USAGE

```
serviced metric [global options] command [command options] [arguments...]
```

COMMANDS

`push`

Push a metric value

`help, h`

Shows a list of commands or help for one command

OPTIONS

`--generate-bash-completion`

`--help, -h`

Show the help message

serviced pool

Use the `serviced pool` command to view and manage Control Center resource pools.

USAGE

```
serviced pool [global options] command [command options] [arguments...]
```

OPTIONS

`[--help|-h]`

Show the help message.

COMMANDS

`list`

List all pools.

`add`

Add a new resource pool.

`[remove|rm]`

Remove an existing resource pool.

`list-ips`

Lists the IP addresses for a resource pool.

`add-virtual-ip`

Add a virtual IP address to a resource pool.

`remove-virtual-ip`

Remove a virtual IP address from a resource pool.

`set-conn-timeout`

Set a connection timeout for a high-latency resource pool (for example, 5m, 2h, 6.6s).

`set-permission`

Set permission flags for hosts in a resource pool.

`[help|h]`

Show a list of commands or the help for a single command.

serviced pool set-conn-timeout

The `serviced pool set-conn-timeout` command sets the length of time the scheduler waits for a disconnected delegate to rejoin its pool before moving the services scheduled for the delegate to a different host in the pool.

Syntax:

```
serviced pool set-conn-timeout POOLID TIMEOUT
```

The TIMEOUT value is specified with an integer followed by the units identifier. This command accepts the following units identifiers:

`ms`, milliseconds

`s`, seconds

`m`, minutes

`h`, hours

For example, `50ms` or `2m`.

serviced pool set-permission

The `serviced pool set-permission` command sets permission flags for hosts in the resource pool, POOLID. Before removing access to administrative functions (`--admin`), remove DFS access.

Syntax:

```
serviced pool set-permission [--dfs[=false]][--admin[=false]] POOLID
```

Command options:

`--dfs[=false]`

Add or remove permission to mount the DFS. The default value is true.

`--admin[=false]`

Add or remove permission to perform administrative functions DFS. The default value is true.

EXAMPLES

Give a resource pool distributed file system (DFS) access:

```
serviced pool set-permission --dfs pool_01_140620
```

Give a resource pool access to administrative functions:

```
serviced pool set-permission --admin pool_01_140620
```

Remove DFS access permission from a resource pool. Note: If you need to remove access to administrative functions, first remove DFS access.

```
serviced pool set-permission --dfs=false pool_01_140620
```

Remove resource pool access to administrative functions. Note: If you need to remove access to administrative functions, first remove DFS access.

```
serviced pool set-permission --admin=false pool_01_140620
```

Set the connection timeout value to 3 minutes:

```
serviced pool set-conn-timeout pool_01_140620 3m
```

serviced restore

The `serviced restore command` restores an instance of an application from a backup file on the same system. You can also duplicate your instance on a similar deployment for testing or failover purposes by restoring a backup file to a new, similarly configured deployment.

If you are restoring from a backup that was taken on another system, copy the backup archive file to the target system.

USAGE

```
serviced restore [command options] [arguments...]
```

OPTIONS

`[--help|-h]`

Show the help message.

serviced script

The `serviced script` command verifies or performs the commands in a script file.

USAGE

A script file is a text file that contains commands to automate common or repetitive tasks and tasks that might require specific services or conditions.

The `serviced script` command supports the following subcommands:

`help`

Display the help message.

`parse`

Verify the syntax of a script file.

`run`

Perform the commands in a script file.

The correct invocation of `serviced script run` depends on whether the REQUIRE_SVC command is present in a script file.

- If a script file does not include REQUIRE_SVC, no additional parameters are required. For example:

  ```
  serviced script run task1.txt
  ```

- If a script file includes REQUIRE_SVC, the `--service` parameter is required. For example:

  ```
  serviced script run task2.txt --service Zenoss.resmgr
  ```

The log file of a `serviced script run` invocation is `/var/log/serviced/script-TIMESTAMP-$USER.log`.

To commit a container, a `serviced script run` invocation must be performed on the Control Center master host.

SYNTAX

The script file syntax rules are as follows:

- Lines that contain no text and lines that start with the number sign character (#) are ignored.
- Lines are terminated with LF or CR+LF.
- A command and its arguments cannot span lines.
- The maximum number of characters per line (command and arguments) is 300000.
- Unless otherwise noted, all command arguments are treated as strings.

COMMANDS

Commands are performed in the order in which they occur in a script. Scripts terminate on completion and when a command returns an exit code other than zero.

`DESCRIPTION` *argument* ...

A statement about the script.

Scripts may contain one or zero DESCRIPTION commands. At least one argument is required.

`VERSION` *argument*

A revision reference for the script.

Scripts may contain one or zero VERSION commands. Only one argument can be used.

`REQUIRE_SVC`

The script needs a reference service in order to perform some or all of its tasks. The service is specified with the `--service` parameter of the `serviced script run` command.

Scripts may contain one or zero REQUIRE_SVC commands.

`SNAPSHOT`

Perform a snapshot. If a script command fails, `serviced` rolls back to the most recent snapshot.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SNAPSHOT commands.

`SVC_USE` *Image-ID*

Use the specified image for script commands that occur after this SVC_USE command. If your application uses multiple images, enter additional SVC_USE commands to specify each image. If the specified image is not present in the local Docker registry, `serviced` attempts to pull it from Docker Hub.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_USE commands. Only one argument can be used.

`SVC_RUN` *Service Run-Command arguments*

Invoke one of the pre-defined commands associated with a service.

Service must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_RUN commands. Multiple arguments can be used.

`SVC_EXEC [COMMIT|NO_COMMIT]` *Service argument* ...

Start a new container to run arbitrary commands. (Equivalent to a non-interactive invocation of `serviced service shell`.)

When COMMIT is specified, changes are committed on successful completion of the commands in *argument*. When NO_COMMIT is specified, changes are not committed.

Service must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_EXEC commands.

`SVC_START {auto|recurse}` *Service*

Start a new instance of *Service*.

If `auto` or `recurse` is not specified, all configured instances of *Service* are started. If `auto` or `recurse` is specified, all configured instances of *Service* and all of its child services are started.

Service must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_START commands.

`SVC_STOP {auto|recurse}` *Service*

Stop the specified service.

If `auto` or `recurse` is not specified, all instances of *Service* are stopped. If `auto` or `recurse` is specified, all instances of *Service* and all of its child services are stopped.

Service must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_STOP commands.

`SVC_RESTART {auto|recurse}` *Service*

Restart the specified service.

If `auto` or `recurse` is not specified, all instances of *Service* are stopped. If `auto` or `recurse` is specified, all instances of *Service* and all of its child services are stopped.

Service must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_RESTART commands.

`SVC_WAIT` *Service* `... [started|stopped|paused]` *Duration*

Pause *Duration* seconds, or pause until the specified service or services reach the started, stopped, or paused state. If the state is not reached when *Duration* expires, the command fails.

Duration must be an integer.

Each *Service* must be the absolute path of a service, with each service in the path separated by the solidus character (/). For example, `Zenoss.resmgr/Zope`.

The REQUIRE_SVC command must be present in the script.

Scripts may contain multiple SVC_WAIT commands.

serviced service

The `serviced service` command lets you manage an application's individual services.

Use this command to perform administrative actions on a specific service or view service status information.

USAGE

```
serviced service [global options] command [command options] [arguments...]
```

COMMANDS

`list`

List all services.

`status`

Display the status of deployed services.

The status `emergency-stopped` indicates that Control Center performed an emergency shutdown of the service due to low storage. Before you can restart the service, resolve the space issue, and then use the `clear-emergency` command to remove the `emergency-shutdown` flags. For more information, see Emergency shutdown of services.

`add`

Add a service.

`clear-emergency`

Reset the `emergency-shutdown` flag for a service that was shut down due to a low-storage condition.

`clone`

Clone a service.

`[remove|rm]`

Remove a service.

`edit`

Edit a service in a text editor.

`assign-ip`

Assign an IP address to service endpoints that require an explicit IP address.

`start`

Start one or more services.

`restart`

Restart one or more services.

`stop`

Stop one or more services.

`shell`

Start a service instance.

`run`

Run a service command in a service instance.

`attach`

Run an arbitrary command in a running service container.

`action`

Run a predefined action in a running service container.

`logs`

Display the log contents for a running service container by calling `docker logs`.

`list-snapshots`

List all snapshots of a service.

`snapshot`

Take a snapshot of a service.

`endpoints`

List the endpoints that are defined for a service.

`public-endpoints`

Manage public endpoints for a service.

`[help|h]`

Show a list of commands or the help for a single command.

GLOBAL OPTIONS

`[--help|-h]`

Shows the help for an option.

serviced service [start|stop]

By default, Control Center schedules services in the background to start, and stop. The asynchronous scheduling improves the speed of these operations, especially in large-scale installations. However, if you use a script that depends on synchronous scheduling that was used in earlier versions of Control Center, specify the command line option `--sync`.

Syntax:

```
serviced service [start|stop] [-s|--sync] ServiceID
```

Command option:

`[--sync|-s]`

Schedules services synchronously. Specify this flag if a script expects the `serviced service [start|stop]` command to wait to return until the service operation has been scheduled. If this flag is not specified, services are scheduled asynchronously, in the background.

serviced service restart

By default, Control Center schedules services in the background to restart to improve the speed of the operation, especially in large-scale installations. However, if you use a script that depends on synchronous scheduling that was used in earlier versions of Control Center, specify the command line option `--sync`.

Syntax:

```
serviced service restart [commandOptions] {ServiceID|instanceID}
```

Command options:

`--auto-launch`

Recursively schedules child services.

`[--sync|-s]`

Schedules services synchronously. Specify this flag if a script expects the `serviced service restart` command to wait to return until the service operation has been scheduled. If this flag is not specified, services are scheduled asynchronously, in the background.

`--rebalance`

Stops all instances of a service before restarting, instead of performing a serial restart of multi-instance services.

serviced snapshot

The `serviced snapshot` command administers environment snapshots.

Use this command to create a short-term save point of a system.

USAGE

```
serviced snapshot [global options] command [command options] [arguments...]
```

COMMANDS

`list`

List all snapshots.

`add`

Take a snapshot of an existing service.

`[remove|rm]`

Remove an existing snapshot.

`commit`

Take a snapshot of and commit a given service instance.

`rollback`

Restore the environment to the state of the given snapshot.

`tag`

Tag an existing snapshot with TAG-NAME.

`untag`

Remove a tag from an existing snapshot.

`[help|h]`

Show a list of commands or the help for a single command.

OPTIONS

`[--help|-h]`

Show the help message.

serviced-storage

The `serviced-storage` command manages Control Center storage.

Use this command to create LVM thin pools for Docker and Control Center.

USAGE

```
serviced-storage [GlobalOptions] Command [CommandOptions]
```

GLOBAL OPTIONS

`[--help|-h]`

Show the help message.

`-o` *DeviceMapperOption=Value*

Specify a device mapper option. Applies only to device mapper drivers.

`-v`

Display verbose logging.

COMMANDS

`check`

Check for orphaned devices.

`create`

Create a volume on a driver.

`create-thin-pool`

Create an LVM thin pool.

`disable`

Disable a driver.

`init`

Initialize a driver.

`list`

Print volumes on a driver.

`mount`

Mount an existing volume from a driver.

`remove`

Remove an existing volume from a driver.

`resize`

Resize an existing volume.

`set`

Set the default driver.

`status`

Print the driver status

`sync`

Sync data from a volume to another volume.

`unset`

Unset the default driver.

`version`

Print the version and exit.

serviced-storage check

The `serviced-storage check` command searches for orphaned snapshot devices in the `serviced` application data thin pool and removes them, if requested. This command requires the path of serviced tenant volumes, which is determined by the SERVICED_VOLUMES_PATH variable in /etc/default /serviced. The default path is /opt/serviced/var/volumes.

Syntax:

```
serviced-storage [GlobalOptions] check [-c|--clean] Path
```

Command options:

`[-c|–clean]`

Remove orphaned snapshot devices.

serviced-storage create-thin-pool

The `serviced-storage create-thin-pool` command creates an LVM thin pool either for Docker data or for Control Center application data. When devices are specified, the command creates an LVM volume group.

Syntax:

```
serviced-storage [GlobalOptions] create-thin-pool \
  [-s|--size]=[Value][G|%] [docker|serviced] \
  [DevicePath [DevicePath...]|VolumeGroupName]
```

Command options:

`[-s|–size]=[Value][G|%]`

The size of the thin pool to create. The size can be a fixed value (in gigabytes) or a relative value (a percentage) of the available storage. When this option is not used, the thin pool size defaults to 90% of the specified storage resource.

serviced-storage resize

The `serviced-storage resize` command increases the size of a `serviced` tenant device in its LVM thin pool. Like LVM thin pools, the size of a `serviced` tenant device can never decrease.

Stop the `serviced` daemon before running `serviced-storage resize`.

Syntax:

```
serviced-storage [GlobalOptions] resize [-d|--driver]=Value TenantID NewSize
```

Command options:

`[-d|–driver]=Value`

The path of the tenant volume.

EXAMPLES

Create an LVM volume group named zenoss and use it for both thin pools:

```
vgcreate zenoss /dev/sdb /dev/sdc
serviced-storage create-thin-pool --size=50G docker zenoss
serviced-storage create-thin-pool --size=50% serviced zenoss
```

If you specify devices or partitions, serviced-storage creates an LVM volume group with the same name as the thin pool. The following example yields the same result as the previous, except the name of the volume group is docker instead of zenoss:

```
serviced-storage create-thin-pool docker /dev/sdb /dev/sdc
serviced-storage create-thin-pool serviced docker
```

Create thin pools on separate block devices:

```
serviced-storage create-thin-pool docker /dev/sdb
serviced-storage create-thin-pool serviced /dev/sdc
```

Create thin pools on separate partitions:

```
serviced-storage create-thin-pool docker /dev/sdb1
serviced-storage create-thin-pool serviced /dev/sdc3
```

Increase the size of the serviced LVM thin pool, and then increase the size of a serviced tenant device.

```
lvextend -L+300G zenoss/serviced-pool
systemctl stop serviced
serviced-storage -o dm.thinpooldev=/dev/mapper/zenoss-serviced--pool \
  resize -d /opt/serviced/var/volumes 58uuetj38draeu9alp6002b1y 200G
systemctl start serviced
```

Identify the serviced application data thin pool, and then remove orphaned snapshot devices.

```
ls /dev/mapper | grep serviced
serviced-storage -o dm.thinpooldev=/dev/mapper/zenoss-serviced--pool \
  check -c /opt/serviced/var/volumes
```

serviced template

The `serviced metric` command allows you to administer templates

USAGE

```
serviced template [global options] command [command options] [arguments...]
```

COMMANDS

`list`

List all templates

`add`

Add a new template

`remove, rm`

Remove an existing template

`compile`

Convert a directory of service definitions into a template

`help, h`

Shows a list of commands or help for one command

OPTIONS

`--generate-bash-completion`

`--help, -h`

Show the help message

serviced version

The `serviced version` command displays the version information for Control Center and Go

USAGE

```
serviced version
```

Administering Control Center

This section contains information about and procedures for performing administrative tasks in Control Center.

Control Center application data storage requirements

Control Center uses an LVM thin pool to store tenant (application) data. LVM thin pools include separate storage areas for data and for metadata. For each tenant it manages, Control Center maintains a separate virtual device (a volume) in the data storage area of its LVM thin pool. Also, Control Center creates virtual devices in the data storage area for snapshots of tenant data.

To ensure consistency, Control Center requires the following, minimum amount of free space in its thin pool:

- 3GiB available for each tenant volume
- 3GiB available in the data storage area
- 62MiB available in the metadata storage area
- The total amount of metadata storage must be 1% of total data storage

In addition, the ratio of application data stored in the thin pool to the total amount of storage in the pool should always be 50%. When the amount of available space is less than double the amount of total storage, the snapshot feature is prone to failure.

Examining application data storage status

Beginning with release 1.3.0, Control Center initiates an emergency shutdown when the minimum required amounts of free space are not available in the `serviced` thin pool or tenant volumes.

Use this procedure to display the amount of free space in a Control Center thin pool and tenant volumes, to determine how much space is available in the LVM volume group that contains the thin pool, and to determine whether additional steps are required.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Display the amount of space available in the `serviced` thin pool.

   ```
   serviced volume status
   ```

   Example output with highlighted values:

   ```
   Status for volume /opt/serviced/var/volumes:

   Driver:      devicemapper
   Driver Type: direct-lvm
   Volume Path: /opt/serviced/var/volumes

   Thin Pool
   ---------
   Logical Volume:              serviced-serviced--pool
   Metadata (total/used/avail): 2.2 GiB / 64.18 MiB (2.85%) / 2.137 GiB (97.15%)
   Data (total/used/avail):     200 GiB   / 30.576 GiB (15%)   / 169.424 GiB (85%)

   3ir81rg1h8b09ipowynz3qx2f Application Data
   ------------------------------------------
   Volume Mount Point:           /opt/serviced/var/volumes/3ir81rg1h8b09ipowynz3qx2f
   Filesystem (total/used/avail): 98.31 GiB / 1.511 GiB (1.5%) / 91.78 GiB (93%)
   Virtual device size:          100 GiB
   ```

   The result includes detailed information about the `serviced` thin pool and each tenant volume.
   - If the amount of free space in the `serviced` thin pool is sufficient, stop. No further action is required.
   - If the amount of free space in the data or metadata portions of the `serviced` thin pool is not sufficient, perform the following steps.
3. Identify the volume group to which the `serviced` thin pool belongs.

   ```
   lvs --options=lv_name,vg_name,lv_size
   ```

   The volume group associated with `serviced-pool` contains the `serviced` thin pool.
4. Display the amount of free space in the volume group that contains the `serviced` thin pool.
   Replace *Volume-Group* with the name of the volume group identified in the previous step:

   ```
   vgs --no-headings --options=vg_free Volume-Group
   ```

   - If the amount of free space in the volume group is not sufficient to increase one or both of the storage areas of the `serviced` thin pool to their required minimums, add physical or logical storage to the volume group. For more information, refer to your operating system documentation.
   - If the amount of free space in the volume group is sufficient to increase one or both of the storage areas of the `serviced` thin pool to their required minimums, proceed to the next step.
5. Increase the free space in one or both areas of the `serviced` thin pool.
   - To increase the amount of space in the metadata area, proceed to Adding space to the metadata area of a Control Center thin pool.
   - To increase the amount of space in the data area, proceed to Adding space to the data area of a Control Center thin pool.

Adding space to the metadata area of a Control Center thin pool

Use this procedure to increase the amount of space in the metadata area of a Control Center thin pool.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Display information about LVM logical volumes on the host.

   ```
   lvs --options=lv_name,vg_name,lv_size
   ```

   Typically, the logical volume is `serviced-pool` and the containing volume group is `serviced`.
3. Add space to the metadata storage area of the serviced thin pool.
   In the following command:
   - Replace *Size* with the amount of space to add (in megabytes) and the units identifier (M).
   - Replace *Volume-Group* with the name of the LVM volume group identified in the previous step.
   - Replace *Logical-Volume* with the name of the logical volume identified in the previous step.

   ```
   lvextend -L+SizeM Volume-Group/Logical-Volume_tmeta
   ```

Adding space to the data area of a Control Center thin pool

Use this procedure to increase the amount of space in the data area of a Control Center thin pool.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Display information about LVM logical volumes on the host.

   ```
   lvs --options=lv_name,vg_name,lv_size
   ```

   Typically, the logical volume is `serviced-pool` and the containing volume group is `serviced`.
3. Add space to the data storage area of the `serviced` thin pool.
   In the following command:
   - Replace *Total-Size* with the sum of the existing device size plus the space to add to the device, in gigabytes. Include the units identifier, G.
   - Replace *Volume-Group* with the name of the LVM volume group identified in the previous step.
   - Replace *Logical-Volume* with the name of the logical volume identified in the previous step.

   ```
   lvextend -L+Total-SizeG Volume-Group/Logical-Volume
   ```

About adding space to a tenant volume

The LVM thin pool for application data can be used to store multiple tenant volumes and multiple snapshots of tenant data.

- If you added space to the thin pool to create a new tenant volume, see serviced-storage.
- If you added space to the thin pool to provide additional storage for snapshots, then the tenant volume does not need to be resized.
- If you added space to the thin pool because the tenant devices were oversubscribed, then the tenant volume does not need to be resized. A tenant volume is oversubscribed when its size at creation exceeded the amount of space available in the thin pool.

Adding space to a tenant volume

Use this procedure to increase the size of a tenant volume in a Control Center thin pool.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Identify the tenant device to resize.

   ```
   serviced volume status
   ```

3. Display the device mapper name of the `serviced` thin pool.

   ```
   grep -E '^\b*SERVICED_DM_THINPOOLDEV' /etc/default/serviced | sed -e 's/.*=//'
   ```

   Typically, the name is `/dev/mapper/serviced-serviced--pool`.
4. Increase the size of the tenant device.
   In the following command:
   - Replace *Device-Mapper-Name* with the device mapper name of the thin pool.
   - Replace *Tenant-ID* with the identifier of the tenant device.
   - Replace *Total-Size* with the sum of the existing device size plus the space to add to the device, in gigabytes. Include the units identifier, G.

   ```
   serviced-storage resize -d /opt/serviced/var/volumes \
     -o dm.thinpooldev=Device-Mapper-Name Tenant-ID Total-SizeG
   ```

Emergency shutdown of services

Control Center monitors each service's short-term storage usage trends and current usage levels. When Control Center predicts that a service is about to exhaust storage space, it initiates an automatic emergency shutdown of the service. By shutting down while enough space is available to perform recovery operations, Control Center minimizes the risk of data corruption.

Emergency shutdown is performed for services that are in a resource pool that has DFS access. Services that are in pools that do not have DFS permissions and do not write to the DFS continue running.

Control Center displays thin pool and DFS information in the following graphs in the browser interface:

- On the Applications tab, the Thin Pool Usage graph shows used and available bytes for the thin pool.
- On the Applications page for each service, the DFS Usage graph shows used and available bytes for the DFS.

By comparing usage to the available storage and the amount of space that must be reserved, Control Center determines when a service must be shut down before filling the thin pool or DFS, and initiates the emergency shutdown. The browser interface identifies services in the emergency shutdown state, as does issuing the `serviced service status` command.

To minimize data loss, Control Center shuts down services in the following order: databases; services that cannot be recovered; indices and services that are difficult to recover; any other services. Services in emergency shutdown status cannot be restarted until the underlying cause of the shutdown is resolved.

To resolve an emergency shutdown:

1. Examine the service that was shut down to determine why it was using excessive storage and correct the issue. For example:
    - If an application was writing a large amount of performance data to the tenant device, add space to the device. See Control Center application data storage requirements.
    - If too many snapshots are stored on the device, delete those that you no longer need. See Creating snapshots and rolling back.
    - If a usage anomaly might have occurred, wait for usage levels to return to normal.
2. Clear the emergency shutdown flags.
3. Start the service by using the browser interface or command line interface. Control Center starts services in the reverse order of shutdown.

Resetting emergency shutdown flags

After resolving the issue that caused an emergency shutdown, use this procedure to determine service status, and then restart the service.

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Check services for the emergency shutdown flag:

   ```
   for s in $(serviced service list --show-fields ServiceID \
     | grep -v ServiceID); do serviced service list $s \
     | grep EmergencyShutdown\"; done
   ```

   "True" indicates an emergency shutdown.
3. Clear the emergency-shutdown flag for a service or the entire application:

   ```
   serviced service clear-emergency SERVICEID
   ```

   ```
   serviced service clear-emergency APPLICATION_ID
   ```

4. Check services for the emergency shutdown flag:

   ```
   for s in $(serviced service list --show-fields ServiceID \
     | grep -v ServiceID); do serviced service list $s \
     | grep EmergencyShutdown\"; done
   ```

   "False" indicates that the flags have been cleared.
5. Start the service:

   ```
   serviced service start SERVICEID
   ```

Using Control Center with a NAT device

Network address translation (NAT) enables one device (a router, switch, or firewall) to connect a local area network with the internet and outside devices. The NAT device forwards traffic to the intended host, and serves as a firewall to systems that are behind the device, making them inaccessible from outside the network.

In a Control Center system without a NAT device, the master host connects directly to the delegate host's address:rpcport and requests host information.

When the Control Center master host is outside the network, it can connect to the NAT device, but cannot access delegate hosts behind the device because they have private IP addresses. The NAT device forwards its port to the delegate hosts' address:rpcport.
When you add a delegate host by using either Control Center interface (browser or command-line) you must specify the hostname or IP address and port for the NAT device. After you add delegate hosts, you must transfer host keys to the delegate hosts and register them.

The Control Center master host always attempts registration on port 22. If the NAT device forwards port 22 to the delegate host that you are registering, you can remotely register the keys.

```
serviced host add Hostname-Or-IP:Host-Port \
  --nat-address==NAT-Hostname-Or-IP:NAT-Port \
  [--register]
```

If you have two delegates behind the NAT device, change the port forwarding for port 22 between adding the hosts, or transfer the keys file manually to and register from each delegate.
When resetting keys, the CLI supports the --nat-address argument. If the delegate is behind a NAT device and port 22 is forwarded to that delegate, you can attempt to register the delegate when resetting the key:

```
serviced key reset Hostname-Or-IP:Host-Port --register \
  --nat-address==NAT-Hostname-Or-IP:NAT-Port
```

Example: Adding delegate hosts to a resource pool

The master host is outside the network. Delegate hosts delegate1 and delegate2 are behind a NAT router. IP address and port information is as follow:

- NAT router: 192.0.2.0
- Delegate1: 198.51.100.0:4979
- Delegate2: 203.0.113.0:4979

The router forwards port 4979 to delegate1's RPC port (4979):

```
serviced host add 198.51.100.0:4979 Resource-Pool \
  --nat-address=192.0.2.0:4979
```

The router forwards port 4980 to delegate2's RPC port (4979):

```
serviced host add 203.0.113.0:4979 Resource-Pool \
  --nat-address=192.0.2.0:4980
```

Security considerations for using Control Center with a NAT device

To attach to a service on a delegates behind the NAT device, you must use ssh to access the delegate. From the delegate host, run serviced service attach . For security reasons, you cannot use serviced service attach from the master to connect to a delegate.

In the Control Center browser interface, for security reasons, you cannot drill down to a service that is running on a delegate behind a NAT device and click Container Log for the instance

Backing up and restoring

Use Control Center to back up applications that Control Center manages.

Having accurate and tested system backups can mitigate problems caused by software or hardware issues. To have a historical archive, you might back up on a regular schedule. Back up as needed when you want to move data from one instance to another or duplicate an instance for testing or failover purposes.

Backups include the current state of the system, the state of all services, configuration information, and application data. The backup process leverages snapshot functionality. Therefore, when a backup is running, you can start and restart services; there is no need to shut down the application or Docker containers. The services are only momentarily suspended to enable reading the data.

You can back up and restore applications by using the browser interface or the command-line interface (CLI). Results are comparable; however, the CLI offers an option to exclude subdirectories from a backup.

The Control Center backup process creates a compressed tar archive file (.tgz) that can be restored on the same deployment or a similar deployment. Before starting a backup, Control Center estimates the size of the backup file and compares it to the amount of free space. If storage space is insufficient, Control Center does not start a backup.

The default directory for backup files is `/opt/serviced/var/backups`, which directory can be changed by specifying the SERVICED_BACKUPS_PATH variable.

You must perform your backups using the documented procedures below. Do not use the backup and restore functionality provided in your vSphere environment as a substitute to these procedures.

When a full backup is not necessary, such as when you need a checkpoint before installing software, you can perform a snapshot of the system. If you need to revert back to a snapshot, use the rollback feature. You use the CLI to perform snapshot and rollback.

With both backup and snapshot, Control Center

- Creates a tag for the Docker image of each service with metadata about the application data.
- Creates a separate snapshot of the LVM thin pool, which stores both application data and snapshots of the application data.

When you create a backup, Control Center also exports the snapshots to an archive file and moves them out of the LVM thin pool. Backups do not affect time-to-live (TTL).

For more information, see Creating snapshots and rolling back and Command-line interface reference.

Best practices for backup and restore

Review considerations and best practices that apply to application backup and restore.

Backup storage

- Ensure that you have enough free space to receive and store backups. Running low on available disk space results in errors and affects system performance.
- Regularly back up the production environment and potentially the system from the initial deployment.
- Store backups on a machine other than the Control Center master. This can be a file server, a separate disaster recovery system, or test environment.
- To provide a historical archive, back up on a regular schedule. Back up as needed when you perform less-frequent tasks such as moving data from one instance to another or duplicating an instance for testing or failover purposes.

Restoring from a backup

- Before upgrading or testing an application, ensure that you have a recent backup that successfully restores.
- Frequently test restoring from a backup to ensure that the backup restores successfully, and that the restored system is an accurate representation of the state of the deployment when the backup was performed.
- You can restore a backup to the system on which it was created or to an alternate system. When restoring a backup from one system to an alternate system, ensure that
    - The alternate system mirrors at least one device from the backed-up system.
    - Services that were added to the alternate system by a previous restore have been manually deleted.
- Restoring from a backup file will remove services that were added after taking the backup. That is, if you create a backup, add a service, and then restore from the backup, the service is deleted as part of the restore process.
- If an outage occurs during a restore from a backup, you can resume the restore because Control Center preserves complete data on the system. For example, if two of six backed up snapshots are restored before an outage, when you resume the restore, those two snapshots are saved on the system, and are not downloaded again.

Version considerations

- Backups that were created using Control Center 1.0.x cannot be restored in Control Center 1.1.x or later.
- In Control Center 1.2.x and later, you can restore backups that were created using Control Center 1.1.x or later.

Backup contents

Control Center backups contain:

- Any applications installed deployed in Control Center
- The Docker images used by that application
- The contents of the Distributed File System (DFS)

Control Center backups do not contain:

- The Control Center audit logs
- The Control Center Elastic database where Resource Pools and their member delegates are defined
- The Control Center configuration file, /etc/default/serviced
- The serviced journal from the Control Center master host or any delegate hosts
- The serviced .rpm or the directory structures created by it (i.e., serviced must be installed before a backup can be restored)

Backing up using the browser interface

Using the browser interface, you can create a backup of your entire system.

1. Log in to the Control Center browser interface.
2. Click the Backup / Restore tab.
3. Click Create Backup.
   Control Center estimates the size of the backup file and displays the amount of free space.
4. Depending on available storage space, proceed as follows:
   - If storage space is adequate, click Create Backup.
   - If storage space is insufficient, take action to increase available space, and then try the backup again.

After a successful backup, the system displays the name of the backup file.

Backing up using the CLI

As an alternative to performing application backup from the Control Center browser interface, you can use the command-line interface (CLI).

Using the CLI, you can back up your entire system or specify one or more tenant volumes to exclude from the backup. For example, to save resources, you might exclude the tenant volume that contains performance data, hbase-master. If you want to create a backup to restore on another system, you might exclude the tenant volumes for the events database and index, mariadb-events and zeneventserver.

Before starting a backup, Control Center estimates the size of the backup file and compares it to the amount of free space. If storage space is insufficient, Control Center does not start a backup. Exit code 1 indicates insufficient space. If storage space is insufficient, take action to increase available space, and then try the backup again.

The default directory for backup files is `/opt/serviced/var/backups`, which directory can be changed by specifying the SERVICED_BACKUPS_PATH variable.

Backing up the entire system

To back up the entire system, follow these steps:

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Start the backup:

   ```
   serviced backup /opt/serviced/var/backups
   ```

   After a successful backup, the system displays the name of the backup file.

   Example result:

   ```
   backup-2017-03-07-203717.tgz
   ```

Exclude one tenant volume from the backup:

By default, Control Center stores application data in /opt/serviced/var/volumes, which can be changed by specifying the SERVICED_VOLUMES_PATH variable.

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Display the Control Center tenant identifier.
   If necessary, replace `/opt/serviced/var/volumes` with your path.

   ```
   ls /opt/serviced/var/volume
   ```

   Example result:

   ```
   cvs0ul2tmvjcitolrm7p0d8bx
   ```

3. Display the directories under the Control Center tenant identifier.
   Replace Tenant-ID with the identifier displayed in the previous step.

   ```
   ls /opt/serviced/var/volumes/Tenant-ID
   ```

4. Exclude the tenant volume from the backup.
   For example, exclude `hbase-master`:

   ```
   serviced backup /opt/serviced/var/backups --exclude hbase-master
   ```

If you use automated backups, edit the scripts to exclude tenant volumes.

Exclude multiple tenant volumes from the backup:

By default, Control Center stores application data in `/opt/serviced/var/volumes`. The directory can be changed by specifying the SERVICED_VOLUMES_PATH environment variable in the Control Center configuration file, `/etc/default/serviced`.

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Display the Control Center tenant identifier.
   If necessary, replace `/opt/serviced/var/volumes` with your path.

   ```
   ls /opt/serviced/var/volumes
   ```

   Example result:

   ```
   cvs0ul2tmvjcitolrm7p0d8bx
   ```

3. Display the directories under the Control Center tenant identifier.
   Replace Tenant-ID with the identifier displayed in the previous step.

   ```
   ls /opt/serviced/var/volumes/Tenant-ID
   ```

4. Exclude multiple tenant volumes from the backup.
   For example, exclude `mariadb-events` and `zeneventserver`.

   ```
   serviced backup /opt/serviced/var/backups --exclude mariadb-events --exclude zeneventserver
   ```

Restoring from a backup

Restore an instance of an application from a backup file on the same system, or restore from a backup file to duplicate an instance on a new, similarly configured deployment.

If you are restoring from a backup that was created on another system, copy the backup archive file to the target system.

Restoring using the browser interface:

1. Log in to the Control Center browser interface.
2. In the Applications table, identify the name of the application instance.
3. Stop the instance and verify that its subservices are stopped.
    a. In the Actions column of the Applications table, click Stop.
    b. In the Stop Service dialog box, click Stop Service and Children.
    c. In the Applications column of the Applications table, click the name of the stopped instance, and then scroll down to the Services table to verify that all services are stopped.

   Because snapshots are loaded to disk, during a restore you are not *required* to stop services while the file is loaded. Though the restore will not succeed, the snapshot and images are available for you to manually rollback each application.
4. Click the Backup / Restore tab.
5. Beside the backup file that you want to use to restore your application, click Restore Backup. Confirm your selection by clicking Restore.
6. When the restore is finished, click the Applications tab, then click Start beside the instance you just restored.
7. Review and if necessary, define IP assignments.
    a. Click Applications and then click the application instance.
    b. Review the IP Assignments table. If all services have an IP assignment, no action is required.
    c. For any service that does not have an automatic IP assignment, click Assign, choose an IP, and then click Assign IP.

Restoring using the CLI:

1. Log in to the CLI of the Control Center master as root or another users with serviced permissions.
2. Identify the name of the application instance to be restored with the `serviced service status` command.
3. Stop the service if it is running with the `serviced service stop $SERVICEDID` command. In the case of the Resource Manager application, the command would be `serviced service stop Zenoss.resmgr`.
4. Restore the backup file with the `serviced restore` command. For a backup file named `backup-2019-01-29-232324.tgz` located in the default /opt/serviced/var/backups/ directory, the command would be `serviced restore /opt/serviced/var/backups/backup-2019-01-29-232324.tgz`.
5. A successful restore will print nothing to stdout. Detailed messaging can be found in the serviced journal on the Control Center master. The format in that case should look like the following:

   ```
   time="2019-01-30T16:33:49Z" level=info msg="Completed Restoring from Backup" action=restore backupfile="/opt
   /serviced/var/backups/backup-2019-01-29-232324.tgz" elapsed=223.894515sec
   ```
6. Start the service with the `serviced service start $SERVICEID` command. In the case of the Resource Manager application, the command would be `serviced service start Zenoss.resmgr.`

Tenant device states

In some circumstances, probably after a restore, serviced switches tenant devices but is unable to remove the previous device from the NFS export mount. When serviced is in this state, the master host and the delegates are using different devices, and the two devices quickly get out of sync.

To prevent errors, serviced does not start services when the tenant mount and export mount are backed by different devices, and the services are assigned to a resource pool that has DFS access permissions. When tenant mount and export mount are backed by different devices, and services are assigned to a resource pool that does not have DFS access permissions, or when the tenant mount and export mount are backed by the same device, serv iced does start services.

Creating snapshots and rolling back

Though backups are the most reliable and durable way to preserve Docker images and configurations, creating a backup of an entire application is not always practical. However, you need to safeguard against potential risk when changing the system. In these cases, you can create a snapshot of the system.

Snapshot functionality provides a time-efficient and space-efficient method of copying data. Create a snapshot whenever you need a save point for Docker images, such as before committing container changes.

With both snapshot and backup, Control Center

- Creates a tag for the Docker image of each service with metadata about the application data.
- Creates a separate snapshot of the LVM thin pool, which stores both application data and snapshots of the application data.

Snapshots are intended to serve as short-term save points only, and therefore have a default time-to-live (TTL) value of 12 hours. If you need to keep a snapshot beyond the TTL, tag the snapshot to prevent it from being deleted after the TTL expires. For historical backups of data that you need to save long-term, create full backups instead of snapshots.

You can use the rollback functionality to go back to a snapshot image. For example, roll back if changes to an application cause a failure or other degradation. Rolling back returns the application and distributed file system to the state that existed at the time of the snapshot.

Note: Rolling back from a snapshot does not remove services that you added after creating the snapshot. That is, if you create a snapshot, add a service, and then roll back, the service remains on the system; it is not deleted as part of the roll back.

Control Center uses *thin provisioning*, which enables it to create snapshots of the application data volume. Thin provisioning is a virtualization method that allocates data blocks only when data is written (copy-on-write).

Because snapshots track changes to the file system over time, their space requirements expand incrementally as application data changes. Application data and snapshots share the same base device; therefore, ensure that snapshots do not fill up the base device storage. For information about extending storage, see Control Center application data storage requirements.

Creating a snapshot

1. Log in to the Control Center host as a user with `serviced` CLI privileges.
2. Find the identifier of the service; for example, Zenoss.resmgr.
   ```
   serviced service list
   ```
3. Create the snapshot.
   Replace SERVICEID with the identifier of the service.
   ```
   serviced snapshot add SERVICEID
   ```
4. Verify the existence of the snapshot.
   ```
   serviced snapshot list
   ```
5. To keep the snapshot for longer than the default 12-hour TTL, tag it.
   Replace SNAPSHOTID with the identifier of your snapshot and TAG-NAME with your text.
   ```
   serviced snapshot tag SNAPSHOTID TAG-NAME
   ```
6. To make a snapshot subject to the TTL value, untag it.
   Replace SNAPSHOTID with the identifier of your snapshot and TAG-NAME with your text.
   ```
   serviced snapshot untag SNAPSHOTID TAG-NAME
   ```

Rolling back to a snapshot

Before rolling back, you must stop services that are used in the snapshot image. The following procedure includes this step.

1. Log in to the Control Center host as a user with `serviced` CLI privileges.
2. To roll back to a snapshot, you must find the identifier of the snapshot.
   ```
   serviced snapshot list
   ```
3. Roll back to the snapshot.
   Replace SNAPSHOTID with the identifier of your snapshot. The `--force-restart` flag automatically stops the affected services before rollback and starts them after completion.
   ```
   serviced snapshot rollback SNAPSHOTID --force-restart
   ```

Rolling restart of services

To reduce or eliminate downtime for services with multiple instances, Control Center restarts instances of the service one at a time.

The Control Center browser interface visually indicates when a service is restarting and whether an instance is down during the restart. When a wide area network (WAN) outage occurs, the rolling restart proceeds, with instances on the disconnected hosts restarting when the WAN is restored.

The serial process applies to *restart* only. When stopping and starting services, Control Center starts and stops all instances immediately.

Optionally, you can specify that Control Center is to stop all instances of a service before restarting.

Changing rolling restart

Use this procedure to stop all instances of a service before restarting, instead of performing a rolling restart.

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Restart all instances of the service with the `--rebalance` option.
   Replace *SERVICE* with the name of the service to restart:

```
for service in $(serviced service list --show-fields ServiceID,Name | awk '/SERVICE/ {print $1'})
do
  serviced service restart --rebalance $service
done
```

Control Center audit logging

The `serviced` service writes messages to an audit log file on the master host when configuration changes occur on Control Center hosts. The messages record the time, user identity, and information about the change in plain text.

The default location of the `serviced` audit log file is `/var/log/serviced`. The location is determined by the SERVICED_LOG_PATH variable in `/etc/default/serviced`. The log file name is `serviced-audit.log`.

The `serviced` audit log directory contains additional files:

- `serviced.access.log` records HTTP/S requests and is always present.
- `application-audit.log` records application audit messages, and is present only if Zenoss Resource Manager is installed.

The files in the `serviced` audit log directory are managed by `logrotate`. The serviced RPM installation process installs `logrotate`, if necessary, and creates `/etc/cron.hourly/serviced`. Then, the `anacron` service invokes `logrotate` every hour.

The operations that `logrotate` performs on audit log files are specified in `/opt/serviced/etc/logrotate.conf`. The default configuration rotates, compresses, and removes files as necessary to ensure that the logs occupy no more than 10GB of storage. To store larger volumes of log files, choose one or more of the following options:

- Mount the `serviced` audit log directory on a larger local or remote file system.
- Modify the `logrotate` configuration file.
- Forward the log files to a log management application.
- Use a `cron` job to copy the files to a larger local or remote file system.

Rotating container log files

When a Control Center internal service has to stop unexpectedly, it writes the last 1000 lines of its log data to a file named `/tmp/Docker-ID.container.log`. Over time, the log files could fill the `/tmp` area. Zenoss recommends implementing a log file rotation strategy to avoid filling `/tmp`.

Configuring a private master NTP server

Control Center requires a common time source. The procedures in this section configure a private master NTP server to synchronize the system clocks of all Control Center hosts.

A private master server is only required when a multi-host deployment does not have internet access and no time synchronization servers are available behind the firewall. Single-host deployments do not require time synchronization, and deployments with internet access can rely on the default public time servers that are configured in /etc/ntp.conf.

VMware vSphere guest systems can synchronize their system clocks with the host system. If that feature is enabled, it must be disabled to configure a private master NTP server or to use a time synchronization server that is available behind the firewall. For more information, refer to the VMware documentation for your version of vSphere.

The following procedures assume that Control Center is not installed. If it is installed, stop the serviced service before configuring NTP.

Configuring an NTP master server

Use this procedure to configure an NTP master server on the Control Center master host. Perform this procedure only if the host does not have internet access.

On VMware vSphere guests, before performing this procedure, disable time synchronization between guest and host operating systems.

1. Log in to the Control Center master host as root, or as a user with superuser privileges.
2. Create a backup of the NTP configuration file.

   ```
   cp -p /etc/ntp.conf /etc/ntp.conf.orig
   ```

3. Edit the NTP configuration file.
   a. Open /etc/ntp.conf with a text editor.
   b. Replace all of the lines in the file with the following lines:

      ```
      # Use the local clock
      server 127.127.1.0 prefer
      fudge  127.127.1.0 stratum 10
      driftfile /var/lib/ntp/drift
      broadcastdelay 0.008

      # Give localhost full access rights
      restrict 127.0.0.1

      # Grant access to client hosts
      restrict Address-Range mask Netmask nomodify notrap
      ```

   c. Replace Address-Range with the range of IPv4 network addresses that are allowed to query this NTP server.
      For example, the following IP addresses are assigned to Control Center hosts:
      - 203.0.113.10
      - 203.0.113.11
      - 203.0.113.12
      - 203.0.113.13

      For the preceding addresses, the value for Address-Range is 203.0.113.0.
   d. Replace Netmask with the IPv4 network mask that corresponds with the address range.
      For example, a valid network mask for 203.0.113.0 is 255.255.255.0.
   e. Save the file and exit the editor.
4. Stop Control Center.

   ```
   systemctl stop serviced
   ```

5. Enable and start the NTP daemon.
   a. Enable the ntpd daemon.

      ```
      systemctl enable ntpd
      ```

   b. Configure ntpd to start when the system starts.
      Currently, an unresolved issue associated with NTP prevents ntpd from restarting correctly after a reboot, and 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
      ```

   c. Start ntpd.

      ```
      systemctl start ntpd
      ```

6. Start Control Center.

   ```
   systemctl start serviced
   ```

Configuring NTP clients

Use this procedure to configure a delegate hosts to synchronize its clocks with the NTP server on the Control Center master host. Perform this procedure only if the hosts do not have internet access. Repeat this procedure on each Control Center delegate host.

On VMware vSphere guests, before performing this procedure, disable time synchronization between guest and host operating systems.

1. Log in to the Control Center delegate host as root, or as a user with superuser privileges.
2. Create a backup of the NTP configuration file.

   ```
   cp -p /etc/ntp.conf /etc/ntp.conf.orig
   ```

3. Edit the NTP configuration file.
   a. Open /etc/ntp.conf with a text editor.
   b. Replace all of the lines in the file with the following lines:

      ```
      # Point to the master time server
      server Master-Address

      restrict default ignore
      restrict 127.0.0.1
      restrict Master-Address mask 255.255.255.255 nomodify notrap noquery

      driftfile /var/lib/ntp/drift
      ```

   c. Replace both instances of Master-Address with the IPv4 address of the host where the NTP server is running (the Control Center master host).
   d. Save the file and exit the editor.
4. Stop Control Center.

   ```
   systemctl stop serviced
   ```

5. Synchronize the clock with the master server.

   ```
   ntpd -gq
   ```

6. Enable and start the NTP daemon.
   a. Enable the ntpd daemon.

      ```
      systemctl enable ntpd
      ```

   b. Configure ntpd to start when the system starts.
      Currently, an unresolved issue associated with NTP prevents ntpd from restarting correctly after a reboot, and 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
      ```

   c. Start ntpd.

      ```
      systemctl start ntpd
      ```

7. Start Control Center.

   ```
   systemctl start serviced
   ```

Stopping and starting Control Center

Before performing maintenance, such as operating system upgrades or applying patches, properly stop and start Control Center. This section provides procedures for single-host and multi-host deployments.

Stopping Control Center on the master host

Use this procedure to stop the Control Center service (`serviced`) on the master host.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Stop the top-level service `serviced` is managing, if necessary.
    a. Show the status of running services.
    ```
    serviced service status
    ```
    The top-level service is the service listed immediately below the headings line.
        - If the status of the top-level service and all child services is stopped, proceed to the next step.
        - If the status of the top-level service and all child services is **not** stopped, perform the remaining substeps.
    b. Stop the top-level service.
    ```
    serviced service stop Zenoss.resmgr
    ```
    c. Monitor the stop.
    ```
    serviced service status
    ```
    When the status of the top-level service and all child services is stopped, proceed to the next step.
3. Stop the Control Center service.
    ```
    systemctl stop serviced
    ```
4. Ensure that no containers remain in the local repository.
    a. Display the identifiers of all containers, running and exited.
    ```
    docker ps -qa
    ```
        - If the command returns no result, stop. This procedure is complete.
        - If the command returns a result, perform the following substeps.
    b. Remove all remaining containers.
    ```
    docker ps -qa | xargs --no-run-if-empty docker rm -fv
    ```
    c. Display the identifiers of all containers, running and exited.
    ```
    docker ps -qa
    ```
        - If the command returns no result, stop. This procedure is complete.
        - If the command returns a result, perform the remaining substeps.
    d. Disable the automatic startup of serviced.
    ```
    systemctl disable serviced
    ```
    e. Reboot the host.
    ```
    reboot
    ```
    f. Log in to the master host as root, or as a user with superuser privileges.
    g. Enable the automatic startup of `serviced`.
    ```
    systemctl enable serviced
    ```

Stopping Control Center on a delegate host

Use this procedure to stop the Control Center service (`serviced`) on a delegate host in a multi-host deployment. Repeat this procedure on each delegate host in your deployment.
Before performing this procedure on any delegate host, stop Control Center on the master host.

1. Log in to the delegate host as root, or as a user with superuser privileges.
2. Stop the Control Center service.
```
systemctl stop serviced
```
3. Ensure that no containers remain in the local repository.
   a. Display the identifiers of all containers, running and exited.
```
docker ps -qa
```
      - If the command returns no result, proceed to the next step.
      - If the command returns a result, perform the following substeps.
   b. Remove all remaining containers.
```
docker ps -qa | xargs --no-run-if-empty docker rm -fv
```
      - If the remove command completes, proceed to the next step.
      - If the remove command does not complete, the most likely cause is an NFS conflict. Perform the following substeps.
   c. Stop the NFS and Docker services.
```
systemctl stop nfs && systemctl stop docker
```
   d. Start the NFS and Docker services.
```
systemctl start nfs && systemctl start docker
```
   e. Repeat the attempt to remove all remaining containers.
```
docker ps -qa | xargs --no-run-if-empty docker rm -fv
```
      - If the remove command completes, proceed to the next step.
      - If the remove command does not complete, perform the remaining substeps.
   f. Disable the automatic startup of serviced.
```
systemctl disable serviced
```
   g. Reboot the host.
```
reboot
```
   h. Log in to the delegate host as root, or as a user with superuser privileges.
   i. Enable the automatic startup of serviced.
```
systemctl enable serviced
```
4. Dismount all filesystems mounted from the Control Center master host.
   This step ensures no stale mounts remain when the storage on the master host is replaced.
   a. Identify filesystems mounted from the master host.
```
awk '/serviced/ { print $1, $2 }' < /proc/mounts | grep -v '/opt/serviced/var/isvcs'
```
      - If the preceding command returns no result, stop. This procedure is complete.
      - If the preceding command returns a result, perform the following substeps.
   b. Force the filesystems to dismount.
```
for FS in $(awk '/serviced/ { print $2 }' < /proc/mounts | grep -v '/opt/serviced/var/isvcs')
do
  umount -f $FS
done
```

c. Identify filesystems mounted from the master host.

```
awk '/serviced/ { print $1, $2 }' < /proc/mounts | grep -v '/opt/serviced/var/isvcs'
```

- If the preceding command returns no result, stop. This procedure is complete.
- If the preceding command returns a result, perform the following substeps.

d. Perform a lazy dismount.

```
for FS in $(awk '/serviced/ { print $2 }' < /proc/mounts | grep -v '/opt/serviced/var/isvcs')
do
  umount -f -l $FS
done
```

e. Restart the NFS service.

```
systemctl restart nfs
```

f. Determine whether any filesystems remain mounted from the master host.

```
awk '/serviced/ { print $1, $2 }' < /proc/mounts | grep -v '/opt/serviced/var/isvcs'
```

- If the preceding command returns no result, stop. This procedure is complete.
- If the preceding command returns a result, perform the remaining substeps.

g. Disable the automatic startup of serviced.

```
systemctl disable serviced
```

h. Reboot the host.

```
reboot
```

i. Log in to the delegate host as root, or as a user with superuser privileges.
j. Enable the automatic startup of serviced.

```
systemctl enable serviced
```

Starting Control Center (single-host deployment)

Use this procedure to start Control Center in a single-host deployment. The default configuration of the Control Center service (serviced) is to start when the host starts. This procedure is only needed after stopping serviced to perform maintenance tasks.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Determine whether serviced is configured to start when the system starts.

   ```
   systemctl is-enabled serviced
   ```

   - If the result is enabled, proceed to the next step.
   - If the result is disabled, enter the following command:

     ```
     systemctl enable serviced
     ```

3. Start the Control Center service.

   ```
   systemctl start serviced
   ```

4. **Optional:** Monitor the startup, if desired.

   ```
   journalctl -u serviced -f -o cat
   ```

Once Control Center is started, it is ready to start managing applications. For more information, see Adding Resource Manager to a Control Center deployment.

Starting Control Center (multi-host deployment)

Use this procedure to start Control Center in a multi-host deployment. The default configuration of the Control Center service (serviced) is to start when the host starts. This procedure is only needed after stopping serviced to perform maintenance tasks.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Determine whether serviced is configured to start when the system starts.
   ```
   systemctl is-enabled serviced
   ```
   - If the result is enabled, proceed to the next step.
   - If the result is disabled, enter the following command:
     ```
     systemctl enable serviced
     ```
3. Identify the hosts in the ZooKeeper ensemble.
   ```
   grep -E '^\b*SERVICED_ZK=' /etc/default/serviced
   ```
   The result is a list of 1, 3, or 5 hosts, separated by the comma character (,). The master host is always a node in the ZooKeeper ensemble.
4. In separate windows, log in to each of the delegate hosts that are nodes in the ZooKeeper ensemble as root, or as a user with superuser privileges.
5. On all ensemble hosts, start serviced.
   The window of time for starting a ZooKeeper ensemble is relatively short. The goal of this step is to start Control Center on each ensemble node at about the same time, so that each node can participate in electing the leader.
   ```
   systemctl start serviced
   ```
6. On the master host, check the status of the ZooKeeper ensemble.
   a. Attach to the container of the ZooKeeper service.
      ```
      docker exec -it serviced-isvcs_zookeeper bash
      ```
   b. Query the master host and identify its role in the ensemble.
      Replace Master with the hostname or IP address of the master host:
      ```
      { echo stats; sleep 1; } | nc Master 2181 | grep Mode
      ```
      The result includes leader or follower. When multiple hosts rely on the ZooKeeper instance on the master host, the result includes standalone.
   c. Query the other delegate hosts to identify their role in the ensemble.
      Replace Delegate with the hostname or IP address of a delegate host:
      ```
      { echo stats; sleep 1; } | nc Delegate 2181 | grep Mode
      ```
   d. Detach from the container of the ZooKeeper service.
      ```
      exit
      ```
   If none of the nodes reports that it is the ensemble leader within a few minutes of starting serviced, reboot the ensemble hosts.
7. Log in to each of the delegate hosts that are not nodes in the ZooKeeper ensemble as root, or as a user with superuser privileges, and then start serviced.
   ```
   systemctl start serviced
   ```
8. **Optional:** Monitor the startup, if desired.
   ```
   journalctl -u serviced -f -o cat
   ```

Once Control Center is started, it is ready to start managing applications. For more information, see see Adding Resource Manager to a Control Center deployment.

Control Center releases and images

This section associates Control Center releases with the tags of the required Docker images for each release. The images provide the virtual containers of the Control Center internal services and the ZooKeeper service. In addition, this section includes a procedure for identifying installed images.

Releases and image tags

Release	Internal services image tag	ZooKeeper image tag
Control Center 1.6.5	zenoss/serviced-isvcs:v63	zenoss/isvcs-zookeeper:v11
Control Center 1.6.3	zenoss/serviced-isvcs:v61	zenoss/isvcs-zookeeper:v11
Control Center 1.5.1	zenoss/serviced-isvcs:v61	zenoss/isvcs-zookeeper:v10
Control Center 1.5.0	zenoss/serviced-isvcs:v61	zenoss/isvcs-zookeeper:v10
Control Center 1.4.1	zenoss/serviced-isvcs:v60	zenoss/isvcs-zookeeper:v10

Identifying installed Docker images

Use this procedure to identify the local Docker images for Control Center that are installed on a host.

1. Log in to the Control Center host as root, or as a user with superuser privileges.
2. Display the local Docker images for Control Center.

```
docker images | awk '/isvcs/ { print $1, " ", $2}'
```

- If the installed image versions are higher than the versions that accompany a release, the images need to be updated. The upgrade procedures include steps for installing the required images.
- If the installed image versions are not higher than the versions that accompany a release, the images do not need to be updated.

Removing unused images

Use this procedure to identify and remove unused Control Center images.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Identify the images associated with the installed version of serviced.

   ```
   serviced version | grep Images
   ```

   Example result:

   ```
   IsvcsImages: [zenoss/serviced-isvcs:v61 zenoss/isvcs-zookeeper:v10]
   ```

3. Start Docker, if necessary.

   ```
   systemctl status docker || systemctl start docker
   ```

4. Display the serviced images in the local repository.

   ```
   docker images | awk '/REPO|isvcs/'
   ```

   Example result (edited to fit):

   ```
   REPOSITORY                    TAG       IMAGE ID
   zenoss/serviced-isvcs         v40       88cd6c24cc82
   zenoss/serviced-isvcs         v61       0aab5a2123f2
   zenoss/isvcs-zookeeper        v3        46fa0a2fc4bf
   zenoss/isvcs-zookeeper        v10       0ff3b3117fb8
   ```

   The example result shows the current versions and one set of previous versions. Your result may include additional previous versions and will show different images IDs.
5. Remove unused images.
   Replace Image-ID with the image ID of an image for a previous version.

   ```
   docker rmi Image-ID
   ```

   Repeat this command for each unused image.

Enabling serviced debug messages

You can enable debug messages without restarting `serviced`.

Enabling all debug messages

To enable debug messages for all of the packages included in `serviced`, follow these steps:

1. Log in to the host as `root`, or as a user with superuser privileges.
2. Toggle debug mode on.

   ```
   pkill -f -USR1 /opt/serviced/bin/serviced
   ```

3. Display the messages.

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

To turn off debug messages, repeat step 2.

Enabling debug messages for specific packages

The `serviced` daemon is written in Go, and many of the packages included in it use logrus logging. One of the features of logrus is the ability to change the logging level for individual packages without restarting the daemon. The daemon watches a configuration file and adjusts levels when the file changes.

To enable debug messages for `serviced` packages that use logrus logging, follow these steps:

1. Log in to the Control Center host as root, or as a user with superuser privileges.
2. Open `/opt/serviced/etc/logconfig-server.yaml` with a text editor.
3. Modify the logging for the package or packages that interest you.

   For example, to log everything at debug level except RPC requests (which are frequent and might hide useful logs), modify the file as follows:

   ```
   - logger: '*'
     level: debug
   - logger: rpc
     level: info
   ```

   For more package names, see the next section.
4. Save the file, and then exit the text editor.
5. Display the messages.

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

Control Center packages that use logrus logging

The following packages in release 1.5.1 use logrus logging. However, some do not include debug messages, and sometimes, understanding debug messages requires knowledge of the source code.

audit	coordinator/client/zookeeper	dfs/ttl	domain/properties	facade	rpc/master	stats
auth	coordinator/storage	domain	domain/registry	isvcs	rpc/rpcutils	utils
cli/api	dao/elasticsearch	domain/addressassignment	domain/service	metrics	scheduler	utils/iostat
cli/cmd	datastore/elastic	domain/host	domain/serviceconfigfile	node	scheduler/servicestatemanager	web
config	dfs	domain/hostkey	domain/servicedefinition	proxy	script	zzk
container	dfs/docker	domain/logfilter	domain/servicetemplate	rpc	servicedversion	zzk/registry
coordinator/client	dfs/nfs	domain/pool	domain/user	rpc/agent	shell	zzk/service

Control Center maintenance scripts

The scripts in the following list are installed when Control Center is installed, and are started either daily or weekly by `anacron`.

Script	Description	Leave installed on hosts in this pool?		
		Master	Resource Manager	Collector
`/etc/cron.hourly /serviced`	This script invokes `logrotate` hourly, to manage the files in `/var/log/serviced`.	Yes	No	No
`/etc/cron.weekly /serviced-fstrim`	This script invokes `fstrim` weekly, to reclaim unused blocks in the application data thin pool. The life span of a solid-state drive (SSD) degrades when `fstrim` is run too frequently. If the block storage of the application data thin pool is an SSD, you can reduce the frequency at which this script is invoked, as long as the thin pool never runs out of free space. An identical copy of this script is located in `/opt/serviced/bin`.	Yes	No	No
`/etc/cron.d /cron_zenossdbpack`	This script invokes `/opt/serviced/bin/serviced-zenossdbpack`, the database maintenance script for Resource Manager, every Sunday at midnight. If Resource Manager is not installed or is offline, the command fails. You can change the day of the week and time of day when the maintenance script is invoked by editing `/etc/cron.d/cron_zenossdbpack`. This script is required on one host in the resource pool in which the Resource Manager database services run.	Multi-host: No Single-host: Yes	Remove from all but one host	No

Configuration variables

This section includes the following major subsections:

Best practices for configuration files

The Control Center configuration file, `/etc/default/serviced`, contains Bash environment variables that are read by the serviced daemon startup script. The following list describes recommended best practices for its use and maintenance:

1. When in doubt, make a backup. Before editing, making a backup copy of the configuration file is always the safest choice.
2. Copy a variable, then edit the copy. If you need to revert a variable to its default value, you don't have to leave the file to look it up.
3. Copy and edit a variable only if the default value needs to be changed. It's easier to troubleshoot problems when only non-default variables are copied and edited.
4. Put the first character of the variable declaration in the first column of its line. It's easier to grep for settings when each one starts a line.
5. Add customizations to the top of the file. Customizations at the end of the file or scattered throughout the file may be overlooked.
6. In high-availability deployments, the contents of /etc/default/serviced on the master nodes must be identical. Use a utility like sum to compare the files quickly.

Master host configuration variables

The tables in this section provide an overview of the serviced configuration variables that apply to the Control Center master host. Set these variables as required for your environment or applications.

Storage variables

The variables in the following table are set only on the master host.

- Use one of the first two groups of variables but not both.
- Before starting the master host for the first time, you might need to change the defaults of the third group.
- Typically, the defaults of the last two groups of variables are not changed until Control Center has managed an application for a while and a need arises.

The SERVICED_STORAGE_STATS_UPDATE_INTERVAL variable sets the interval for collecting kernel statistics about the application data thin pool. Its default value is unlikely to require a change until a need arises.

Variable	Description	Purpose
SERVICED_FS_TYPE SERVICED_DM_ARGS SERVICED_DM_BASESIZE SERVICED_DM_THINPOOLDEV	The specifications of a devicemapper-based application data storage resource for production use.	Provide basic information about the data storage resource.
SERVICED_FS_TYPE SERVICED_DM_LOOPDATASIZE SERVICED_DM_LOOPMETADATASIZE SERVICED_ALLOW_LOOP_BACK	The specifications of a devicemapper-based application data storage resource for development use.	Provide basic information about the data storage resource.
SERVICED_ISVCS_PATH SERVICED_VOLUMES_PATH SERVICED_BACKUPS_PATH	The data storage paths of separate functional components of Control Center internal services.	Enable separate storage areas for one or more components. The default installation process puts all three components on the same device.
SERVICED_SNAPSHOT_TTL SERVICED_SNAPSHOT_USE_PERCENT SERVICED_MAX_DFS_TIMEOUT	The snapshot retention interval, the percentage of the data storage thin pool that is unused, and the snapshot attempt timeout interval.	Prevent the creation of snapshots that are too large to fit the thin pool.
SERVICED_LOGSTASH_MAX_DAYS SERVICED_LOGSTASH_MAX_SIZE SERVICED_LOGSTASH_CYCLE_TIME	The variables that manage the amount of space used by the application log storage service.	Prevent application logs from filling the storage device that logstash uses.

Internal services endpoint variables

The variables in the following table must be set identically on all Control Center delegate hosts.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Endpoint	Description
SERVICED_DOCKER_REGISTRY	(varies)	The local Docker registry for Control Center internal services images and application images.
SERVICED_ENDPOINT	Master-Host:4979	The serviced RPC server. The endpoint port number must match the value of SERVICED_RPC_PORT.

SERVICED_LOG_ADDRESS	Master-Host:5042	The logstash service.
SERVICED_LOGSTASH_ES	Master-Host:9100	The Elasticsearch service for logstash.
SERVICED_STATS_PORT	Master-Host:8443	The serviced metrics consumer service.
SERVICED_AUTH_TOKEN_EXPIRATION	(none)	The length of time a delegate authentication token is valid.

RPC service variables

The variables in the following table must be set identically on all Control Center hosts, except:

- SERVICED_RPC_PORT, set only on the master
- SERVICED_MAX_RPC_CLIENTS, set only on delegates

By default, serviced uses TLS to encrypt all RPC traffic. The SERVICED_KEY_FILE and SERVICED_CERT_FILE variables identify the digital certificate used for RPC, mux, and HTTP traffic.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Where to set
SERVICED_ENDPOINT	Master, delegates
SERVICED_MAX_RPC_CLIENTS	Delegates
SERVICED_RPC_PORT	Master
SERVICED_RPC_CERT_VERIFY	Master, delegates
SERVICED_RPC_DISABLE_TLS	Master, delegates
SERVICED_RPC_TLS_MIN_VERSION	Master, delegates
SERVICED_RPC_TLS_CIPHERS	Master, delegates
SERVICED_KEY_FILE	Master
SERVICED_CERT_FILE	Master
SERVICED_RPC_DIAL_TIMEOUT	Master, delegates
SERVICED_AUTH_TOKEN_EXPIRATION	Master

Multiplexer variables

The variables in the following table must be set identically on all Control Center hosts.

By default, serviced uses TLS to encrypt all mux traffic. The SERVICED_KEY_FILE and SERVICED_CERT_FILE variables identify the digital certificate used for RPC, mux, and HTTP traffic.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Where to set
SERVICED_MUX_PORT	Master, delegates
SERVICED_MUX_DISABLE_TLS	Master, delegates
SERVICED_MUX_TLS_MIN_VERSION	Master, delegates
SERVICED_MUX_TLS_CIPHERS	Master, delegates
SERVICED_KEY_FILE	Master
SERVICED_CERT_FILE	Master
SERVICED_AUTH_TOKEN_EXPIRATION	Master

HTTP server variables

The variables in the following table are set only on the master host, except the SERVICED_UI_PORT variable, which must be set identically on all Control Center hosts.

By default, serviced uses TLS to encrypt all HTTP traffic. The SERVICED_KEY_FILE and SERVICED_CERT_FILE variables identify the digital certificate used for RPC, mux, and HTTP traffic.

Variable	Description
SERVICED_UI_PORT	The port on which the HTTP server listens for requests.
SERVICED_TLS_MIN_VERSION	The minimum version of TLS that serviced accepts for HTTP traffic.
SERVICED_TLS_CIPHERS	The list TLS ciphers that serviced accepts for HTTP traffic.
SERVICED_KEY_FILE	The path of a digital certificate key file.
SERVICED_CERT_FILE	The path of a digital certificate file.

Browser interface variables (master host only)

The variables in the following table are set only on the master host.

Variable	Description
SERVICED_UI_POLL_FREQUENCY	The number of seconds between polls from browser interface clients.
SERVICED_SVCSTATS_CACHE_TIMEOUT	The number of seconds to cache statistics about services.
SERVICED_ADMIN_GROUP	The group on the master host whose members can use the browser interface.
SERVICED_ALLOW_ROOT_LOGIN	Determines whether root on the master host can use the browser interface.

Tuning variables (master host only)

Variable	Description
GOMAXPROCS	The maximum number of CPU cores that serviced uses.
SERVICED_ES_STARTUP_TIMEOUT	The number of seconds to wait for the Elasticsearch service to start.
SERVICED_MASTER_POOLID	The name of the default resource pool. This variable is only used the first time serviced is started.

Delegate host configuration variables

The tables in this section provide an overview of the serviced configuration variables that apply to Control Center delegate hosts. Set these variables as required for your environment or applications.

Delegate variables

The following miscellaneous variables apply only to delegate hosts.

Variable	Description
SERVICED_ZK	The list of hosts in the ZooKeeper ensemble.
SERVICED_STATS_PERIOD	The frequency at which delegates gather metrics to send to the master host.
SERVICED_IPTABLES_MAX_CONNECTIONS	The maximum number of open connections to allow on a delegate. The number increases when the master is unavailable and decreases when the master returns to service.

Internal services endpoint variables

The variables in the following table must be set identically on all Control Center delegate hosts.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Endpoint	Description
SERVICED_DOCKER_REGISTRY	(varies)	The local Docker registry for Control Center internal services images and application images.
SERVICED_ENDPOINT	Master-Host:4979	The serviced RPC server. The endpoint port number must match the value of SERVICED_RPC_PORT.
SERVICED_LOG_ADDRESS	Master-Host:5042	The logstash service.
SERVICED_LOGSTASH_ES	Master-Host:9100	The Elasticsearch service for logstash.
SERVICED_STATS_PORT	Master-Host:8443	The serviced metrics consumer service.
SERVICED_AUTH_TOKEN_EXPIRATION	(none)	The length of time a delegate authentication token is valid.

RPC service variables

The variables in the following table must be set identically on all Control Center hosts, except:

- SERVICED_RPC_PORT, set only on the master
- SERVICED_MAX_RPC_CLIENTS, set only on delegates

By default, serviced uses TLS to encrypt all RPC traffic. The SERVICED_KEY_FILE and SERVICED_CERT_FILE variables identify the digital certificate used for RPC, mux, and HTTP traffic.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Where to set
SERVICED_ENDPOINT	Master, delegates
SERVICED_MAX_RPC_CLIENTS	Delegates
SERVICED_RPC_PORT	Master
SERVICED_RPC_CERT_VERIFY	Master, delegates
SERVICED_RPC_DISABLE_TLS	Master, delegates
SERVICED_RPC_TLS_MIN_VERSION	Master, delegates
SERVICED_RPC_TLS_CIPHERS	Master, delegates
SERVICED_KEY_FILE	Master

SERVICED_CERT_FILE	Master
SERVICED_RPC_DIAL_TIMEOUT	Master, delegates
SERVICED_AUTH_TOKEN_EXPIRATION	Master

Multiplexer variables

The variables in the following table must be set identically on all Control Center hosts.

By default, serviced uses TLS to encrypt all mux traffic. The SERVICED_KEY_FILE and SERVICED_CERT_FILE variables identify the digital certificate used for RPC, mux, and HTTP traffic.

The SERVICED_AUTH_TOKEN_EXPIRATION variable affects RPC, mux, and internal services endpoint traffic.

Variable	Where to set
SERVICED_MUX_PORT	Master, delegates
SERVICED_MUX_DISABLE_TLS	Master, delegates
SERVICED_MUX_TLS_MIN_VERSION	Master, delegates
SERVICED_MUX_TLS_CIPHERS	Master, delegates
SERVICED_KEY_FILE	Master
SERVICED_CERT_FILE	Master
SERVICED_AUTH_TOKEN_EXPIRATION	Master

Universal configuration variables

The tables in this section provide an overview of the serviced configuration variables that apply to all Control Center hosts. Set these variables as required for your environment or applications.

Role variable

Variable	Description
SERVICED_MASTER	Assigns the role of a serviced instance, either master or delegate. The master runs the application services scheduler and other internal services. Delegates run the application services assigned to the resource pool to which they belong.

Browser interface variable

Variable	Description
SERVICED_UI_PORT	The port on which the HTTP server listens for requests.

Networking variables

Variable	Description
SERVICED_STATIC_IPS	A list of one or more static IP addresses for IP assignment.
SERVICED_OUTBOUND_IP	The IP address of the network interface for serviced to use. When this variable is not set, serviced uses the IP address of the default network interface and assumes it has internet access. To prevent serviced from assuming it has internet access, set this variable.
SERVICED_VIRTUAL_ADDRESS_SUBNET	The private network for containers that use virtual IP addresses. The default is 10.3.0.0/16, and the network can be unique on each host. A /29 network is sufficient.
SERVICED_DOCKER_DNS	A list of one or more DNS servers. The list is injected into all Docker containers.

Debugging variables

Variable	Description
SERVICED_LOG_LEVEL	The log level serviced uses when writing to the system log.
SERVICED_DEBUG_PORT	The port on which serviced listens for HTTP requests for the Go profiler.
SERVICED_DOCKER_LOG_DRIVER	The log driver for all Docker container logs.
SERVICED_DOCKER_LOG_CONFIG	Docker --log-opt options.

Tuning variables (all Control Center hosts)

Variable	Description
SERVICED_MAX_CONTAINER_AGE	The number of seconds serviced waits before removing a stopped container.
SERVICED_ISVCS_ENV_[0-9]+	Startup arguments to pass to specific internal services.
SERVICED_SERVICE_MIGRATION_TAG	Overrides the default value for the service migration image.
SERVICED_OPTS	Startup arguments for serviced.

SERVICED_CONTROLLER_BINARY	The path of the serviced-controller binary.
SERVICED_HOME	The path of the home directory for serviced.
SERVICED_ETC_PATH	The path of the directory for serviced configuration files.
SERVICED_VHOST_ALIASES	A list of hostname aliases for a host; for example, localhost.
SERVICED_ZK_CONNECT_TIMEOUT	The number of seconds Control Center waits for a connection to the lead ZooKeeper host.
SERVICED_ZK_PER_HOST_CONNECT_DELAY	The number of seconds Control Center waits before attempting to connect to the next host in its round-robin list of ZooKeeper hosts.
SERVICED_ZK_RECONNECT_START_DELAY SERVICED_ZK_RECONNECT_MAX_DELAY	These are used together when Control Center is unable to re-establish a connection with the lead ZooKeeper host.

Configuration file

The Control Center configuration file, `/etc/default/serviced`, contains Bash environment variables that are read by the `serviced` daemon startup script. The order of the following list matches the order of the variables in the file.

HOME

Default: (the value of shell variable HOME)

The path Docker clients use to locate the `.docker/config.json` authentication file, which contains Docker Hub credentials.

TMPDIR

Default: (the value of shell variable TMPDIR)

The path `serviced` uses for temporary files.

GOMAXPROCS

Default: 2

The maximum number of CPU cores `serviced` uses. This setting is relevant only on the master host.

The value can safely be set to 50% of the available processors on the host, and higher if necessary, but must always be fewer than the total number of available processors.

SERVICED_MASTER

Default: 1 (true)

Assigns the role of a `serviced` instance, either master or delegate. The master runs the application services scheduler and other internal services. Delegates run the application services assigned to the resource pool to which they belong.

Only one `serviced` instance can be the master; all other instances must be delegates. The default value assigns the master role. To assign the delegate role, set the value to 0 (false). This variable must be explicitly set on all Control Center hosts.

SERVICED_MASTER_IP

Default: `127.0.0.1`

A convenience variable, for use in places where the IP address or hostname of the master host is required. This variable is unused unless it is both set here and referenced elsewhere. (For example, by replacing `{{SERVICED_MASTER_IP}}` with `$SERVICED_MASTER_IP`.)

SERVICED_MASTER_POOLID

Default: `default`

The name of the default resource pool. This variable is only used the first time `serviced` is started.

SERVICED_ZK

Default: (none)

The list of endpoints in the `serviced` ZooKeeper ensemble, separated by the comma character (`,`). Each endpoint identifies an ensemble node. Each Control Center server and in-container proxy uses SERVICED_ZK to create a randomized, round-robin list, and cycles through the list when it attempts to establish a connection with the lead ZooKeeper host.

SERVICED_DOCKER_REGISTRY

Default: `localhost:5000`

The endpoint of the local Docker registry, which `serviced` uses to store internal services and application images.

If the default value is changed, the host's Docker configuration file must include the `--insecure-registry` flag with the same value as this variable.

The safest replacement for localhost is the IPv4 address of the registry host. Otherwise, the fully-qualified domain name of the host must be specified.

SERVICED_OUTBOUND_IP

Default: (none)

The IPv4 address that delegates use to connect to the master host. When no address is specified, `serviced` attempts to discover its public IP address by pinging google.com.

This variable must be set on all Control Center hosts in either of the following scenarios:

- Control Center is deployed behind a firewall and google.com is not reachable. Set the value to the IPv4 address of the master host.
- Control Center is deployed in a high-availability cluster. Set the value to the virtual IPv4 address of the high-availability cluster (HA-Virtual-IP).

Setting the Docker HTTP_PROXY or HTTPS_PROXY environment variables prevents access to the IP address defined with this variable. To enable access, unset the Docker variables, and then reboot the host.

SERVICED_STATIC_IPS

Default: (none)

A list of one or more static IP addresses that are available for IP assignment. Use the comma character (`,`) to separate addresses.

SERVICED_ENDPOINT

Default: `{{SERVICED_MASTER_IP}}:4979`

The endpoint of the `serviced` RPC server. Replace `{{SERVICED_MASTER_IP}}` with the IP address or hostname of the `serviced` master host. The port number of this endpoint must match the value of the SERVICED_RPC_PORT variable defined on the `serviced` master host.

SERVICED_MAX_RPC_CLIENTS

Default: 3

The preferred maximum number of simultaneous connections a `serviced` delegate uses for RPC requests. The value is used to create a pool of sockets, which are reused as needed. Increasing the value increases the number of open sockets and the use of socket-related operating system resources.

When the demand for connections exceeds the supply of open sockets, `serviced` opens more sockets. When demand eases, `serviced` reduces the number of open sockets to the preferred maximum.

SERVICED_RPC_PORT

Default: 4979

The port on which the `serviced` RPC server listens for connections. The value of this variable must match the port number defined for the SERVICED_ENDPOINT variable on all `serviced` delegate hosts.

SERVICED_RPC_CERT_VERIFY

Default: false

Determines whether `serviced` performs TLS certificate verification for RPC connections. The certificate is defined by the SERVICED_CERT_FILE variable.

SERVICED_RPC_DISABLE_TLS

Default: false

Determines whether `serviced` encrypts RPC traffic with TLS.

SERVICED_RPC_TLS_MIN_VERSION

Default: `VersionTLS10`

The minimum version of TLS `serviced` accepts for RPC connections. Valid values include the default, `VersionTLS11`, and `VersionTLS12`.

SERVICED_RPC_TLS_CIPHERS

Default: (list of ciphers)

The list of TLS ciphers `serviced` prefers for RPC connections, separated by the comma character (`,`):

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Other ciphers are supported; the preceding ciphers provide strong security for relatively low processing overhead.

An instance of `serviced` is on both ends of an RPC connection, so both daemons use the first cipher in the list. To use a different cipher, put it first in the list, on all Control Center hosts.

SERVICED_UI_PORT

Default: `:443`

The port on which the `serviced` HTTP server listens for requests for its internal services and for tenant services. The value may be expressed as follows:

IP-Address`:`*Port-Number*

`:`*Port-Number*

Port-Number

Tenant applications can specify alternative ports with the port public endpoint feature.

The value of this variable must be identical on all Control Center hosts in a deployment.

SERVICED_UI_POLL_FREQUENCY

Default: 3

The number of seconds between polls from Control Center browser interface clients. The value is included in a JavaScript library that is sent to the clients.

SERVICED_MUX_PORT

Default: 22250

The port `serviced` uses for traffic among Docker containers.

SERVICED_MUX_DISABLE_TLS

Default: 0

Determines whether inter-host traffic among Docker containers is encrypted with TLS. Intra-host traffic among Docker containers is not encrypted. To disable encryption, set the value to 1.

SERVICED_MUX_TLS_MIN_VERSION

Default: `VersionTLS10`

The minimum version of TLS `serviced` accepts for mux traffic. Valid values include the default, `VersionTLS11`, and `VersionTLS12`.

SERVICED_MUX_TLS_CIPHERS

Default: (list of ciphers)

The list of TLS ciphers `serviced` prefers for mux traffic, separated by the comma character (`,`):

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Other ciphers are supported; the preceding ciphers provide strong security for relatively low processing overhead.

An instance of `serviced` is on both ends of a mux connection, so both daemons use the first cipher in the list. To use a different cipher, put it first in the list, on all Control Center hosts.

SERVICED_ISVCS_PATH

Default: `/opt/serviced/var/isvcs`

The location of `serviced` internal services data.

SERVICED_VOLUMES_PATH

Default: `/opt/serviced/var/volumes`

The location of `serviced` application data.

SERVICED_BACKUPS_PATH

Default: `/opt/serviced/var/backups`

The location of `serviced` backup files.

SERVICED_LOG_PATH

Default: `/var/log/serviced`

The location of `serviced` audit log files. Non-audit (operations) messages are written to `journald`.

SERVICED_KEY_FILE

Default: `$TMPDIR/zenoss_key.[0-9]+`

The path of a digital certificate key file. Choose a location that is not modified during operating system updates, such as `/etc`.

This key file is used for all TLS-encrypted communications (RPC, mux, and HTTP). The default, insecure key file is created when the `serviced` web server first starts, and is based on a public key that is compiled into `serviced`.

SERVICED_CERT_FILE

Default: `$TMPDIR/zenoss_cert.[0-9]+`

The path of a digital certificate file. Choose a location that is not modified during operating system updates, such as `/etc`. Certificates with passphrases are not supported.

This certificate file is used for all TLS-encrypted communications (RPC, mux, and HTTP). The default, insecure certificate file is created when the `serviced` web server first starts, and is based on a public certificate that is compiled into `serviced`.

SERVICED_TLS_MIN_VERSION

Default: `VersionTLS10`

The minimum version of TLS that `serviced` accepts for HTTP traffic. Valid values include the default, `VersionTLS11`, and `VersionTLS12`.

SERVICED_TLS_CIPHERS

Default: (list of ciphers)

The list of TLS ciphers that `serviced` accepts for HTTP traffic, separated by the comma character (`,`):

1. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
2. TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
3. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
4. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
5. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
6. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
7. TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
8. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
9. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
10. TLS_RSA_WITH_AES_256_CBC_SHA
11. TLS_RSA_WITH_AES_128_CBC_SHA
12. TLS_RSA_WITH_3DES_EDE_CBC_SHA
13. TLS_RSA_WITH_AES_128_GCM_SHA256
14. TLS_RSA_WITH_AES_256_GCM_SHA384

To disable support for most ciphers, you can remove them from the list. The following rules apply to the list:

- The first cipher, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, must always be present in the list of ciphers.
- The first four ciphers in the list must always precede any of the ciphers that appear after the first four. The first four ciphers are valid for HTTP/2, while the remaining ciphers are not.

SERVICED_FS_TYPE

Default: `devicemapper`

The driver to manage application data storage on the `serviced` master host. Only `devicemapper` is supported in production deployments.

The only supported storage layout for the `devicemapper` driver is an LVM thin pool. To create a thin pool, use the serviced-storage utility. To specify the name of the thin pool device, use SERVICED_DM_THINPOOLDEV.

SERVICED_DM_ARGS

Default: (none)

Customized startup arguments for the `devicemapper` storage driver.

SERVICED_DM_BASESIZE

Default: `100G`

The base size of virtual storage devices for tenants in the application data thin pool, in gigabytes. The units symbol (G) is required. This variable is used when `serviced` starts for the first time, to set the initial size of tenant devices, and when a backup is restored, to set the size of the restored tenant device.

The base size device is sparse device that occupies at most 1MB of space in the application data thin pool; its size has no immediate practical impact. However, the application data thin pool should have enough space for twice the size of each tenant device it supports, to store both the data itself and snapshots of the data. Since the application data thin pool is an LVM logical volume, its size can be increased at any time. Likewise, the size of a tenant device can be increased, as long as the available space in the thin pool can support the larger tenant device plus snapshots.

SERVICED_DM_LOOPDATASIZE

Default: `100G`

Specifies the size of the data portion of the loop-back file. This setting is ignored when SERVICED_ALLOW_LOOP_BACK is false.

SERVICED_DM_LOOPMETADATASIZE

Default: 2G

Specifies the size of the metadata portion of the loop-back file. This setting is ignored when SERVICED_ALLOW_LOOP_BACK is false.

SERVICED_DM_THINPOOLDEV

Default: (none)

The name of the thin pool device to use with the `devicemapper` storage driver.

SERVICED_STORAGE_STATS_UPDATE_INTERVAL

Default: 300 (5 minutes)

The number of seconds between polls of kernel statistics about the application data thin pool.

This setting is ignored when the operating system kernel version is less than 3.10.0-366.

SERVICED_ALLOW_LOOP_BACK

Default: false

Determines whether loop-back files can be used with the `devicemapper` storage driver. This option is not supported for production use.

SERVICED_MAX_CONTAINER_AGE

Default: 86400 (24 hours)

The number of seconds `serviced` waits before removing a stopped container.

SERVICED_VIRTUAL_ADDRESS_SUBNET

Default: `10.3.0.0/16`

The private subnet for containers that use virtual IP addresses on a host. This value may be unique on each Control Center host, if necessary.

RFC 1918 restricts private networks to the 10.0/24, 172.16/20, and 192.168/16 address spaces. However, `serviced` accepts any valid IPv4 address space.

Specify the value in CIDR notation. A /29 network provides sufficient address space.

SERVICED_LOG_LEVEL

Default: 0

The log level `serviced` uses when writing to the system log. Valid values are 0 (normal) and 2 (debug).

See also Enabling serviced debug messages.

SERVICED_LOG_ADDRESS

Default: `{{SERVICED_MASTER_IP}}:5042`

The endpoint of the logstash service. Replace {{SERVICED_MASTER_IP}} with the IP address or hostname of the `serviced` master host.

SERVICED_LOGSTASH_ES

Default: `{{SERVICED_MASTER_IP}}:9100`

The endpoint of the Elasticsearch service for logstash. On delegate hosts, replace {{SERVICED_MASTER_IP}} with the IP address or hostname of the Elasticsearch host, which by default is the `serviced` master host.

SERVICED_LOGSTASH_MAX_DAYS

Default: 14

The maximum number of days to keep application logs in the logstash database before purging them.

SERVICED_LOGSTASH_MAX_SIZE

Default: 10

The maximum size of the logstash database, in gigabytes.

SERVICED_LOGSTASH_CYCLE_TIME

Default: 6

The amount of time between logstash purges, in hours.

SERVICED_STATS_PORT

Default: `{{SERVICED_MASTER_IP}}:8443`

The endpoint of the `serviced` metrics consumer service. Replace {{SERVICED_MASTER_IP}} with the IP address or hostname of the `serviced` master host.

SERVICED_STATS_PERIOD

Default: 10

The frequency, in seconds, at which delegates gather metrics to send to the `serviced` metrics consumer service on the master host.

SERVICED_SVCSTATS_CACHE_TIMEOUT

Default: 5

The number of seconds to cache statistics about services. The cache is used by Control Center browser interface clients.

SERVICED_DEBUG_PORT

Default: 6006

The port on which `serviced` listens for HTTP requests for the Go profiler. To stop listening for requests, set the value to -1.

SERVICED_ISVCS_ENV_[0-9]+

Default: (none)

Startup arguments to pass to internal services. You may define multiple arguments, each for a different internal service. The variables themselves, and their arguments, use the following syntax:

`SERVICED_ISVCS_ENV_%d`

Each variable name ends with a unique integer in place of `%d`.

Service-Name`:`*Key*`=`*Value*

The value of each variable includes the following elements, in order:

1. *Service-Name*, the internal service name. The following command returns the internal service names that may be used for *Service-Name*:

```
docker ps | awk '/serviced-isvcs:/{print $NF}'
```

2. The colon character (`:`).

3. *Key*, a variable to pass to the internal service.

4. The equals sign character (=).

5. *Value*, the definition of the variable to pass to the internal service.

The following example variable passes `ES_JAVA_OPTS=-Xmx4g` to the Elasticsearch internal service.

```
SERVICED_ISVCS_ENV_0=serviced-isvcs_elasticsearch-logstash:ES_JAVA_OPTS=-Xmx4g
```

SERVICED_ADMIN_GROUP

Default: `wheel`

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

SERVICED_ALLOW_ROOT_LOGIN

Default: 1 (true)

Determines whether the root user account on the `serviced` master host may be used to gain access to the `serviced` browser interface.

SERVICED_IPTABLES_MAX_CONNECTIONS

Default: 655360

The default value of this variable ensures that a `serviced` delegate does not run out of connections if the `serviced` master goes down. The connections are automatically cleaned up by the kernel soon after the `serviced` master comes back online.

SERVICED_SNAPSHOT_TTL

Default: 12

The number of hours an application data snapshot is retained before removal. To disable snapshot removal, set the value to zero. The application data storage can fill up rapidly when this value is zero or too high.

SERVICED_NFS_CLIENT

Default: 1

DEPRECATED: Prevent a delegate host from mounting the DFS.

SERVICED_SERVICE_MIGRATION_TAG

Default: 1.0.2

Overrides the default value for the service migration image.

SERVICED_ISVCS_START

Default: (none)

Enables one or more internal services to run on a delegate host. Currently, only `zookeeper` has been tested.

SERVICED_ISVCS_ZOOKEEPER_ID

Default: (none)

The unique identifier of a ZooKeeper ensemble node. The identifier must be a positive integer.

SERVICED_ISVCS_ZOOKEEPER_QUORUM

Default: (none)

The comma-separated list of nodes in a ZooKeeper ensemble. Each entry in the list specifies the ZooKeeper ID, IP address or hostname, peer communications port, and leader communications port of a node in the ensemble. Each quorum definition must be unique, so the IP address or hostname of the "current" host must be 0.0.0.0.

The following example shows the syntax of a node entry:

ZooKeeper-ID`@`*Host-IP-Or-Name*`:2888:3888`

SERVICED_DOCKER_LOG_DRIVER

Default: `json-file`

The log driver for all Docker container logs, including containers for Control Center internal services. Valid values:

- `json-file`
- `syslog`
- `journald`
- `gelf`
- `fluentd`
- `none`

This is a direct port of the Docker `--log-driver` option.

SERVICED_DOCKER_LOG_CONFIG

Default: `max-file=5,max-size=10m`

A comma-separated list of Docker `--log-opt` options as key=value pairs. To specify the default values for a log driver, or for drivers that need no additional options, such as `journald`, use a single comma character (`,`) as the value of this variable.

SERVICED_DOCKER_DNS

Default: (empty)

The IP address of one or more DNS servers. The value of this variable is injected into each Docker container that `serviced` starts. Separate multiple values with the comma character (,).

SERVICED_OPTS

Default: (empty)

Special options for the `serviced` startup command.

SERVICED_SNAPSHOT_USE_PERCENT

Default: 20

The amount of free space in the thin pool specified with SERVICED_DM_THINPOOLDEV, expressed as a percentage the total size. This value is used to determine whether the thin pool can hold a new snapshot.

SERVICED_ZK_SESSION_TIMEOUT

Default: 15

The number of seconds the lead ZooKeeper host waits before flushing an inactive connection.

SERVICED_ZK_CONNECT_TIMEOUT

Default: 1

The number of seconds Control Center waits for a connection to the lead ZooKeeper host.

SERVICED_ZK_PER_HOST_CONNECT_DELAY

Default: 0

The number of seconds `serviced` waits before attempting to connect to the next host in its round-robin list of ZooKeeper hosts. For more information about the round-robin list, see SERVICED_ZK.

SERVICED_ZK_RECONNECT_START_DELAY

Default: 1

SERVICED_ZK_RECONNECT_START_DELAY and SERVICED_ZK_RECONNECT_MAX_DELAY are used together when Control Center is unable to re-establish a connection with the lead ZooKeeper host.

To prevent unnecessary spikes in TCP traffic, `serviced` waits a randomized amount of time that is equal to plus or minus 20% of the value of SERVICED_ZK_RECONNECT_START_DELAY. If `serviced` is unable to reconnect after contacting all of the hosts in its round-robin list of ZooKeeper hosts, the wait time is increased by a randomized value and the process of attempting to reconnect begins again. If the attempts fail again, the process repeats until the wait time reaches the value of SERVICED_ZK_RECONNECT_MAX_DELAY, and the wait time of subsequent reconnection attempts is capped at SERVICED_ZK_RECONNECT_MAX_DELAY. Once connection is re-established, the wait time is reset to SERVICED_ZK_RECONNECT_START_DELAY.

For more information about the round-robin list, see SERVICED_ZK.

SERVICED_ZK_RECONNECT_MAX_DELAY

Default: 1

See SERVICED_ZK_RECONNECT_START_DELAY.

SERVICED_ES_STARTUP_TIMEOUT

Default: 240

The number of seconds to wait for the Elasticsearch service to start.

SERVICED_MAX_DFS_TIMEOUT

Default: 300

The number of seconds until a DFS snapshot attempt times out.

SERVICED_RPC_DIAL_TIMEOUT

Default: 30

The number of seconds until an RPC connection attempt times out.

SERVICED_AUTH_TOKEN_EXPIRATION

Default: 3600 (1 hour)

The expiration time, in seconds, of delegate authentication tokens. This timeout affects RPC, mux, and `serviced` internal services endpoint communications.

SERVICED_CONTROLLER_BINARY

Default: `/opt/serviced/bin/serviced-controller`

The path of the `serviced-controller` binary, which runs in every container that `serviced` manages.

SERVICED_HOME

Default: `/opt/serviced`

The path of the home directory for `serviced`.

SERVICED_ETC_PATH

Default: `/opt/serviced/etc`

The path of the directory for `serviced` configuration files. The default is `SERVICED_HOME/etc`.

SERVICED_VHOST_ALIASES

Default: (none)

A list of hostname aliases for a host; for example, `localhost`. Separate multiple values with the comma character (`,`).

Administering Linux systems

The following sections include Linux system administration procedures that can be useful for Control Center administrators.

- Cleaning up logs on RHEL/CentOS systems
- Managing storage on Linux hosts

Cleaning up logs on RHEL/CentOS systems

Control Center (`serviced`) uses the `systemd` journal facility to store its log messages. If the logs are cluttered with old messages, it can be time-consuming to get to the latest messages.

The following command removes all but the last 24 hours worth of log messages:

```
journalctl --vacuum-time=1d
```

Note that the preceding command removes log messages for all applications that use the `systemd` journal facility, not just `serviced` messages.

For more information, refer to the `journalctl` man page on your host.

Managing storage on Linux hosts

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

Identifying storage devices and their configuration

This procedure identifies block storage devices attached to a host and demonstrates how devices are configured.

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

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

The following example result shows three disks (sda, sdb, and sdc) and one CD/DVD drive (sr0).

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

Example result:

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

For more information about lsblk, enter man lsblk.

Creating primary partitions

To perform this procedure, you need:

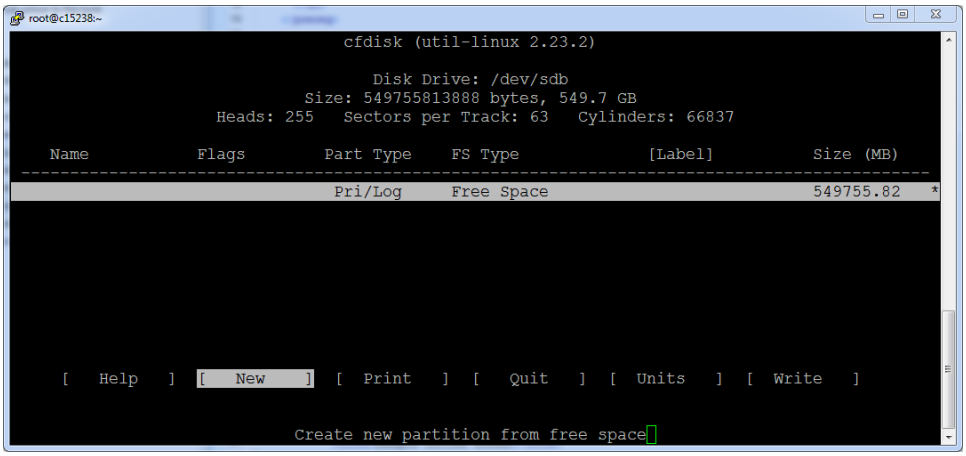
- The password of the root user account on a Linux host or a user account that belongs to the wheel group.
- A Linux host with at least one local or remote disk.

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

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

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

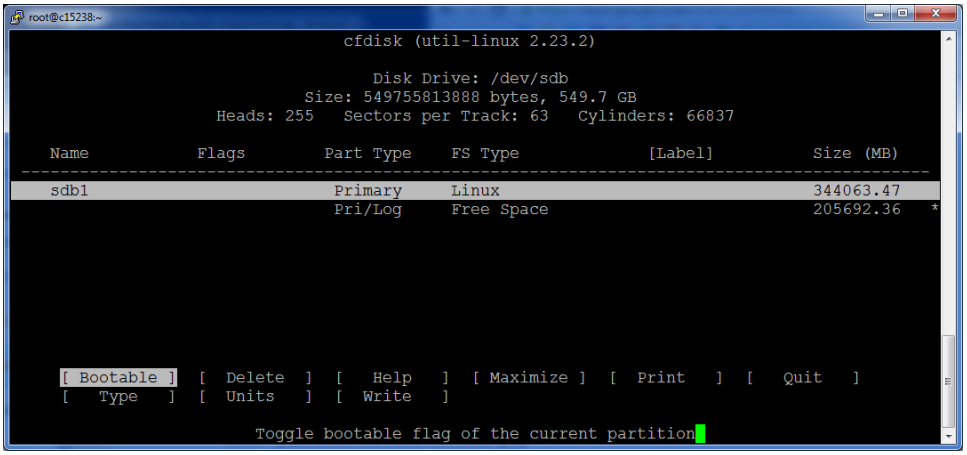
   ```
   cfdisk /dev/sdb
   ```

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

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

   For more information about cfdisk, enter man cfdisk.
3. Create a new partition.
   Repeat the following substeps to create up to four primary partitions.
   a. Select the table entry with the value Free Space in the FS Type column.
   b. Select [New] and press **Enter**.
   c. Select [Primary] and press **Enter**.
   d. At the Size (in MB) prompt, enter the size of the partition to create in megabytes, and then press **Enter**.
      To accept the default value (all free space on the disk), press **Enter**.
   e. **Optional:** Note: If you created a single partition that uses all available disk space, skip this substep. Select [Beginning] and press **Enter**.



4. Write the partition table to disk, and then exit the partition table editor.
    a. Select [Write] and press **Enter**.
    b. At the confirmation prompt, enter yes and then press **Enter**.
    You can ignore the warning about a bootable partition.
    c. Select [Quit] and press **Enter**.

Creating a swap partition

To perform this procedure, you need:

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

Perform this procedure to configure a primary partition on a local disk as swap space. Typically, configuring one swap partition or swap file on each local disk maximizes swap space performance.
This procedure does not use LVM tools to create a swap space. For more information about LVM, refer to your operating system documentation.

1. Log in to the target host as root, or as a user with superuser privileges.
2. Identify one or more primary partitions for use as swap space.

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

3. Create and enable swap space on each target primary partition.
   a. Disable swapping on all swap devices.

      ```
      swapoff -a
      ```

   b. Create swap space.
      Repeat the following command for each primary partition to use as swap space.
      Replace Device with the path of a primary partition:

      ```
      mkswap Device
      ```

   c. Update the file system table.
      Repeat the following command for each swap partition created in the previous substep.
      Replace Device with the path of a swap partition:

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

   d. Enable swapping on all swap devices.

      ```
      swapon -a
      ```

Updating Control Center

This section includes procedures for updating Control Center to the latest version.

The following table summarizes the new features of the update paths that are included in this section.

From	To	New features
1.5.x or 1.6.3	1.6.5	The following changes are addressed during this update. • This release replaces Docker Community Edition (CE) 17.09.0 with Docker CE 18.09.6. • OpenTSDB is updated to 2.3.1, so the update requires a new version of the `serviced-isvcs` Docker image.
1.4.1 or 1.4.2	1.6.5	The following changes are addressed during this update. • Automatic restart when a service fails 3 health checks in a row. • RHEL/CentOS 7.4, 7.5, and 7.6 are added; RHEL/CentOS 7.1 is withdrawn. The update process includes an optional step for updating the operating system. • On RHEL/CentOS 7.4 and up, there may be a file locking defect in NFS 4.1. To avoid the issue, delegate hosts are configured to use NFS 4.0. • The `serviced-zenossdbpack` maintenance script is moved from `/etc/cron.weekly` to `/opt/serviced/bin` and a new `cron` job, `/etc/cron.d/cron_zenossdbpack`, is installed to invoke the maintenance script. By default, the script runs every Sunday at midnight. • This release replaces Docker Community Edition (CE) 17.03.1 with Docker CE 17.09.0.

The following list outlines recommended best practices for updating Control Center deployments:

1. Review the release notes for this release. The latest information is provided there.
2. Compare the Docker images that accompany this release with the images of the installed release, and determine whether new image files need to be downloaded and installed. For more information, see Releases and image tags.
3. On delegate hosts, most of the update steps are identical. Use screen, tmux or a similar program to establish sessions on each delegate host and perform the steps at the same time.
4. Review and verify the settings in delegate host configuration files (`/etc/default/serviced`) before starting the update. Ideally, the settings on all delegate hosts are identical, except on ZooKeeper nodes and delegate hosts that do not mount the DFS.
5. Review the update procedures before performing them. Every effort is made to avoid mistakes and anticipate needs; nevertheless, the instructions may be incorrect or inadequate for some requirements or environments.

Keep this page open, and open new tabs or windows for each update procedure.

Updating 1.5.x or 1.6.3 to 1.6.5

1. Download required files
2. Install the repository mirror
3. Stage Docker image files
4. Update the master host:
    a. Stop Control Center
    b. Update Docker
    c. Load image files
    d. Update the serviced binary
5. Update delegate hosts:
    a. Stop Control Center
    b. Update Docker
    c. Update the serviced binary
6. Start Control Center:
    - Start Control Center (single-host deployment)
    - Start Control Center (multi-host deployment)
7. Perform post-upgrade procedures:
    - Set the connection timeout of a resource pool

Updating 1.4.1 or 1.4.2 to 1.6.5

1. Download required files
2. Install the repository mirror
3. Stage Docker image files
4. Update the master host:
    a. Stop Control Center
    b. Update Docker
    c. Load image files
    d. Update the serviced binary
5. Update delegate hosts:
    a. Stop Control Center
    b. Update Docker
    c. Configure NFS 4.0

    d. Update the serviced binary
    e. Update the ZooKeeper image on ensemble nodes
6. Start Control Center:
    - Start Control Center (single-host deployment)
    - Start Control Center (multi-host deployment)
7. Perform post-upgrade procedures:
    - Set the connection timeout of a resource pool
    - Remove unused images

Updating Docker 17.09.1 to 18.09.6

Use this procedure to update Docker to version 18.09.6.

1. Log in as root, or as a user with superuser privileges.
2. Determine which version of Docker is installed.

```
docker -v
```

   - If the result is 18.09.6, stop. This procedure is unnecessary; proceed to the next one.
   - If the result is 17.03.1, proceed to Updating Docker 17.03.1 to 18.09.6.
   - If the result is 17.09.1, perform the remaining steps in this procedure.
3. Update the operating system, if necessary.
   a. Determine which release is installed.

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

      - If the result is greater than 7.2, proceed to the next step.
      - If the result is less than or equal to 7.1, perform the remaining substeps.
   b. Disable automatic start of `serviced`.

```
systemctl disable serviced
```

   c. Update the operating system, and then restart the host.
   The following commands require internet access or a local mirror of operating system packages.

```
yum makecache fast && yum update && reboot
```

   d. Log in as root, or as a user with superuser privileges.
   e. Enable automatic start of `serviced`.

```
systemctl enable serviced
```

4. Update the Linux kernel, if necessary.
   a. Determine which kernel version is installed.

```
uname -r
```

   If the result is lower than 3.10.0-327.22.2.el7.x86_64, perform the following substeps.
   b. Disable automatic start of serviced.

```
systemctl disable serviced
```

   c. Update the kernel, and then restart the host.
   The following commands require internet access or a local mirror of operating system packages.

```
yum makecache fast && yum update kernel && reboot
```

   d. Log in as root, or as a user with superuser privileges.
   e. Enable automatic start of `serviced`.

```
systemctl enable serviced
```

5. Stop the Docker service.

```
systemctl stop docker
```

6. Remove Docker 17.09.0.
   a. Remove without checking dependencies.

```
rpm -e --nodeps docker-ce
```

   b. Clean the `yum` databases.

```
yum clean all
```

7. Install Docker CE 18.09.6.

```
yum install --enablerepo=zenoss-mirror docker-ce-18.09.6-3.el7
```

   If yum returns an error due to dependency issues, see Resolving package dependency conflicts for potential resolutions.
8. Start the Docker service.

```
systemctl start docker
```

Updating Docker 17.03.1 to 18.09.6

Use this procedure to update Docker to version 17.03.1 to 18.09.6.

1. Log in as root, or as a user with superuser privileges.
2. Determine which version of Docker is installed.

```
docker -v
```

   - If the result is 18.09.6, stop. This procedure is unnecessary; proceed to the next one.
   - If the result is 17.09.1, proceed to Updating Docker 17.09.1 to 18.09.6.
   - If the result is 17.03.1, perform the remaining steps in this procedure.
3. Remove the Docker repository description file, if present.
   Upgrades no longer require external repositories.

```
rm -f /etc/yum.repos.d/docker.repo
```

4. Update the operating system, if necessary.
   a. Determine which release is installed.

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

      - If the result is 7.2 or greater, proceed to the next step.
      - If the result is 7.1, perform the remaining substeps.
   b. Disable automatic start of `serviced`.

```
systemctl disable serviced
```

   c. Update the operating system, and then restart the host.
      The following commands require internet access or a local mirror of operating system packages.

```
yum makecache fast && yum update && reboot
```

   d. Log in as root, or as a user with superuser privileges.
   e. Enable automatic start of `serviced`.

```
systemctl enable serviced
```

5. Update the Linux kernel, if necessary.
   a. Determine which kernel version is installed.

```
uname -r
```

      If the result is lower than 3.10.0-327.22.2.el7.x86_64, perform the following substeps.
   b. Disable automatic start of serviced.

```
systemctl disable serviced
```

   c. Update the kernel, and then restart the host.
      The following commands require internet access or a local mirror of operating system packages.

```
yum makecache fast && yum update kernel && reboot
```

   d. Log in as root, or as a user with superuser privileges.
   e. Enable automatic start of `serviced`.

```
systemctl enable serviced
```

6. Stop the Docker service.

```
systemctl stop docker
```

7. Remove Docker 17.03.1.
   a. Remove without checking dependencies.

```
rpm -e --nodeps docker-ce
```

b. Clean the `yum` databases.

```
yum clean all
```

8. Install Docker CE 18.09.6.

```
yum install --enablerepo=zenoss-mirror docker-ce-18.09.6-3.el7
```

If yum returns an error due to dependency issues, see Resolving package dependency conflicts for potential resolutions.

9. Start the Docker service.

```
systemctl start docker
```

10. Remove a previous workaround for an NFS `rpcbind` issue.
    a. Remove the NFS service drop-in file, if it exists.

```
test -f /etc/systemd/system/nfs-server.service.d/nfs-server.conf \
  && rm -f /etc/systemd/system/nfs-server.service.d/nfs-server.conf
```

    b. Reload the `systemd` manager configuration.

```
systemctl daemon-reload
```

Updating the serviced binary

Use this procedure to update the `serviced` binary on a Control Center host. Perform this procedure on each host in your Control Center deployment. In multi-host deployments, stop `serviced` on the master host first.

1. Log in to the host as root, or as a user with superuser privileges.
2. Save the current `serviced` configuration file as a reference.
    a. Rename the file.
    Replace *VERSION* with the new version number; for example, 1.6.5:

    ```
    mv /etc/default/serviced /etc/default/serviced-pre-VERSION
    ```

    b. Set permissions to read-only.

    ```
    chmod 0440 /etc/default/serviced-pre-VERSION
    ```

3. Install the new `serviced` RPM package.
    Replace *VERSION* with the new version number; for example, 1.6.5:

    ```
    yum install --enablerepo=zenoss-mirror /opt/zenoss-repo-mirror/serviced-VERSION-1.x86_64.rpm
    ```

    If `yum` returns an error due to dependency issues, see Resolving package dependency conflicts for potential resolutions.
4. Make a backup copy of the new configuration file.
    a. Copy the file.
    Replace *VERSION* with the new version number; for example, 1.6.3:

    ```
    cp /etc/default/serviced /etc/default/serviced-VERSION-orig
    ```

    b. Set permissions to read-only.

    ```
    chmod 0440 /etc/default/serviced-VERSION-orig
    ```

5. Compare the new configuration file with the configuration file of the previous release.
    a. Identify the configuration files to compare.

    ```
    ls -l /etc/default/serviced*
    ```

    The original versions of the configuration files should end with `orig`, but you may have to compare the dates of the files.
    b. Compare the new and previous configuration files.
    Replace *New-Version* with the name of the new configuration file, and replace *Previous-Version* with the name of the previous configuration file:

    ```
    diff New-Version Previous-Version
    ```

    For example, to compare versions 1.6.3 and the most recent version (*VERSION*), enter the following command:

    ```
    diff /etc/default/serviced-1.6.3-orig /etc/default/serviced-VERSION-orig
    ```

    - If the command returns no result, restore the backup of the previous configuration file.

    ```
    cp /etc/default/serviced-pre-VERSION /etc/default/serviced && chmod 0644 /etc/default/serviced
    ```

    - If the command returns a result, restore the backup of the previous configuration file, and then optionally, use the results to edit the restored version.

    ```
    cp /etc/default/serviced-pre-VERSION /etc/default/serviced && chmod 0644 /etc/default/serviced
    ```

    For more information about configuring a host, see Configuration variables.
6. Reload the `systemd` manager configuration.

    ```
    systemctl daemon-reload
    ```

7. **Single-host deployments only**: Start the `serviced` service.

```
systemctl start serviced
```

Updating the ZooKeeper image on ensemble nodes

Use this procedure to install a new Docker image for ZooKeeper on ZooKeeper ensemble nodes.

1. Log in to the master host as root, or as a user with superuser privileges.
2. Identify the hosts in the ZooKeeper ensemble.

   ```
   grep -E '^\b*SERVICED_ZK=' /etc/default/serviced
   ```

   The result is a list of 3 or 5 hosts, separated by the comma character (,). The master host is always a node in the ZooKeeper ensemble.
3. Log in to a ZooKeeper ensemble node as root, or as a user with superuser privileges.
4. Change directory to /root.

   ```
   cd /root
   ```

5. Extract the ZooKeeper image.

   ```
   yes | ./install-zenoss-isvcs-zookeeper_v*.run
   ```

6. Optional: Delete the archive file.

   ```
   rm -i ./install-zenoss-isvcs-zookeeper_v*.run
   ```

7. Repeat the preceding four steps on each delegate that is a node in the ZooKeeper ensemble.

Release notes

This section contains important information about the following releases of Control Center:

Control Center 1.6.5

Beginning with version 1.4.0, Docker images for Control Center are no longer available on Docker Hub. The images are included in self-installing archive files that are available for download from Zenoss. Likewise, the RPM packages that are required for installations and upgrades are no longer available from Zenoss repositories on the internet, and must be downloaded. However, the Docker images and RPM packages for previous releases are still available from Docker Hub and Zenoss repositories, respectively.

Beginning 1 December 2017, downloads for Zenoss Service Dynamics customers are available on delivery.zenoss.com.

New features

This release includes an upgrade of Docker CE to 18.09.6, and an upgrade of OpenTSDB to 2.3.1.

Fixed issues

ID	Description
CC-3490, CC-3955, CC-4056, CC-4190	Backups fail due to various conditions
CC-4000	OpenTSDB files are insecure because permissions on `/opt/serviced` are not 750
CC-4145	The zenoss-installer script does not calculate block device space correctly
CC-4208	The `serviced tune` option `--ramThreshold` does not support percentage values
CC-4212	OpenTSDB 2.3.0 vulnerabilities
CC-4231	Upgrade documentation should include a check for the installed version of Docker
CC-4232	Self-monitoring does not include statistics for JVM-based services (ElasticSearch and OpenTSDB)
CC-4236	Hourly snapshots create too many images, which do not get cleaned up

Known issues

ID	Description
CC-4213	The zentrap and zensyslog services cannot support load balancing
CC-4145	The zenoss-installer script does not calculate available storage space correctly
CC-4077	Service restarts are not included in audit logs
CC-4072	Logstash health checks do not properly verify service health

Control Center 1.6.3

Beginning with version 1.4.0, Docker images for Control Center are no longer available on Docker Hub. The images are included in self-installing archive files that are available for download from Zenoss. Likewise, the RPM packages that are required for installations and upgrades are no longer available from Zenoss repositories on the internet, and must be downloaded. However, the Docker images and RPM packages for previous releases are still available from Docker Hub and Zenoss repositories, respectively.

Beginning 1 December 2017, downloads for Zenoss Service Dynamics customers are available on delivery.zenoss.com.

New features

Automatic restart when a service fails 3 health checks in a row

(CC-3559) This feature ensures deadlocked services are restarted after 3 failed health checks (2.5 minutes). Zope-based services in particular (`zauth`, `zenreports`, `zope`, and `zenapi`) benefit from this update.

Fixed issues

ID	Description
CC-4170	Health checks for the API proxy leak connections.
CC-4162	The ZooKeeper client library panics when it receives a large message.
CC-4119	Memory leak caused by the `time.After` function.
CC-4105	No authentication or authorization is enforced when a server attempts to join a ZooKeeper quorum.
CC-3559	Health check does not initiate restart of deadlocked service.

Control Center 1.5.1

Beginning with version 1.4.0, Docker images for Control Center are no longer available on Docker Hub. The images are included in self-installing archive files that are available for download from Zenoss. Likewise, the RPM packages that are required for installations and upgrades are no longer available from Zenoss repositories on the internet, and must be downloaded. However, the Docker images and RPM packages for previous releases are still available from Docker Hub and Zenoss repositories, respectively.

Beginning 1 December 2017, downloads for Zenoss Service Dynamics customers are available on delivery.zenoss.com.

New features

Configurable service restart thresholds

(CC-4100) Restart thresholds are available to manage services. In the Control Center browser interface, you can edit the service to specify a value in the new field Restart when memory usage exceeds % of requested RAM. When your threshold is reached, the service restarts automatically.

For more information about how this feature can be used, see the release notes for Resource Manager 6.2.1.

Fixed issues

ID	Description
CC-3367	The login page does not display the Control Center version.
CC-3766	Error message about locking services does not include enough information to trace the cause.
CC-3893	The logstash filebeat agent does not release file handles and container fills and crashes.
CC-3909	Upgrade process reports aborting but continues to completion when thin pool metadata size is too small.
CC-3926	Missing container not detected when a service is down.
CC-3937	Control Center does not export the same graph data it displays in its browser interface.
CC-3959	Setting a non-default value for SERVICED_UI_PORT breaks compatibility with Zenoss Resource Manager.
CC-3964	Users who are members of more than 99 groups are unable to log in to browser interface.
CC-3997	Pre-upgrade script fails to parse output of lvs command.
CC-4000	The permissions on OpenTSDB files on the Control Center master host are too open.
CC-4082	Log messages do not include the current resource pool timeout value.
CC-4100	Add restart thresholds. For more information, see New features.

Known issues

ID	Description
CC-4039	rpcbind fails to start when IPv6 is disabled. For more information, see Notes and workarounds.

Notes and workarounds

CC-4039: rpcbind fails to start when IPv6 is disabled

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.