The Zenoss Enablement Series:

Configuring Control Center 1.0.x for HA

Document Version 500-P4

Zenoss, Inc.

www.zenoss.com

Table of Contents

Introduction

Applies To

The procedure outlined in this document applies to the following Linux and Zenoss versions:

- Control Center Version 1.0.6 and higher
- RHEL/CentOS Linux 7.0
- Pacemaker version 1.1.12
- Corosync version 2.3.4
- DRBD version 8.4.6

Summary

This document provides an overview and step-by-step directions to deploy highly-available Control Center master agents on a fast local area network with RHEL or CentOS 7.

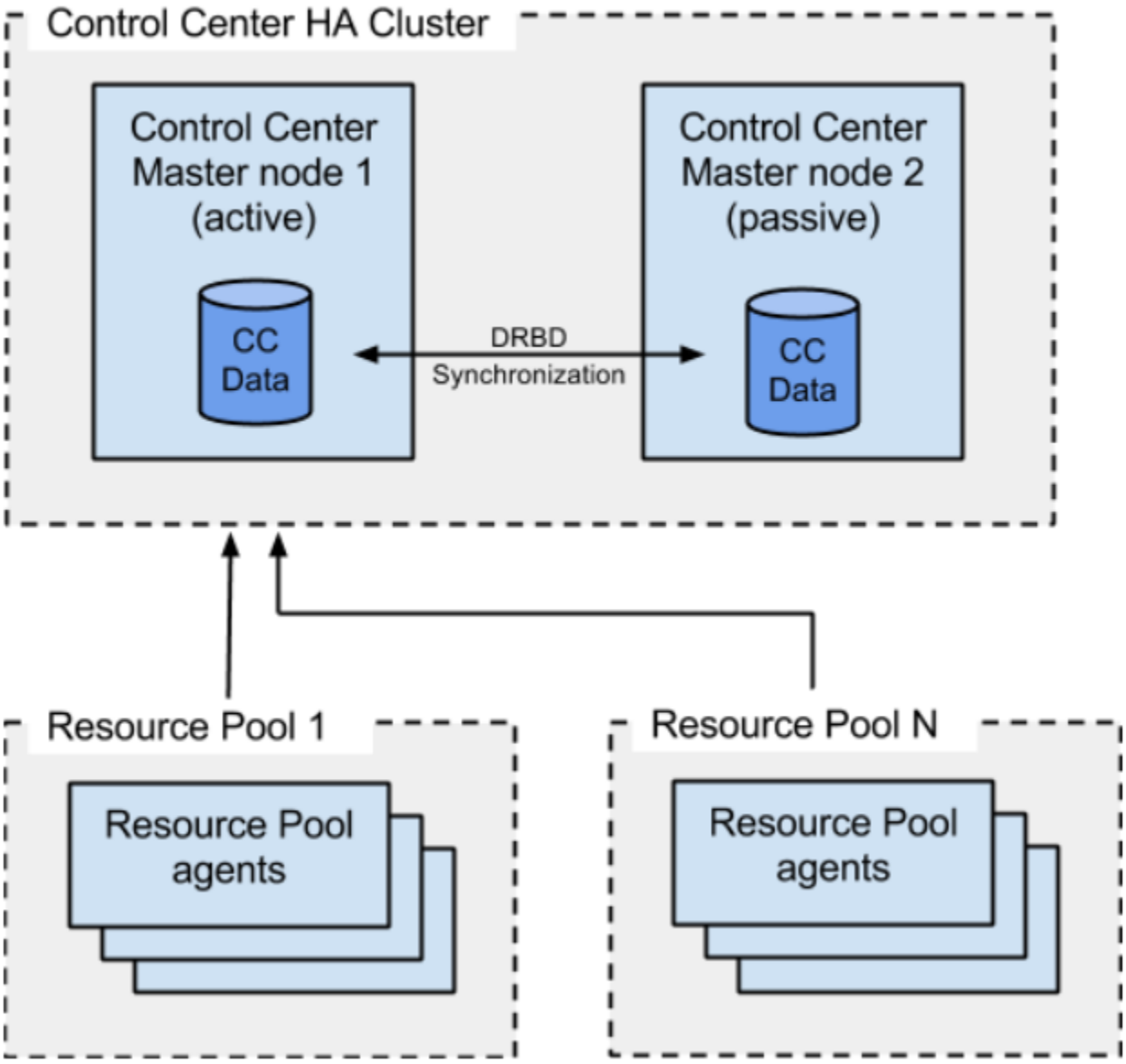
The Control Center (CC) architecture supports the notion of a resource pool which is a collection of compute, network and storage resources necessary to run a service, such as Zenoss Resource Manager. Specifying two or more CC agent hosts to a given a resource pool is the primary means of ensuring availability of the services in that pool in the event one of the hosts in that pool goes down. However, every CC deployment requires a single host to run as a master agent for CC, which creates a potential single point of failure for CC itself.

The objective of setting up the Control Center master agent in a 'high availability' cluster is to minimize, to the greatest degree possible, the downtime associated with a hardware or (non Zenoss) software failure of the system hosting the Control Center master agent. High availability clusters can have various configurations, including, but not limited to:

- Active-Passive, non geo-diverse
- Active-Active non geo-diverse
- Active-Passive, geo-diverse

This document describes an Active – Passive high availability cluster without geo diversity that uses Pacemaker, Corosync and Distributed Replicated Block Device (DRBD). For our scenario, at least two identical servers per cluster are deployed. At any given time, one node serves as the 'primary' active server and a second identical server stands by ready to take over provision of the Control Center master agent in the event the first server fails or otherwise becomes unavailable. This solution is termed *lacking geo diversity* because the two servers are co-located in the same facility. As such, this solution provides no protection against a scenario that destroys or renders unreachable the facility hosting the Control Center master agent.

The following diagram shows the relationship between the HA hardware cluster for master agents and pools of resource agents.



Naming Conventions

The following naming conventions are used in this document. Replace these example names with the names or values for your environment

- CLUSTERVIP - the floating Virtual IP (VIP) address on the public network for the cluster. This VIP is automatically assigned to the active node by the cluster management software. All CC applications running outside of the cluster will use this IP (or its associated host name) to communicate with the CC master agent running in the cluster, that includes the CC CLI, web UI, and CC resource pool agents.

- NODE{1,2}-PUBLIC - hostname of the CC master node that resolves to its public IP address. The node should have this name as its hostname as returned by `uname -n`.
- NODE{1,2}-PUBLIC-IP - the public IP of the CC master node. This IP does not have to be public to the entire enterprise, but it should be public to the machines used by CC administrators.
- NODE{1,2}-PRIVATE - the hostname of the CC master node that resolves to its private IP
- NODE{1,2}-PRIVATE-IP - the private IP of the CC master node.

The sample commands in this document are presented in a fixed-width font on a gray background. Text highlighted in red identifies values that you can customize for your environment. The values in less than (**<**) and greater than (**>**) symbols are ones you *must* replace with values that are appropriate for your environment. The other text in red represents you *can* customize if you want to. Consider the following example:

```
# pcs cluster setup --name serviced-ha \
        <NODE1-PUBLIC> <NODE2-PUBLIC>
```

The value `serviced-ha` is the logical name of the cluster in this command, but that value is just a suggestion - you can use another name if you want to. However, you *must* specify your own environment-specific values for `<NODE1-PUBLIC>` and `<NODE2-PUBLIC>`.

The following acronyms are used in this guide:

- **CC** - Control Center
- **VIP** - Virtual IP Address
- **RM** - Zenoss Resource Manager
- **RM Install Guide** - Zenoss Resource Manager Installation Guide

Prerequisites

The following hardware requirements must be satisfied to successfully install and configure Control Center in a high availability environment:

- Two identical RHEL 7.0/7.1 or CentOS 7.0/7.1 machines to run as the master Control Center agents. Consult the RM Install Guide for the required system specifications for CC master hosts.
- Two or more identical machines to run as resource pool hosts for Control Center. To provide HA for CC resource pools, each pool must have at least N+1 hosts where N is the number of hosts needed to satisfy the performance and scalability requirements for that pool. The additional host is necessary so that if one host fails, the remaining hosts in the pool have sufficient capacity to handle the workload for that pool. Consult the RM Install Guide for the required system specifications for CC resource pool hosts.
- The same hardware architecture for all node systems. The cluster manager node and cluster nodes should have the same processor architecture (x86_64) and OS version (RHEL 7.0, 7.1 or CentOS 7.0, 7.1). Because the cluster manager node configures and manages the clusters and creates DRBD RPM packages, it should share the same architecture as the node systems
- At least two filesystems per node for the CC data that must be replicated.
- Zenoss recommends using a supported fencing device. When fencing is employed, two network interface cards per machine is recommended with IP addresses configured for both public and private networks. When fencing cannot be employed, all cluster traffic should go through a single network interface. See Appendix A: Fencing for more information.

Consider the following prior to implementing Control Center in an HA cluster:

- The host clocks must be synchronized to a time server via Network Time Protocol (NTP).
- SELinux must be disabled on all hosts because it is not supported by Zenoss.
- Nodes should be located within a single LAN with multicast support.

The commands listed in this document are run as the *root* user unless otherwise specified. If you do not have direct access to the root account, then all of the commands in this guide should be run with `sudo`. We also assume that the Linux servers hosting the Control Center master agent have access to the standard Yum repositories (this means they have internet access).

Planning Your Deployment

The main characteristics of the recommended deployment architecture are:

- Each host in the HA cluster should have two NICs such that one1 NIC on each host is dedicated to disk synchronization via DRBD and the other is used by RM and CC application traffic.
- RM services are deployed on dedicated CC resource pool agents outside of the HA cluster.
- Fencing is employed on production HA clusters.

Having two NICs on each HA cluster node allows network traffic required for disk synchronization (DRBD) to be separated from network traffic required by the application itself. This results in more up-to-date disk synchronization and better network responsiveness from the CC and RM applications. However, if you do not have two identical, dual-NIC machines available, it is still possible to deploy an HA cluster with single-NIC servers.

Use two hosts for the CC master agent, where the two are configured in an *active/passive* configuration as described in this guide. These two hosts will only be used for running Control Center itself - they should NOT be used for running Zenoss Resource Manager. To run Zenoss Resource Manager, plan on deploying at least an additional two *Resource Pool* hosts. Only the Control Center master agents will be configured in an active/passive cluster using the instructions in this document. The two or more Resource Pool hosts will run as regular Control Center agents.

Fencing is a critical consideration. Fencing is a technique to ensure that a "failed" node in the cluster is completely stopped. This avoids situations where runaway processes on the failed node continue trying to use shared resources, resulting in application conflicts or conflicts with the cluster management software. The enormous number of fencing solutions makes it impractical for Zenoss to document solutions for every possible scenario. Control Center administrators must work with their IT departments to implement a fencing solution that works for their particular infrastructure. While fencing is recommended for production environments, CC administrators who want to stand up an initial test deployment of a CC HA cluster can do so without fencing, and then add fencing later. See Appendix A: Fencing for more information.

The general guidelines for deployment planning of the master agents are the same as those in ***Master host requirements*** in the *Planning your deployment* section of the **RM Install Guide**, except the following two file systems will be mirrored within the cluster:

i. `/opt/serviced/var/isvcs` (ext4)
ii. `/opt/serviced/var/volumes` (btrfs)

Notes:

1. The volume for `/var/lib/docker` does not need to be mirrored because the docker images seldom change, and docker manages downloading those images from public repos as necessary.

2. Because the volume `/opt/serviced/var/backups` does not contain frequently changing runtime data that must be mirrored on a real-time basis, it should not be mirrored.

3. Create partitions, but do not make file systems, for both of the volumes `/opt/serviced/var/isvcs` and `/opt/serviced/var/volumes` on both nodes. The file systems for both of these volumes will be created as part of configuring DRBD.

Prepare both Control Center Master Nodes (Active & Passive)

Perform the steps in this section on both machines.

Prepare the Master Host

Modifications to Standard Preparations

Following the steps outlined in ***Preparing the master host operating system*** in the **RM Install Guide** with the following exception - for the *Create an XFS file system* … step, only create the XFS file system for `/var/lib/docker`. The following is a short summary of the steps necessary to create just that one volume:

```
# mkdir -p /var/lib/docker
# DOCKER_PART=/dev/sdb1
# mkfs -t xfs $DOCKER_PART
# echo "$DOCKER_PART /var/lib/docker xfs defaults 0 0" >> /etc/fstab
# mount -a; mount | egrep docker
```

Additional Preparations for Clustering

Each master host must be assigned two unique static IP address; for example `NODE{1,2}-PUBLIC-IP` and `NODE{1,2}-PRIVATE-IP`. Using DHCP-leased IP addresses with machines in a cluster will interfere with the cluster management software.

Reserve a third static IP address to be used as the floating VIP (Virtual IP Address) for the cluster. Each cluster must have a floating VIP on the public network. The VIP is assigned to the active node automatically by the cluster management software. In case of interruption, the VIP is re-assigned to the standby node. The VIP is used by all processes outside of the cluster to contact the master. This includes end-users trying to login to CC or RM, as well as resource pool agents.

Modify the `/etc/hosts` file on both hosts to specify the static IP addresses and host names for both hosts.

Make sure `autostart` for NFS is disabled as a precaution to avoid unnecessary conflicts managing NFS-mounted volumes. For example:

```
# systemctl stop nfs && systemctl disable nfs
```

The CC master agent uses NFS to share files with CC agents in the various resource pools. Normally, having NFS running on a CC master agent is not a problem. However, special care must be taken in an HA cluster when NFS-mounted volumes are one of the shared cluster resources accessed via the floating VIP. In our case, the cluster management software will handle starting/stopping NFS in the cluster such that NFS is only running on the active node.

Install Control Center on the Master Host

Follow the steps outlined in the subsection ***Installing a master host*** for ***Installing on RHEL or CentOS hosts*** in the **RM Install Guide** with the following exceptions:

1. After you `Stop` and `Restart` Docker, perform two additional tasks:
    - pre-pull the `zenoss/serviced-isvcs` docker image
    - disable automatic startup of docker

    Note that the Control Center resource agent configuration for Pacemaker has a startup timeout. If Control Center fails to start within this timeframe, Pacemaker initiates a failover. The first time a new version of `serviced` starts it must first pull the `zenoss/serviced-isvcs` docker image. Typically, the time required to pull that image is greater than Pacemaker's startup timeout. Because of this time requirement, the docker image must be pulled locally before starting anything. Use the following command to pull the `zenoss/serviced-isvcs` docker image:

    ```
    # docker pull zenoss/serviced-isvcs:v27.1
    ```

    NOTES:

    - In the example above, the image label, *v27.1*, is specific to Control Center 1.0.4.
    - The image label for Control Center **1.0.7** is *v27.2*
    - The `serviced-isvcs` version may vary from one Control Center release to another. Consult the documentation for your specific release to verify which version of `zenoss/serviced-isvcs` is required for your installation.

2. When the isvcs image has been prestaged, docker must be stopped and disabled because the cluster manager will be responsible for starting/stopping docker as necessary:

    ```
    # systemctl stop docker && systemctl disable docker
    ```

3. Skip the Change the volume type for application data step.

4. For the *Configure the master host for a multi-host deployment* section, you must modify the default configuration to ensure that Control Center runs as a *master agent* using the VIP for the cluster. Perform the following:

    a) Create a variable named `CLUSTERVIP` with a value equal to the floating VIP of your cluster:

    ```
    # CLUSTERVIP=x.x.x.x
    ```

    NOTE: This step completely replaces the *Configure the Master host for a multi-host deployment* section

    b) Run the following commands to modify `/etc/default/serviced`:

    ```
    # EXT=$(date +"%j-%H%M%S")
    # test ! -z "${CLUSTERVIP}" && \
    sudo sed -i.${EXT} -e 's|^#[^H]*\(HOME=/root\)|\1|' \
    -e 's|^#[^S]*\(SERVICED_AGENT=\).*$|\11|' \
    -e 's|^#[^S]*\(SERVICED_ENDPOINT=\).*$|\1'${CLUSTERVIP}':4979|' \
    -e 's|^#[^S]*\(SERVICED_FS_TYPE=\).*$|\1btrfs|' \
    -e 's|^#[^S]*\(SERVICED_MASTER=\).*$|\11|' \
    -e 's|^#[^S]*\(SERVICED_OUTBOUND_IP=\).*$|\1'${CLUSTERVIP}'|' \
    ```

```
-e 's|^#[^S]*\(SERVICED_REGISTRY=\).*$|\11|' \
-e 's|^#[^S]*\(SERVICED_VARPATH=\).*$|\1/opt/serviced/var|' \
/etc/default/serviced
# test 1 -eq `grep SERVICED_OUTBOUND_IP= /etc/default/serviced \
>/dev/null 2>&1;echo $?` && \
  sudo echo "SERVICED_OUTBOUND_IP=${CLUSTERVIP}" >> \
/etc/default/serviced
```

c) Verify that `/etc/default/serviced` contains the proper contents. Run the following:

```
# grep ^SERVICED /etc/default/serviced | sort
```

The following is example output:

```
SERVICED_AGENT=1
SERVICED_ENDPOINT=<CLUSTERVIP>:4979
SERVICED_FS_TYPE=btrfs
SERVICED_MASTER=1
SERVICED_OUTBOUND_IP=<CLUSTERVIP>
SERVICED_REGISTRY=1
SERVICED_VARPATH=/opt/serviced/var
```

5. SKIP the *Start the Control Center* step; Do NOT start `serviced`. Instead, disable autostart for serviced. The HA cluster management software will handle starting and stopping CC. For example:

```
# systemctl disable serviced
```

6. Follow the steps outlined in the subsection "*Enabling access to the Control Center web interface*" in the **RM Install Guide**.

Install and Configure DRBD

Add ELRepo to RPM

DRBD packages are available on ELRepo. Issue the following commands to include *ELRepo* for both master hosts:

```
# rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
# rpm -Uvh http://www.elrepo.org/elrepo-release-7.0-2.el7.elrepo.noarch.rpm
# yum update -y
```

Install DRBD

Use the following command to install DRBD on BOTH nodes of the cluster:

```
# yum install -y drbd84-utils kmod-drbd84
```

Configure DRBD

Perform the following steps on BOTH nodes of the cluster:

1. Edit /etc/drbd.d/global_common.conf

Using a text editor, modify the file `/etc/drbd.d/global_common.conf` on both nodes to add **usage-count yes**; and **protocol C**; in the *global* and *common* sections as illustrated below:

```
global {
  usage-count yes;
}
common {
  net {
    protocol C;
  }
}
```

2. Create /etc/drbd.d/serviced-dfs.res

Using a text editor, create the file `/etc/drbd.d/serviced-dfs.res` with the following content on both nodes. The example below assumes **/dev/sdb** is the block device used for both of the mirrored volumes, **/opt/serviced/var/isvcs** and **/opt/serviced/var/volumes**. Use the device(s) appropriate for your installation.

```
resource serviced-dfs {
        volume 0 {
                device /dev/drbd0;
                disk /dev/sdb1;
                meta-disk internal;
        }
        volume 1 {
                 device /dev/drbd1;
                 disk /dev/sdb2;
                 meta-disk internal;
        }
        syncer {
                rate 30M;
        }
        net {
                after-sb-0pri discard-zero-changes;
                after-sb-1pri discard-secondary;
```

```
        }
        on <NODE1-PUBLIC> {
                address <NODE1-PRIVATE-IP>:7789;
        }
        on <NODE2-PUBLIC> {
                address <NODE2-PRIVATE-IP>:7789;
        }
}
```

NOTES:

1. This example assumes that each node in the cluster has two NICs such that the private IP/hostnames are reserved for use by DRBD. This is recommended so that real-time writes for disk synchronization between the DRBD master to slave are not in contention with application level network traffic. However, it is possible to use DRBD with a single NIC.
2. *7789* is the default port number used by DRBD.
3. By putting both volumes in the same DRBD resource definition, we are telling DRBD to synchronize and failover both volumes together.
4. DRBD will store its metadata on each volume (`meta-disk internal`;) so the total amount of space reported on the logical device `/dev/drbd<n>` will always be less than the amount of physical space available on the underlying physical disk partition.
5. The value `of syncer { rate: <value>` controls the rate, in bytes per second, at which DRBD synchronizes disks *after* the master/slave nodes have become out of sync. This rate should be adjusted based on *30%* of the available replication bandwidth. This is the slowest of either the I/O subsystem or the network interface. The example above assumes 90MB/s available for total replication bandwidth (0.30 X 90MB/s = 30MB/s)

Initial Enablement of the DRBD Device

The steps to enable the DRBD device depend on the status of your disks. If the disks are new, skip to **step 2**. If they are previously used, start with **step 1**.

1. If you are creating the DRBD device on disks that have already been used, you must use the `dd` command to zero out the existing filesystem. For example, if *sdb1* and *sdb2* are the two disks to be synchronized:

   ```
   dd if=/dev/zero of=/dev/sdb1 bs=1M count=128
   dd if=/dev/zero of=/dev/sdb2 bs=1M count=128
   ```

2. Execute the following commands to create the device metadata and enable the new DRBD resource on both nodes on the cluster.

   ```
   # drbdadm create-md all
   # drbdadm up all
   ```

Create Mount Points

Create the mount points for both volumes on both nodes of the cluster.

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

Perform First-Time DRBD Initialization

Perform these steps on the primary cluster node only.

1. Perform Initial Device Synchronization

   On the primary cluster node, enter the following command to synchronize the devices on both nodes.

   ```
   # drbdadm primary --force serviced-dfs
   ```

   Although the command may return right away, the synchronization process is running in the background. Depending on the size of the partitions, this process can require several minutes.

   Use the following command to monitor the progress of the synchronization process:

   ```
   # drbd-overview
   ```

   Do not proceed until the synchronization is complete. The process is complete when the status of both devices on both nodes is *UpToDate/UpToDate* as illustrated in the example output, below:

```
# drbd-overview
```

```
 0:serviced-dfs/0  Connected Primary/Secondary UpToDate/UpToDate
/opt/serviced/var/isvcs   ext4  4.8G 196M 4.4G 5%
 1:serviced-dfs/1  Connected Primary/Secondary UpToDate/UpToDate
/opt/serviced/var/volumes btrfs 45G  544K 43G  1%
#
```

   Note that in the command above, the field *Primary/Secondary* shows that this command was run on the primary cluster node. If you run the command on the secondary or slave node, the field will have the value *Secondary/Primary*. Likewise, in the next field, *UpToDate/UpToDate*, the first value is the status on the current node (the node where you executed the command), and the second value is the status on the remote node.

2. Format the Partitions

On the primary node, format the two partitions with the following commands:

```
# mkfs.ext4 /dev/drbd0
# mkfs.btrfs --nodiscard /dev/drbd1
```

Note that DRBD takes care of mirroring the file system partitioning to the slave machine

Install and Configure Cluster Management Software

Pacemaker is an open source cluster resource manager. It can use more than one type of cluster infrastructure application for communication and membership services within the cluster. For the purposes of this document, we are using *Corosync* as the cluster infrastructure application for communication and membership services.

The Pacemaker/Corosync daemon, `pcs.d`, communicates across nodes in the cluster. When `pcs.d` is installed, started and configured, the majority of pcs commands can be run on any available node in the cluster.

Install the Cluster Management Software

Run the following on both machines to install the cluster management software.
Note: For RHEL systems, you might need to add the *Centos Yum* repositories to retrieve these packages.

```
# yum install corosync pacemaker pcs
```

Install the Pacemaker Resource Agent for Control Center

Pacemaker uses “resource agents” that are scripts that implement a standardized interface to manage arbitrary resources in a cluster. Zenoss has developed a Pacemaker Resource Agent to manage the Control Center master in a cluster.

Run the following on both machines to install the Pacemaker Resource Agent for Control Center:

```
# yum --enablerepo=zenoss-stable install -y serviced-resource-agents
```

Configure the Cluster Management Software

Start and enable the PCS daemon on both nodes of the cluster:

```
# systemctl start pcsd.service
# systemctl enable pcsd.service
```

The Pacemaker installation creates a user account called *hacluster*. This account must have the same password on both boxes. Set the account password before proceeding:

```
# passwd hacluster
```

NOTE: All preceding steps must be completed on BOTH nodes in the cluster BEFORE continuing to the next step.

Create the Cluster in Standby Mode

To create a new cluster, you must authenticate each node with the following command. When prompted, enter the password for the *hacluster* user on any one of the nodes:

```
# pcs cluster auth <NODE1-PUBLIC> <NODE2-PUBLIC>
```

After the cluster nodes authenticate with each other, use the following command on the same node to generate and synchronize an initial (empty) cluster definition:

```
# pcs cluster setup --name serviced-ha \
        <NODE1-PUBLIC> <NODE2-PUBLIC>
```

NOTE: Unless noted otherwise, all of the remaining commands can be executed on either node.

To configure the cluster, start the pcs management agents on all nodes in the cluster. At this stage, the cluster definition is empty, so starting the cluster management agents will have no side effects. Use the following command on any one of the nodes to start the cluster management agents on all of the nodes:

```
# pcs cluster start --all
```

At this point the cluster should be running with both nodes, but because there are no resources defined yet, Pacemaker has nothing to actively manage. To check the status of the cluster at this point, execute the following command:

```
# pcs cluster status
```

Both nodes should be *Online*.

The next sections describe cluster configuration using a series of different pcs commands. By default, Pacemaker starts monitoring and managing the different resources as they are defined, even though all of the resources for our cluster will not be defined until the last one is added. To prevent inadvertent problems caused by incomplete resource definitions, use the following command to put both nodes of the cluster in standby mode.

```
# pcs cluster standby --all
```

NOTE: You have choices for starting the PCS at machine boot. The choices are to *enable autostart*, or *start manually.* To start manually, use the command `pcs cluster start --all` after the machine restarts. Zenoss recommends using the optional manual method for post-mortem investigation of potential problems after a reboot.

To *autostart* the cluster services after reboot, use the following commands:

```
systemctl enable corosync
systemctl enable pacemaker
```

For additional information, see the following document:
http://clusterlabs.org/doc/en-US/Pacemaker/1.1/html/Clusters_from_Scratch/ch04.html

For the purposes of Pacemaker, nodes in standby mode are not used for cluster operations such as failover. Although the rest of the processes on both nodes will continue running normally, the cluster-managed resources will not be active on any nodes in the standby mode. When the configuration process is complete, we can take the nodes out of standby one by one, in a controlled fashion, to verify that the configuration and a manual failover works.

Set Property and Resource Defaults

Pacemaker can implement a wide variety of large and complex cluster configurations. For the purposes of Control Center HA, this document describes a relatively simple two-node, active/passive configuration.

Before creating any resources, you must set some defaults that apply to the entire cluster for a basic two-node, active/passive configuration. Set the following defaults:

- `resource-stickiness=100` -
  This keeps all resources bound to the same host.
- `no-quorum-policy=ignore` -
  Pacemaker supports the notion of a "voting" quorum for clusters of three or more nodes. However, in our case, because we have only two nodes, if one of the nodes fails, trying to use the remaining node as a quorum of one does not make sense; therefore, we disable quorums.
- `stonith-enabled=false` -
  "stonith" is an acronym for "shoot the other node in the head"; it is used for "fencing" or isolating a failed node. Stonith should only be disabled for the initial setup and testing period. This example uses a value of *false* for simplicity and to aid in verifying an initial configuration. Production deployments should enable and configure a machine/environment-specific stonith (this means set `stonith-enabled=true`). See Appendix A: Fencing for more details.

Use the following commands to set the defaults:

```
# pcs resource defaults resource-stickiness=100
# pcs property set no-quorum-policy=ignore
# pcs property set stonith-enabled=false
```

After setting the defaults, you can run the following commands and consult the output to verify they were applied correctly. For example:

```
# pcs property
Cluster Properties:
 cluster-infrastructure: corosync
 cluster-name: serviced-ha
```

```
 dc-version: 1.1.12-a14efad
 have-watchdog: false
 no-quorum-policy: ignore
 stonith-enabled: false
```

```
# pcs resource defaults
```

```
 resource-stickiness: 100
```

Define Resources

Seven logical resources need to be defined for the cluster:

- The DRBD Master/Slave DFS set
- The two mirrored file systems running on top of DRBD:
  `/opt/serviced/var/isvcs`
  and
  `/opt/serviced/var/volumes`
- The floating IP address that the cluster will assign to the active primary cluster node
- Docker
- NFS
- Control Center

Define the DRBD Master/Slave set

Define a cluster resource for the DRBD device, and an additional clone of that resource to act as the master. For example:

```
# pcs resource create DFS ocf:linbit:drbd \
      drbd_resource=serviced-dfs \
      op monitor interval=30s role=Master \
      op monitor interval=60s role=Slave

# pcs resource master DFSMaster DFS \
      master-max=1 master-node-max=1 \
      clone-max=2 clone-node-max=1 notify=true
```

Note that for a master/slave resource, Pacemaker requires separate monitoring intervals for the different roles. In this case, the configuration tells Pacemaker to check on the master every 30 seconds and the slave every 60 seconds.

Define File System Resources

Define the filesystems that are mounted on the DRBD devices. For example:

```
# pcs resource create serviced-isvcs Filesystem \
      device=/dev/drbd/by-res/serviced-dfs/0 \
      directory=/opt/serviced/var/isvcs fstype=ext4
```

```
# pcs resource create serviced-volumes Filesystem \
      device=/dev/drbd/by-res/serviced-dfs/1 \
      directory=/opt/serviced/var/volumes fstype=btrfs
```

Note that in the commands above, `serviced-dfs` is the name of the DRBD resource defined previously in `/etc/drbd.d/serviced-dfs.res`.

Define the Floating IP Resource

Define the resource that represents the floating virtual IP address of the cluster using the following command.

```
# pcs resource create VirtualIP ocf:heartbeat:IPaddr2 \
   ip=<CLUSTERVIP> nic=<vip-nic> \
   cidr_netmask=32 op monitor interval=30s
```

Note: In the command above, use the *name of the network interface* that is bound to `<CLUSTERVIP>` instead of the place holder ***<vip-nic>.***

Define the Docker Resource

Issue the following command to define your Docker resource:

```
# pcs resource create docker systemd:docker
```

NOTE: Because Pacemaker will manage Docker startup and shutdown, the normal auto-start of the Docker `systemd` unit should be disabled. Although this should have been accomplished in the preceding steps that installed Control Center, in case it was overlooked use the following command to stop the service and disable Docker startup at boot time:

```
# systemctl stop docker && systemctl disable docker
```

Define the NFS Resource

Control Center uses NFS to share configuration in a multi-host deployment. For failover to work properly, we need to ensure that NFS is stopped on a failed node to completely disconnect any client hosts before restarting on another node. For example:

```
# pcs resource create nfs systemd:nfs
```

NOTE: Because Pacemaker will manage NFS startup and shutdown, the normal auto-start of the NFS `systemd` unit should be disabled. Although this should have been accomplished in the preceding steps that installed Control Center, in case it was overlooked use the following command to stop the service and disable NFS startup at boot time:

```
# systemctl stop nfs && systemctl disable nfs
```

Define the Control Center Resource

Issue the following command to define your Control Center resource:

```
# pcs resource create serviced ocf:zenoss:serviced
```

NOTE: Because Pacemaker manages the Control Center startup and shutdown, the normal auto-start of the Control Center `systemd` unit should be disabled. Although this should have been accomplished in the preceding steps that installed Control Center, in case it was overlooked, use the following command to stop the service and disable Control Center startup at boot time:

```
# systemctl stop serviced && systemctl disable serviced
```

Pacemaker (cluster management software) uses the default timeouts defined by the Pacemaker Resource Agent for Control Center to decide if `serviced` is unable to start or shutdown correctly. Starting with version 0.0.3 of the Pacemaker Resource Agent for Control Center, the default values for the start and stop timeouts are 360 and 90 seconds respectively.

If you are installing a new HA cluster for Control Center 1.0.7 or higher with the Pacemaker Resource Agent for Control Center version 0.0.3 or higher, you do not need to adjust the start or stop timeouts.

If you are upgrading an existing HA cluster to Control Center 1.0.7, you must also upgrade the Pacemaker Resource Agent for Control Center to version 0.0.3, and you must manually increase the start timeout. See the section titled *Upgrading serviced Resource Agent*.

Note that the default startup and shutdown timeouts are based on the worst case scenario. In practice, Control Center will typically start and stop in much less time. This does not mean however that you should decrease these timeouts. There are potential edge cases, especially for startup, where Control Center may take longer than usual to start or stop. If the start/stop timeouts for Pacemaker are set too low, and Control Center encounters one of those edge cases, then Pacemaker will take unnecessary or incorrect actions. For example, if the startup timeout is artificially set too low,2.5 minutes for example, and Control Center startup encounters an unusual case where it requires at least 3 minutes to start, then Pacemaker will failover prematurely.

Define the Control Center Resource Group

The resources in a resource group are started in the order they appear in the group and stopped in the reverse order they appear in the group. In this case, we want to:

1. Mount the file systems
2. Enable the Virtual IP
3. Start Docker
4. Start NFS
5. Start Control Center

In the event of a failover, Pacemaker stops the resources on the failed node in the reverse order they are listed here before starting the resource group on the standby node.

Use the following command to create a resource group that implements those ordering rules:

```
# pcs resource group add serviced-group \
      serviced-isvcs serviced-volumes \
      VirtualIP docker nfs \
      serviced
```

Define Resource Constraints

Pacemaker resource constraints control when and where resources are deployed in a cluster. Enter the following commands to keep the `serviced-group` running on the same node as the *DFSMaster* and ensure that the group is only started after the *DFSMaster* is started:

```
# pcs constraint colocation add serviced-group with DFSMaster \
      INFINITY with-rsc-role=Master

# pcs constraint order promote DFSMaster then \
      start serviced-group
```

Verification

In the *Create the Cluster in Standby Mode* section above, all nodes in the cluster were placed in a 'standby' state while various configuration changes were applied to the cluster. Before enabling the cluster for the first time, review the configuration and make adjustments as necessary. The following sections describe the review process.

Verify DRBD Configuration

Use the following command to dump the full DRBD configuration. The output should be consistent with the changes made for the Steps 1 and 2 of the section titled *Configure DRBD*, on page 9, above.

```
# drbdadm dump
```

Use the following the command to review the synchronization status of the both volumes on both servers:

```
# drbd-overview
```

NOTE: Do **NOT** proceed until the synchronization is complete. The process is complete when the status of both devices on both nodes is *UpToDate/UpToDate*.

Verify Pacemaker Configuration

Review the property and resource defaults for pacemaker. Issue the following command and consult the output. For example:

```
# pcs property
Cluster Properties:
 cluster-infrastructure: corosyncv
 cluster-name: serviced-ha
 dc-version: 1.1.12-a14efad
 have-watchdog: false
 no-quorum-policy: ignore
 stonith-enabled: false
# pcs resource defaults
resource-stickiness: 100
```

NOTE: The value of the `stonith-enabled` property should be ***true*** for production environments with fencing. Although setting the value to *false* is not recommended for production environments, it can be used in controlled test environments. See Appendix A: Fencing for more information.

Review the resource constraints to verify that `serviced-group` resource is started after the *DFSMaster* (the DRBD master), and both the `serviced-group` and DFSMaster are colocated on the same active cluster node. Issue the following command and consult the output. For example:

```
# pcs constraint
        Location Constraints:
        Ordering Constraints:
          promote DFSMaster then start serviced-group (kind:Mandatory)
        Colocation Constraints:
          serviced-group with DFSMaster (score:INFINITY) (with-rsc-role:Master)
```

Use the following command to review the resource definitions:

```
# pcs resource show --full
```

The order of resources in the resource group serviced-group (from top to bottom) is important because this is the order the resources will be started by Pacemaker. The order should be:

1. serviced-isvcs
2. serviced-volumes
3. VirtualIP
4. Docker
5. nfs
6. serviced

Verify Control Center Configuration

Verify that Control Center configuration files are identical by running the following command on both nodes and comparing the values:

```
# cksum /etc/default/serviced
```

If the files are identical, their checksums should be identical. If they differ, inspect their contents using the following command:

```
# grep ^SERVICED /etc/default/serviced | sort
SERVICED_AGENT=1
SERVICED_ENDPOINT=<CLUSTERVIP>:4979
SERVICED_FS_TYPE=btrfs
SERVICED_MASTER=1
SERVICED_OUTBOUND_IP=<CLUSTERVIP>
SERVICED_REGISTRY=1
SERVICED_VARPATH=/opt/serviced/var
```

Verify Cluster Startup

To verify the initial configuration, attempt startup of the resources on a single-node with the other node in standby mode. If there is a problem, you have an opportunity to pause and diagnose it without Pacemaker automatically stopping the entire node, and failing over to the other one.

Assuming both nodes are currently in a standby state, determine which node is currently the DRBD master. Issue the following command:

```
# drbd-overview
```

```
 0:serviced-dfs/0  Connected Primary/Secondary UpToDate/UpToDate
/opt/serviced/var/isvcs   ext4  4.8G 196M 4.4G 5%

 1:serviced-dfs/1  Connected Primary/Secondary UpToDate/UpToDate
/opt/serviced/var/volumes btrfs 45G  544K 43G  1%

#
```

In the example above, the current machine is the DRBD master as indicated by the string ***Primary/Secondary*** in the output. If the current machine is instead the DRBD slave, the string will be ***Secondary/Primary***.

To startup the cluster resources, run the following command.

```
# pcs cluster unstandby <node1>
```

In the command, **<node1>** is the hostname of the DRBD master. Note that you can run pcs commands on any node in the cluster.

You can check the status of cluster resources with the following command:

```
# pcs status
```

Because it can require some time for all of the resources to start, repeat the `pcs status` command until all resources report a status of `Started`.

If one or more resources fail to start, resolve the issue before continuing.

Verify Cluster Failover

When all resources have started successfully, you can enable the other node and simulate a failover. To enable the other node, run the following command:

```
# pcs cluster unstandby <node2>
```

In the command, `<node2>` is the hostname of the DRBD slave node.

Use the pcs status command periodically to verify that the slave node is `Online` before continuing.

When the secondary cluster node is `Online`, you can force a manual failover by putting the currently primary cluster node into standby. For example:

```
# pcs cluster standby <node1>
```

Monitor the cluster status periodically with the `pcs status` command. Because it can require some time for all of the services to failover, repeat the `pcs status` command until all resources report a status of `Started` on <node2>.

If the failover does not succeed, resolve the problem before continuing.

Configure Control Center

A dedicated pool (or pools) of resource pool agents are used to run RM, separate from Control Center master agent(s) in the HA cluster. In this example, we will assign the Control Center master agents to the default pool, and create an additional pool (or pools) to run the RM services.

Populate the Default Pool with HA Nodes

From a browser:

1. Log into Control Center using the hostname for the floating VIP.
2. Navigate to the *Hosts* page.
3. Add a host using the public hostname of the active cluster host.
   Use the public hostname assigned to that node of the cluster (for example, `NODE{1,2}-PUBLIC`).
   For *Resource Pool*, use `default`.
4. From the command line of one of the cluster machines, use the `pcs cluster standby <activeNode>` command to put the active node in *standby* mode.
5. Trigger a manual failover to the other node.
6. From a browser, log into Control Center using the hostname for the floating VIP.
7. Navigate to the *Hosts* page.
8. Add a host using the public hostname of the new, active cluster host.
   Use the public hostname assigned to that node of the cluster (for example `NODE{1,2}-PUBLIC`).
   For Resource Pool, use `default`.

At this point the default pool should have 2 entries, one for each of the public hostnames for the two nodes in the HA cluster (for example, `NODE1-PUBLIC` and `NODE2-PUBLIC`).

NOTE: Using the two public hostnames for the master agents in the default pool enables you to tell which node is currently active in the HA cluster when you login to the Control Center UI.

Create Pool(s) for RM

The HA configuration for Control Center requires a dedicated pool of resource agents to run RM. None of the services of RM itself should be run on the CC master agents in an HA cluster.

From a browser:

1. Log into Control Center using the hostname for the floating VIP.
2. Navigate to the `Resource Pools` page
3. Add one or more pools.
   Note: The hosts for those pools will be added in sections below.

Install Control Center Resource Pool Agent(s)

To provide HA for CC resource pools, each pool must have at least N+1 hosts where N is the number of hosts needed to satisfy the performance and scalability requirements for that pool. For example, if a single resource pool agent is sufficient to run all of the services for that pool, you require two agents in the pool to provide HA for that pool. Alternatively, if another resource pool requires two resource pool agents to handle its load, at least three agents are required in that resource pool to provide HA for the pool.

Follow the standard procedure for preparing and installing resource pool hosts for Control Center with the following difference - in the step titled *Configure the Resource Pool Host for a Multi-Host Deployment* of the *Installing on a resource Pool Host* section of the ***RM Install Guide***, where it discusses setting MHOST, use the value of the floating VIP for the cluster as the value for `MHOST`.

Verify that `/etc/default/serviced` contains the appropriate contents. Run the following command and consult the output:

```
# grep ^SERVICED /etc/default/serviced | sort
SERVICED_AGENT=1
SERVICED_DOCKER_REGISTRY=<CLUSTERVIP>:5000
SERVICED_FS_TYPE=btrfs
SERVICED_LOG_ADDRESS=<CLUSTERVIP>:5042
SERVICED_MASTER=0
SERVICED_REGISTRY=1
SERVICED_STATS_PORT=<CLUSTERVIP>:8443
SERVICED_VARPATH=/opt/serviced/var
SERVICED_ZK=<CLUSTERVIP>:2181
```

After the Resource Agent(s) start, add them to the resource pools(s) created in the section *Create Pool(s) for RM*, on page 20. Do NOT add them to the default resource pool containing the CC master agents in the HA cluster.

Deploy Resource Manager

Follow the instructions in the *Deploying the Resource Manager* section of the ***RM Install Guide*** to load and start RM with the following difference - when selecting a resource pool, use the pool created in the section titled *Create Pool(s) for RM*, on page 20. above, NOT the default resource pool.

Gracefully Shutdown/Startup Cluster Services

Should it become necessary to shut down your cluster services, perform the following:

1. Log into Control Center
2. Stop the RM application
3. Issue the command:

   ```
   pcs resource disable serviced-group
   ```

4. Verify the shutdown status.
   Use the `pcs status` command and consult the output until all services are shut down.

To bring your cluster services back online, perform the following:

1. Issue the following command on each node:

   ```
   drbdadm up all
   ```

2. Issue the following command on the node you want to be primary:

   ```
   drbdadm primary serviced-dfs
   ```

3. Issue the following command on one node (your choice):

   ```
   pcs cluster start --all
   ```

4. Issue the following command on the primary node:

   ```
   pcs resource enable serviced-group
   ```

5. Verify the cluster status. Use the `pcs status` command and consult the output until all services are up.

Enable Fencing

When all components are deployed, and you have verified operation of RM and basic cluster failover in a controlled scenario, configure and enable fencing.

See Appendix A: Fencing for more information.

Upgrading Control Center and Resource Manager

To upgrade Control Center and Resource Manager, follow the instructions in the Zenoss *Resource Manager Upgrade Guide* with the following notes & exceptions.

Downloading Images

Complete the steps for "*Downloading Upgrade Images*" on the active node.
Control Center and Resource Manager can both be running for this step.

This step re pre-pulls the docker images for the upgraded software and saves them to `/var/lib/docker`. Because this is not replicated by drbd, these steps must be completed on both nodes.

On the backup node, `docker` must be started first:

```
systemctl start docker
```

When the download is complete, stop `docker` on the backup node:

```
systemctl stop docker
```

Upgrading serviced Resource Agent

If you have an existing HA cluster and you are upgrading to Control Center *1.0.7*, you must upgrade the `serviced-resource-agent` RPM package to version *0.0.3*. Additionally you must manually adjust the *start timeout* for the `serviced` resource.

To verify the installed `serviced` version, run the following command:

```
rpm -qa |grep serviced
```

To upgrade your `serviced` resource agent, run the following command:

```
yum --enablerepo=zenoss-stable install -y serviced-resource-agents
```

To manually adjust the *start timeout* for `serviced` to the recommended value of **6** minutes, run the following command:

```
pcs resource update serviced op start timeout=360s
```

Upgrading Control Center

Note that if you are upgrading Control Center to version 1.0.7, you must also upgrade the `serviced-resource-agent` RPM package to version *0.0.3*. Additionally, if you previously manually set the time out values for `serviced`, you must manually update that value to the recommended value of *6 minutes. For* information about how to perform these operations, see the section titled *Upgrading serviced Resource Agent.*

In the *Upgrading Control Center* section, disable the cluster services in place of the step that reads "systemctl stop serviced":

```
pcs resource disable serviced-group
```

Complete the step for "Upgrading Control Center on the master host" on both nodes.

In place of the step that reads "*Start Control Center*," enable the cluster resource group:

```
pcs resource enable serviced-group
```

Note that when the command '`serviced host list`' is run, the standby node in the cluster will still show the prior version of `serviced`.

To complete the upgrade, fail over from the primary node to the standby node:

```
pcs cluster standby <primary node>
```

Note that after the failover, the standby node will now show the updated `serviced` version when the command '`serviced host list`' is run.

Upgrading Resource Manager

On the active node, follow the instructions in the Zenoss *Resource Manager Upgrade Guide*.

Converting Docker Storage Driver

If you need to convert the Docker Storage Driver, follow the directions in the Zenoss *Resource Manager Upgrade Guide* with the following notes / exceptions:

On the active node of the cluster, run the command:

```
serviced service stop Zenoss.resmgr.
```

Substitute the command "`systemctl stop serviced && systemctl stop docker`" with the command to disable the resources on the active node:

```
pcs resource disable serviced-group
```

Complete the subsequent steps on both master cluster nodes.

Start `docker` on each node (independent of enabling the cluster resources) only to test a successful startup. Be sure to stop `docker` after a successful test on each node:

```
systemctl stop docker
```

In place of the "*Starting Control Center and Resource Manager*" re-enable the cluster resources on the primary node:

```
pcs resource enable serviced-group
```

Appendix A: Fencing

Fencing is an automated means of isolating a node that appears to be malfunctioning to protect the integrity of the DRBD volume(s).

We recommend placing the fencing device on the public network. The reason for placing the fencing device on the public network is explained by how fencing and communication work in various implementations. In this case the two implementations are a *public* versus a *private* network interface for fencing communications.

Note: If you do not define a fencing method, your failover will fail with an error "no method defined" when the cluster attempts to fail over to the backup node.

Using a Private Network Interface

Although it is possible to pass heartbeat communications through the private network interface, it requires a complex fencing mechanism. (See the *Implementing/Deploying Quorum Disk on RHCS* manual for more information.) The complex fencing mechanism is prone to issues. Consider, for example, a heartbeat communication that passes through the private network interface. If the private network link fails for either of the nodes, heartbeat communications fail. Each node perceives the other as offline although the active node is still online from the point of view of the user. Because the nodes perceive each other to be off line, each machine initiates fencing of the other node. Both nodes can access the fencing device because the public network links of both nodes are still functioning. The fencing device successfully fences, or shuts down both machines. Fencing both nodes leaves the cluster without a healthy, functioning node online. The complicated work around for this scenario is:

1. Move the fencing device to the private network.
2. Renumber the fencing device.
3. Reconfigure systems that use the fencing device.

Using a Public Network Interface

If heartbeat communications pass through the public network and the public network link for a node goes down, both nodes still try to fence each other. The difference in this case however is that the node with the down public network link cannot communicate with the fencing device. This means that the healthy node can fence the unhealthy node, but the unhealthy node cannot fence the healthy node.

Appendix B: Handling DRBD Errors

For information about handling DRDB errors, see *The DRBD User's Guide* at http://www.drbd.org/users-guide/.

Split Brain

The condition of (DRDB) split brain is caused by both nodes switching to the *primary* role due to network disconnection. This situation is the result of a temporary network failure between cluster nodes. It can be caused by such things as human error or cluster management software actions. This is potentially harmful to your data because data modifications might have occurred on either of the two nodes and not replicated to the peer. This means the data is out of synchrony because it is now seen as two diverging sets of data. It is not trivial to merge and recover from these changes.

Recognizing Split Brain

The symptoms of a split-brain are:

- the peers will not reconnect on DRBD startup but stay in a connection state of *StandAlone* or *WFConnection*.
- kernel logs will have messages like:

```
Split-Brain detected, dropping connection!
```

For information on resolving split brain issues specifically, see the following documents:

- http://www.drbd.org/users-guide/s-resolve-split-brain.html
- https://www.hastexo.com/resources/hints-and-kinks/solve-drbd-split-brain-4-steps

How to Resynchronize the Disks

If the disks within the cluster fall out of synchrony, perform the following procedure to resynchronize them.

CAUTION: You must determine if the issue is a split-brain problem before you follow this procedure. If a split-brain problem exists, follow the split-brain procedure instead.

Note: This procedure if incorrectly performed WILL cause LOSS OF DATA.

To reset data and resynchronize the disks:

1. Log into Control Center
2. Stop the RM application
3. Issue the command:

```
pcs resource disable serviced-group
```

4. Verify the shutdown status.
   Use the `pcs status` command and consult the output until all services are shut down.
5. Issue the following commands on *both* nodes:

```
drbdadm up serviced-dfs
drbdadm adjust all
```

6. Issue the following commands on the node with data you want to keep:

```
drbdadm primary --force serviced-dfs
drbdadm invalidate-remote all
drbdadm primary all
```

7. Wait for the system to synchronize.

   The synchronization process is not instant. These commands only start the syncing process.

   You can monitor the progress by periodically using the command `drbd-overview`. Depending on the volume of data this synchronization can require several minutes. Do NOT start the cluster until both volumes are synced.

   When the output displays `Primary/Secondary UpToDate/UpToDate`, the disks are fully mirrored and ready for use

   Restart the cluster and services in the cluster.

8. Issue the following command on each node:

```
drbdadm up all
```

9. Issue the following command on the node you want to be primary:

```
drbdadm primary serviced-dfs
```

10. Issue the following command on one node (your choice):

```
pcs cluster start --all
```

11. Issue the following command on the primary node:

```
pcs resource enable serviced-group
```

12. Verify the cluster status.
    Use the `pcs status` command and consult the output until all services are up.

Appendix C: Command Summary

Although a full summary of the commands used to administer DRBD and Pacemaker is the beyond the scope of this document, this appendix provides a brief overview of some of the more commonly used commands.

DRBD

Review DRBD configuration

The following command dumps the current DRBD configuration:

```
drbdadm dump
```

The output shows the consolidated union of default values and values configured in `/etc/drbd.conf` and `/etc/drbd.d/serviced-dfs.res`.

Check DRBD Status

There are two ways (commands) to display the status of DRBD:

```
cat /proc/drbd
```

or

```
drbd-overview
```

Pacemaker

Review Pacemaker configuration

There are specific commands to review different aspects of the Pacemaker configuration.

- The following command shows all of the details for all of the resources and resource groups:

  ```
  pcs resource show --full
  ```

- The following command shows the values of the global properties:

  ```
  pcs property
  ```

- The following command shows the values of the global resource defaults:

  ```
  pcs resource defaults
  ```

- The following commands shows all of the PCS constraints:

  ```
  pcs constraint show --full
  ```

Cluster status

To check on the current status of the cluster, you can use one of the following commands:

```
pcs status
```

or

```
crm_mon -1Arf
```

To monitor the cluster status continually, use the following command:

```
crm_mon -Arf
```

To cleanup PCS error messages for a failed single resource, use the following command:

```
pcs resource cleanup <resource name>
```

where *<resource name>* is the name of one of the resources in the cluster; for example `serviced`.

To review pacemaker log files, use the following command:

```
journalctl -u pacemaker -o cat -f
```

Cluster Operations

To manually deactivate a node in the cluster, issue the following command:

```
pcs cluster standby <nodename>
```

NOTE: It typically requires several seconds for `serviced`, and hence the entire cluster, to completely start and stop. Monitor the cluster status after issuing a cluster standby command to determine when Pacemaker has stopped all of the services on one node. Assuming the slave or passive node is online, Pacemaker will activate the passive node, and start the necessary resources.

To manually reactivate a node in the cluster

```
pcs cluster unstandby <nodename>
```

To stop PCS-managed resources on the entire cluster, use the following command:

```
pcs cluster stop --all
```

To restart PCS and PCS-managed resources on the entire cluster, use the following command:

```
pcs cluster start --all
```

In some scenarios, you might want to modify aspects of the cluster configuration and apply them all at once or at some later time. Pacemaker supports this option through capturing the current configuration to a file and applying the changes from one or more pcs commands to that file. It then commits the file to apply the changes to the running cluster.

To capture the current configuration in a file, use the following command:

```
# pcs cluster cib cluster_cfg
```

To make changes to the file, use the `-f` argument for any of the pcs resource or pcs property commands.

```
# pcs -f cluster_cfg …
```

When you finish making changes to the file, apply the changes to the current cluster with the following command:

```
# pcs cluster cib-push cluster_cfg
```

Appendix D: References

The following are references for this document:

- ClusterLab's Clusters from Scratch, edition 8.
- Linbit's DRBD User's Guide, version 8.4
- Pacemaker Cluster Resource Manager - http://clusterlabs.org/pacemaker.html
- Corosync Cluster Communication - http://www.corosync.org
- Hastexo's Cluster Management Knowledge Base - https://www.hastexo.com/resources/hints-and-kinks