High Availability Active-Passive sipXecs Implementation using CentOS clustering

This is legacy 4.4 and earlier documentation

High Availability (Active/Passive) sipXecs implementation using existing iSCSI shared storage and CentOS clustering.

Caveats

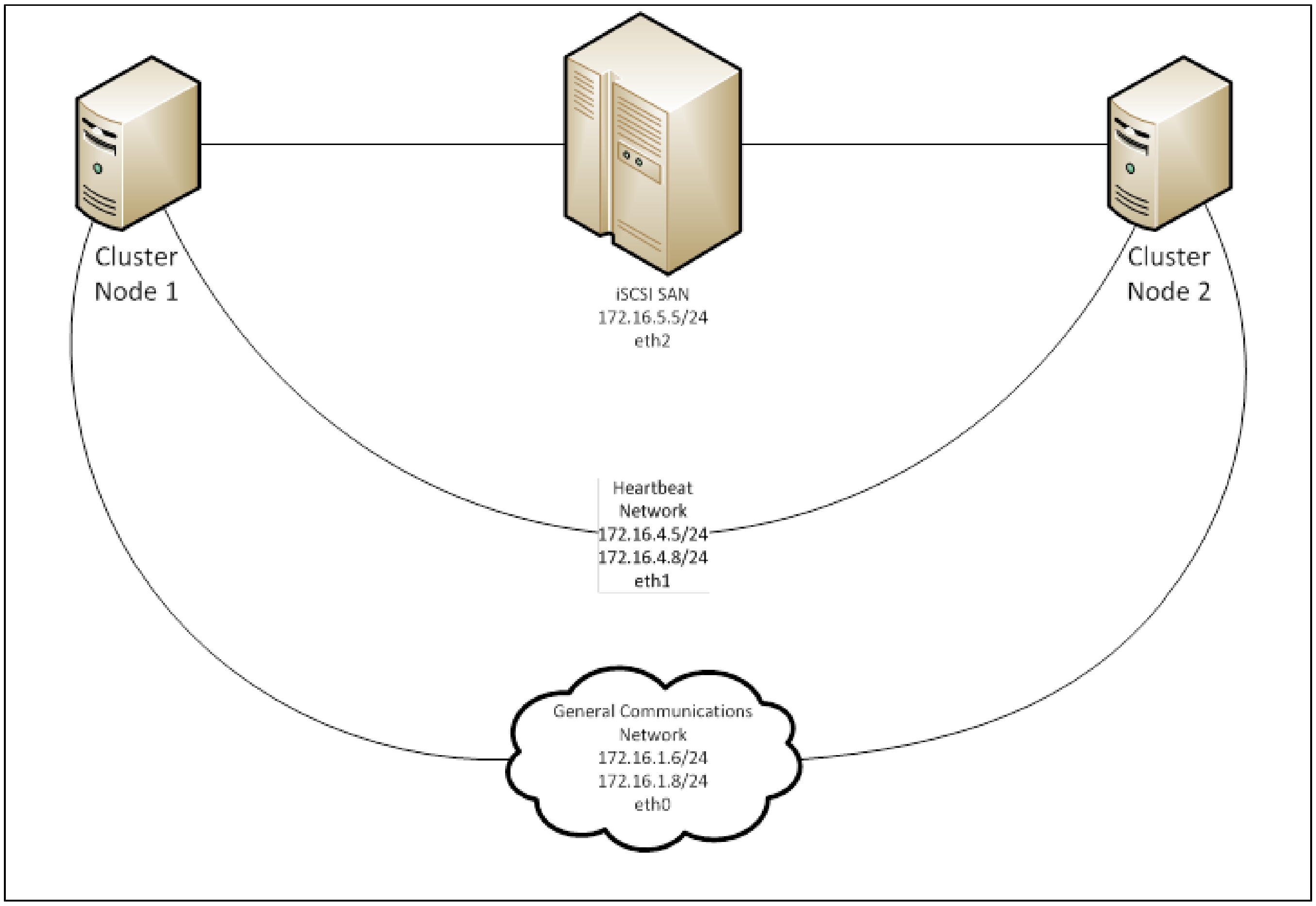
- It should be noted this is a highly complex topic.
- This document does not cover all aspects of clustering and failover scenarios and is not intended to go through the exact installation steps.
- It is advisable to keep the cluster nodes on their own subnet as each node broadcasts "heartbeats" to the other node to ensure it is still alive.

System requirements

- Two high power servers (8 GB RAM minimum) with three NICs in each
- Dedicated networks for each service
- Fencing device on each server such as Dell DRAC or HP iLO. This is so if a cluster node is misbehaving it is immediately powered off instead of causing data corruption as well as to trigger proper failover.

Example Diagram

The following is a diagram of the clustering scenario we will be setting up:



Install OS

Install CentOS 5.6 on two identical servers. You may also configure the NICs at this point. One NIC will be for the iSCSI storage network (eth2), one will be for clustering heartbeat (eth1), and one NIC will be for all other communications, including sipXecs (eth0).

When setting the default gateway, it must be set on the main communication interface (eth0). This is done automatically in initial setup but can be changed to the detriment of the system post install. **DO NOT** set the default gateway on any interface other than the primary communication interface (eth0).

It is also advisable to set the hostname at this time. The hostname **MUST** be the same on both systems otherwise sipXecs will fail to function if switched to another node.

When prompted for software selection, deselect **desktop – Gnome** and select **Clustering** and **Storage Clustering**. Perform the install.

Post Installation

After each system reboots you'll need to disable the firewall and turn of SELinux. This is done when the Setup Agent appears. Choose **Firewall**, then disable all security settings.

After you have disabled the security settings perform a system update on both systems:

```
yum update
```

After the update reboot both systems. Once you've rebooted, install the software package **xauth** to enable X11 forwarding (used later):

```
yum install xauth
```

Connect to Shared iSCSI disk

Each server needs to have iSCSI turned on to connect. Run the following commands to enable iSCSI:

```
chkconfig iscsi on
service iscsi start
```

On each server you need to connect to a blank (fresh) shared iSCSI disk. For example, if your iSCSI shared disk is located at IP address **172.16.5.10** you would run the following command on each server:

```
iscsiadm -m discovery -t sendtargets -p 172.16.5.10
service iscsi restart
```

You will now see the addition of a drive to the server, usually **/dev/sda** or something similar.

Configure LVM to allow for filesystem clustering

Edit **/etc/lvm/lvm.conf** on each node and change the following line:

```
locking_type = 1
```

to

```
locking_type = 3
```

and run the following command for this change to take effect:

```
vgscan
```

Add Clustered Network Script

Create a file on both nodes called **/usr/local/bin/clusnet** and fill it will the following, modifying **SIPX_IP='172.16.1.5'** on line 15 to the be the IP address of the sipXecs system, as well as the Ethernet interfaces for your services:

```
#!/bin/bash
#
#
#
#       Simple script for cluster services to use for bare metal ethernet interface
#
#  chkconfig: 345 89 14
#  description: Starts and stops the clustered service network interface
#  processname: net
#

# Source function library.
. /etc/init.d/functions

SIPX_IP='172.16.1.5'
SIPX_INTERFACE=eth0
CLUS_INTERFACE=eth2
ISCSI_INTERFACE=eth1
PROG_NAME=clusnet
RETVAL=0

DEFAULT_GW=`cat /etc/sysconfig/network-scripts/ifcfg-$SIPX_INTERFACE | grep GATEWAY | sed 's/GATEWAY=//g'`
SUBNET_MASK=`cat /etc/sysconfig/network-scripts/ifcfg-$SIPX_INTERFACE | grep NETMASK | sed 's/NETMASK=//g'`
INTUP=`/sbin/ifconfig | grep -c $SIPX_INTERFACE`;

# <define any local shell functions used by the code that follows>

start() {
        echo -n "Starting $SIPX_INTERFACE: "
        ip address flush dev eth0
        sleep 3
        ifconfig $SIPX_INTERFACE $SIPX_IP netmask $SUBNET_MASK up
        sleep 3
        route add default gw $DEFAULT_GW $SIPX_INTERFACE
        echo
        echo -n $"$SIPX_INTERFACE is up.";
        success $"$SIPX_INTERFACE is up.";
        echo
        chkconfig postgresql off
        return $RETVAL
}


stop() {
        echo -n "Shutting down $SIPX_INTERFACE: "
        ifdown $SIPX_INTERFACE
        ifup $SIPX_INTERFACE
        ifup $ISCSI_INTERFACE
        ifup $CLUS_INTERFACE
        echo
        echo -n $"Interface $SIPX_INTERFACE is down.";
        success $"Interface $SIPX_INTERFACE is down.";
        echo
        chkconfig postgresql off
        return $RETVAL
}

getstatus() {
            if [ $INTUP -lt 1 ]
            then
                echo
                echo -n $"Interface $SIPX_INTERFACE is down."
                failure $"Interface $SIPX_INTERFACE is down."
                echo
                return 1;
            fi
            echo
            echo -n $"Interface $SIPX_INTERFACE is up.";
            success $"Interface $SIPX_INTERFACE is up.";
            echo
            return $RETVAL
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        getstatus
        ;;
    restart)
        stop
        sleep 5
        start
        ;;
    *)
        echo "Usage: $PROG_NAME {start|stop|status|restart}"
        exit 1
        ;;
esac
exit $RETVAL
```

You'll also need to make the script executable by running:

```
chmod +x /usr/local/bin/clusnet*
```

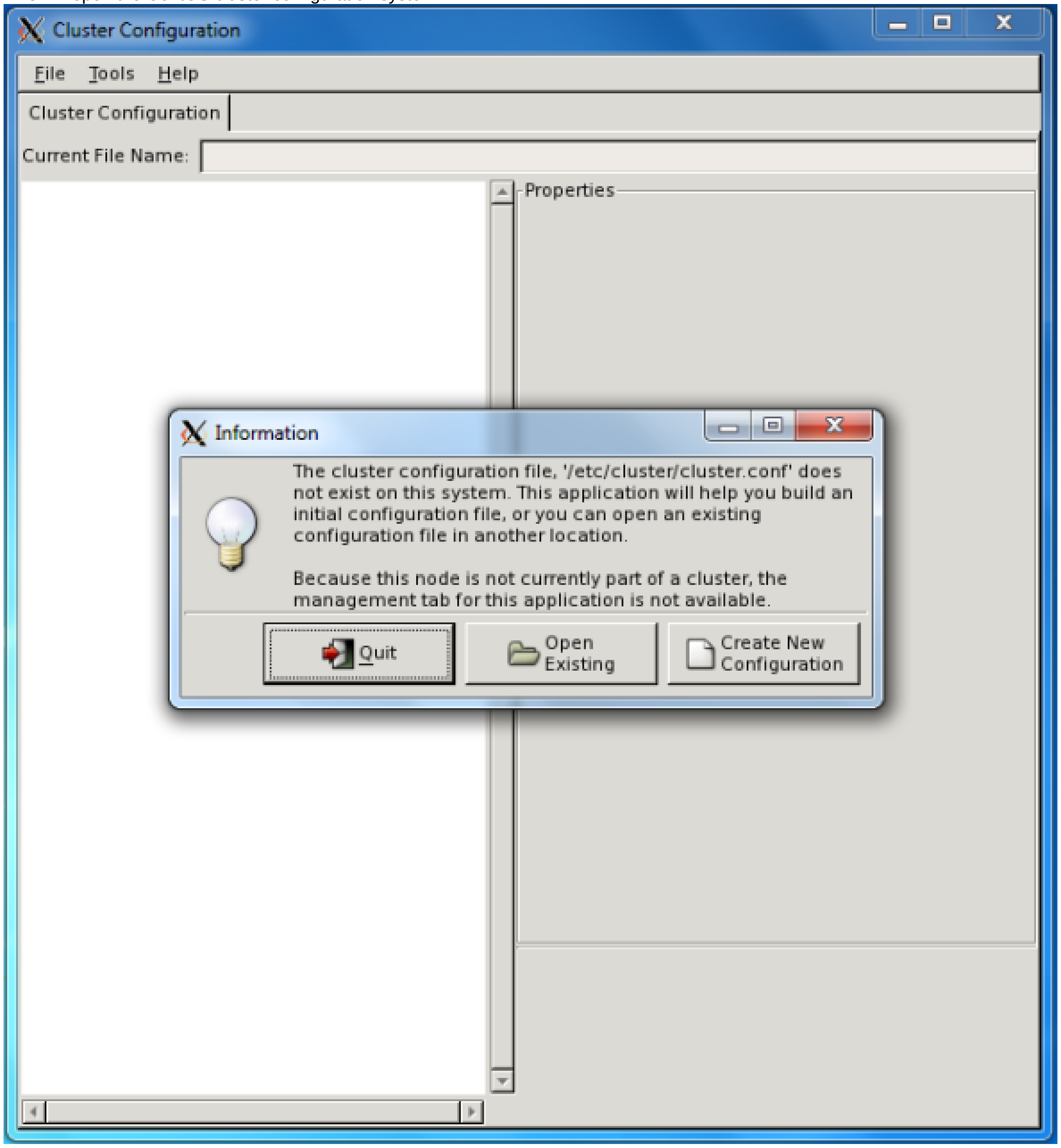
This script replaces your primary communication network's IP address with that of the sipXecs servers IP address. This is so there will not be any IP address conflict within sipXecs services.

Configure Clustering

Initial cluster configuration is done with the utility **system-configure-cluster** which, if you're running a headless (text only) install, will require the use of X forwarding via PuTTY and Xming. Download Xming from http://sourceforge.net/projects/xming/files/Xming/6.9.0.31/Xming-6-9-0-31-setup.exe/download and install it, then start it. In PuTTY, enable the option **Enable X11 Forwarding** under **Connection >> SSH >> X11.** You may then connect to the primary node server and run the following command:

```
system-config-cluster
```

This will open the CentOS cluster configuration system:

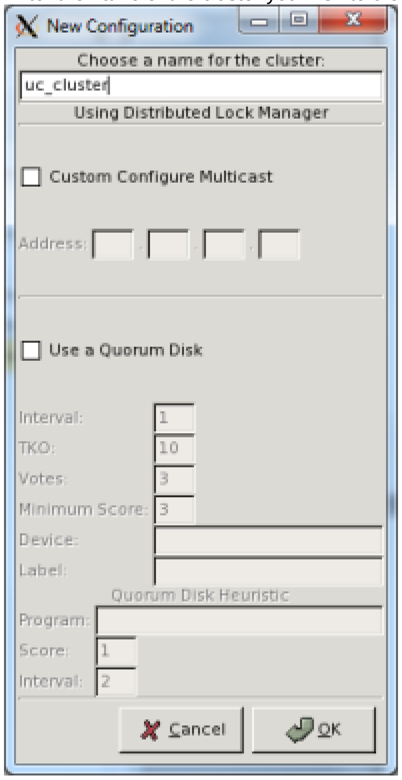


The cluster configuration file, '/etc/cluster/cluster.conf' does not exist on this system. This application will help you build an initial configuration file, or you can open an existing configuration file in another location.

Because this node is not currently part of a cluster, the management tab for this application is not available.

Click on **Create New Configuration** to begin configuring the cluster.

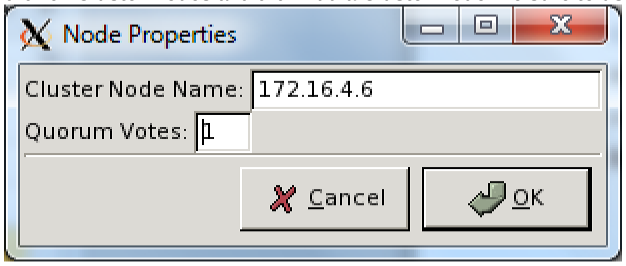
Enter the name of the cluster you wish to create. For this example we will create a cluster called **uc_cluster**:



For a simple two node cluster it is not necessary to use a quorum disk. Click **OK.**

Add Nodes

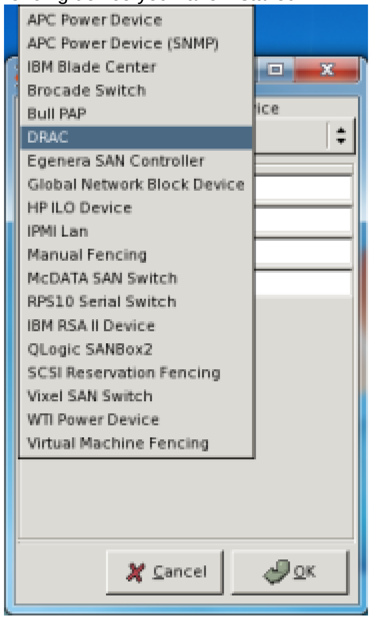
We now need to add all of our cluster information into the cluster configuration manager the first thing we need to do is add our cluster nodes. To do this click on **Cluster Nodes** and click **Add a Cluster Node**. **Be sure to use IP addresses and not DNS host names** :



Click **OK** then add the second node in the same fashion.

Add Fencing Device

In the system requirements section of this document it was noted that you would need some sort of manual fencing device to enable the cluster to power down a troublesome node. To add a fencing device, click on **Fence Devices** then click on **Add a Fencing Device**. You will now need to select the type of fencing device you have installed:

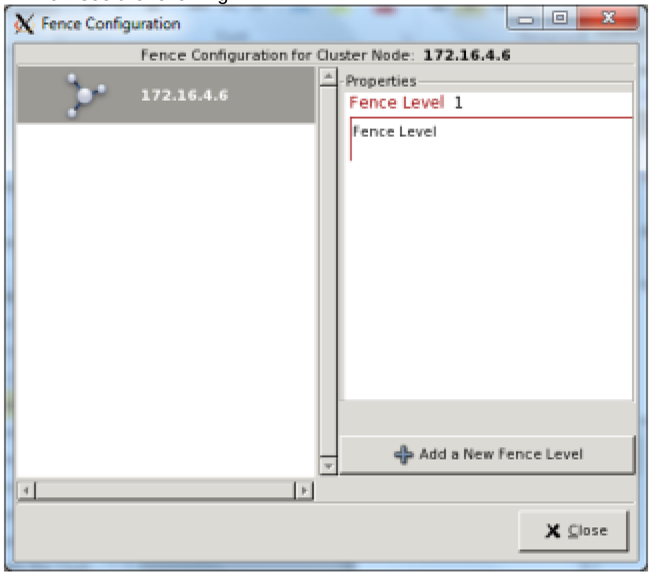


Once you have selected your fencing device you will need to enter a descriptive name, IP address, username, and password to connect to the device.

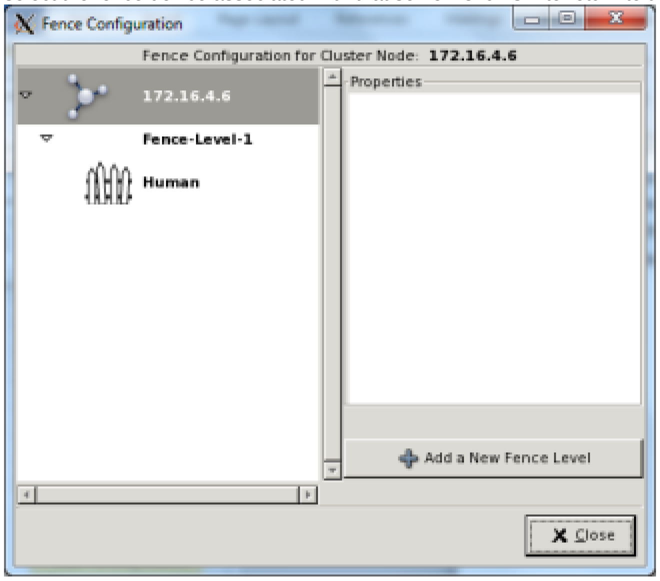
Add one fencing device into the cluster manager for each iLO/DRAC you have (for two servers, you'd have two fencing devices).
Fencing, however, is beyond the scope of this document. For demonstration purposes we will use manual fencing.

Add Fencing Device to Cluster Nodes

Each cluster node needs to have its own fencing device assigned to itself so that the cluster manager knows how to power down the device. To add a fencing device to a cluster node, click on **Cluster Nodes**, click on the cluster node you wish to modify, then click **Manage Fencing For This Node**. You will now see the following:



Click on **Add a New Fence Level** then click on the fence level that was just created, **Fence-Level-1** then click **Add a New Fence to this Level**, then select the fence device associated with that server. Click **OK** to return to the cluster node fence configuration:



Click **Close** to return to the cluster configuration system. Repeat the cluster node fence configuration for the other node in this cluster.

Add Resources

All items utilized by the cluster nodes are known as resources. In our case, our resources are:

- **Network device startup script**
- **IP Address**
- **PostgreSQL script**
- **Phonelogd script**
- **sipXecs script**

When using a script in a clustering environment in must be an init style script, accepting **stop, start, restart,** and **status** as arguments.

To add an IP address click **Create a Resource** then in the resource type selection drop down box select **IP Address.** Enter the IP address that sipXecs will be using along with the subnet mask, then click **OK.**

To add a PostgreSQL script click **Create a Resource** then in the resource type selection drop down box select **Script**. Enter the following information:

- **Name:** PGSQL-8
- **File (with path):** /etc/init.d/postgresql

Now you'll need to add a script for network device startup, phonelogd and sipxecs. To add a script click **Create a Resource** then in the resource type selection drop down box select **Script.** For sipxecs, enter the following information:

- **Name:** sipXecs
- **File (with path):** /etc/init.d/sipxecs

For phonelogd, enter the following information

- **Name:** Phonelogd
- **File (with path):** /etc/init.d/phonelogd

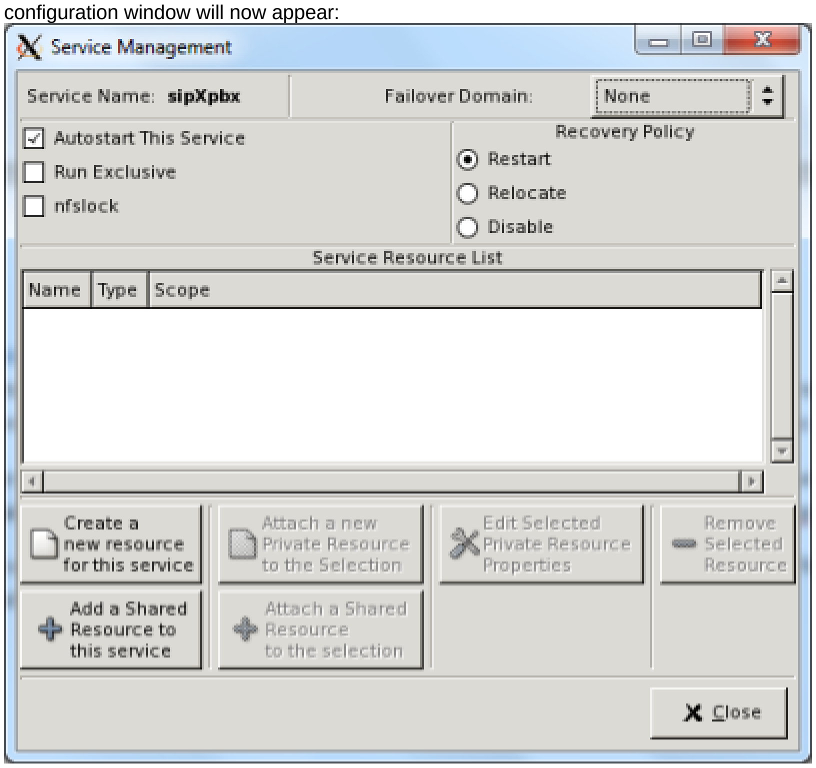
For the network device startup script enter the following information

- **Name:** clusnet
- **File (with path):** /usr/local/bin/clusnet

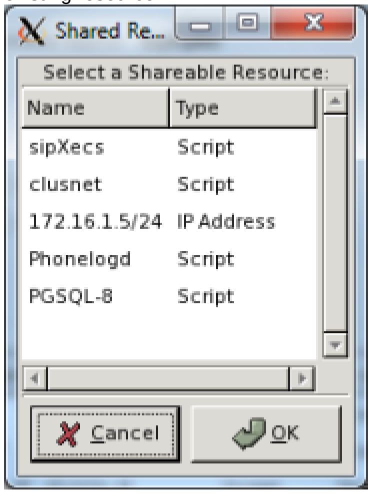
Add Cluster Service

Once you've created all the necessary resources you must now assign them to a cluster service.

To add a service, click on **Services** then click **Add a Service**. Enter a descriptive name for the service, such as sipXpbx, then click **OK**. The service configuration window will now appear:


Service Management
Service Name: sipXpbx
Failover Domain: None
[✓] Autostart This Service
[ ] Run Exclusive
[ ] nfslock
Recovery Policy
(●) Restart
( ) Relocate
( ) Disable
Service Resource List
Name
Type
Scope
Create a new resource for this service
Attach a new Private Resource to the Selection
Edit Selected Private Resource Properties
Remove Selected Resource
Add a Shared Resource to this service
Attach a Shared Resource to the selection
Close

You'll notice there are two options regarding resources: **Create a new resource for this service** and **Add a Shared Resource to this service.** Since we already created the resources we do not need to create them again, so click **Add a Shared Resource to this service** which will prompt you to select an existing resource:

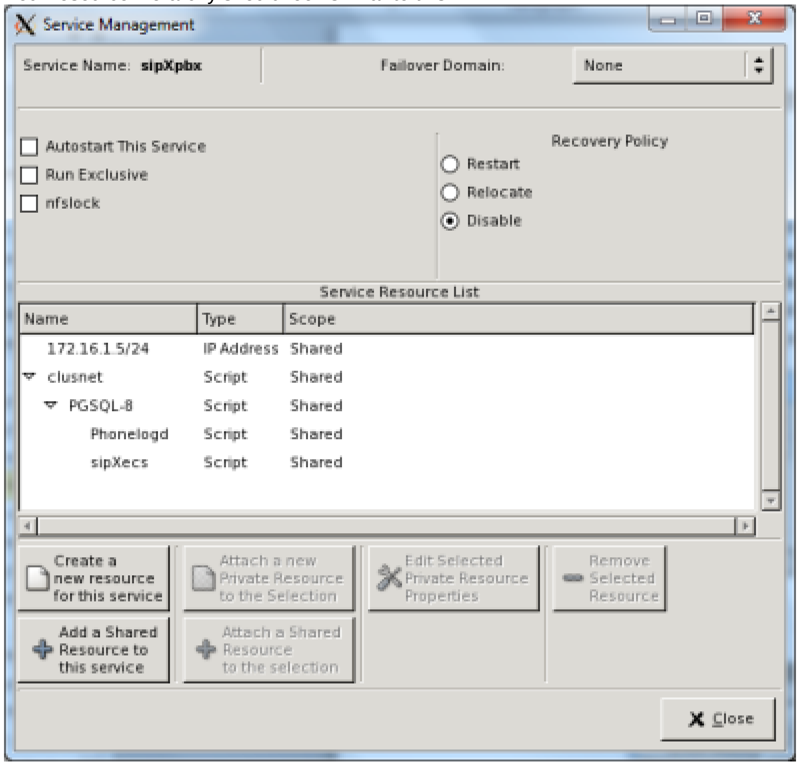


Name	Type
sipXecs	Script
clusnet	Script
172.16.1.5/24	IP Address
Phonelogd	Script
PGSQL-8	Script

You first need to select the IP Address that you created earlier and click **OK**. This will be the top resource in this service, meaning this resource will start before all other resources start.

Once you've added the the IP address you will need to add the network device startup script (clusnet) by clicking **Add Shared resource to this service.** Select the **clusnet** resource created earlier then click **OK**. Select the **clusnet** resource you just added then click **Attach a Shared Resource to the selection,** then select the **PGSQL-8** resource you created earlier and click **OK.** Now add Phonelogd and sipXecs the same way as subordinates of the PostgreSQL resource.

Your resource hierarchy should look similar to this:


Service Management
Create a new resource for this service
Attach a new Private Resource to the Selection
Edit Selected Private Resource Properties
Remove Selected Resource
Add a Shared Resource to this service
Attach a Shared Resource to the selection
Close

Service Name: sipXpbx
Failover Domain: None

Autostart This Service [ ]
Run Exclusive [ ]
nfslock [ ]

Recovery Policy: Restart Relocate [✓ Disable]

Service Resource List

Name	Type	Scope
172.16.1.5/24	IP Address	Shared
clusnet	Script	Shared
PGSQL-8	Script	Shared
Phonelogd	Script	Shared
sipXecs	Script	Shared

For now we don't want this service to automatically start when the cluster service starts so we need to disable automatic startup for this service. To do this uncheck the **Autostart This Service** checkbox, then click **Close**

Save cluster configuration

To save the cluster configuration, click on **File** then click on **Save** and save the configuration to **/etc/cluster/cluster.conf**

Propagate the Cluster Configuration

For the initial propagation of the cluster configuration to the second cluster node you will need to manually copy **/etc/cluster/cluster.conf** from the primary node to the secondary node. This can be done with SCP or by copying and pasting the contents of the file.

Start Clustering Service

To start the clustering service and all of the necessary dependencies, run the following commands on both nodes:

```
service cman start
service rgmanager start
service clvmd start
service gfs start
vgscan
chkconfig cman on
chkconfig clvmd on
chkconfig rgmanager on
chkconfig gfs on
```

To verify that the cluster service is up and running, run the **clustat** command. You should see output similar to the following:

```
Cluster Status for uc_cluster @ Sun Jul  3 14:46:41 2011
Member Status: Quorate

 Member Name                              ID   Status
 ------ ----                              ---- ------
 172.16.4.5                                   1 Online, Local, rgmanager
 172.16.4.8                                   2 Online, rgmanager

 Service Name                    Owner (Last)                   State
 ------- ----                    ----- ------                   -----
 service:sipXpbx                 172.16.4.5                     disabled
```

Create Clustered Storage

As of this writing the best filesystem to use for cluster storage is GFS. This is because unlike most other filesystems, GFS is safe to mount on both nodes at the same time without risking data corruption. This allows for quick failover with less risk of data corruption.

Create LVM

GFS requires LVM to operate properly, so we will need to create a LVM physical volume (PV), volume group (VG), and logical volumes (LV). **This only needs to be done on one node of the cluster, preferably the primary node.**

Enter the following commands to create the PV and the VG (assuming **/dev/sda** for the shared disk)

```
pvcreate /dev/sda vgcreate sipx-vol /dev/sda
```

Now let's create some logical volumes. There will need to be 5 total logical volumes as there are 5 distinct area that need to be shared for proper sipXecs operation:

- **/etc/sipxpbx**
- **/var/sipxdata**
- **/opt/openfire**
- **/opt/freeswitch**
- **/var/lib/pgsql**

We'll start off with a logical volume called **etc_sipxpbx**:

```
lvcreate -L5G -netc_sipxpbx sipx-vol
```

This creates a 5 GB logical volume with the name **etc_sipxpbx** in the volume group **sipx-vol**.

Now we need to create the rest of these volumes. Bear in mind that the PostgreSQL database (/var/lib/pgsql) and the sipXecs data store (/var/sipxdata) will need the most space.

Run the following commands, changing the space values per your needs:

```
lvcreate -L5G -nopenfire sipx-vol
lvcreate -L5G -nfreeswitch sipx-vol
lvcreate -L30G -nsipxdata sipx-vol
lvcreate -L30G -npgsql sipx-vol
```

Create GFS file systems

Run the following command to create a GFS partition on the **etc_sipxpbx** volume (this only needs to be done on one node of the cluster, preferably the primary node):

```
gfs_mkfs -p lock_dlm -t uc_cluster:etc_sipxpbx -j 2 /dev/sipx-vol/etc_sipxpbx
```

Here is an explanation of the arguments:

- -p lock_dlm
    - This defines the locking method.
- -t uc_cluster:etc_sipxpbx

        - This defines the lock table to be used. The first part is the cluster that will be utilizing it (**uc_cluster**) and the second part can be anything but it is easiest to use the LV name.
    - -j 2
        - This defines the number of journals to use. The number of journals is directly coordinated to the number of nodes in your cluster.
    - /dev/sipx-vol/etc_sipxpbx
        - This is the LVM device we wish to create the filesystem on

Create the filesystem on each LV:

```
gfs_mkfs -p lock_dlm -t uc_cluster:freeswitch -j 2 /dev/sipx-vol/freeswitch
gfs_mkfs -p lock_dlm -t uc_cluster:openfire -j 2 /dev/sipx-vol/openfire
gfs_mkfs -p lock_dlm -t uc_cluster:pgsql -j 2 /dev/sipx-vol/pgsql
gfs_mkfs -p lock_dlm -t uc_cluster:sipxdata -j 2 /dev/sipx-vol/sipxdata
```

Mount Filesystems in Appropriate Locations

First we need to create the directories where these filesystems will be located. Run the following commands on both nodes:

```
mkdir /opt/freeswitch
mkdir /opt/openfire
mkdir /var/sipxdata
mkdir /etc/sipxpbx
mkdir /var/lib/pgsql
```

Edit /etc/fstab and add the following lines to the end of the file:

```
/dev/sipx-vol/freeswitch /opt/freeswitch  gfs     defaults 0 0
/dev/sipx-vol/openfire  /opt/openfire  gfs     defaults 0 0
/dev/sipx-vol/sipxdata  /var/sipxdata  gfs     defaults 0 0
/dev/sipx-vol/etc_sipxpbx /etc/sipxpbx  gfs     defaults 0 0
/dev/sipx-vol/pgsql  /var/lib/pgsql  gfs     defaults 0 0
```

Now mount all of the partitions on both servers:

```
mount /etc/sipxpbx
mount /opt/freeswitch
mount /opt/openfire
mount /var/sipxdata
mount /var/lib/pgsql
```

Install sipXecs Packages

Now that we have all the shared storage mounted we can install and set up sipXecs. First we need to install the repository. Edit **/etc/yum.repos.d/sipxecs.repo** on both nodes and add the following information:

```
[sipXecs]
name=sipXecs software for CentOS $releasever - $basearch
baseurl=http://download.sipfoundry.org/pub/sipXecs/4.4.0/CentOS_$releasever/$basearch
gpgcheck=0
```

Now, **on the first node only**, install the necessary packages to support sipXecs:

```
yum install sipxecs caching-nameserver bind ntp
```

After the first installation has completely finished, perform the same installation on the second node.

In the unlikely event that your postgres and sipxchange UID and GID do not match up on the two nodes, you will need to change /etc/passwd and /etc/group on the first node to match the values present on the second node. This is because the installation is performed last on the second node and filesystem permissions are set to the UID of the user on the second node.

Run sipXecs setup

On the primary node, enable the sipXecs NIC buy running the following command:

```
/usr/local/bin/clusnet start
```

Now modify **/etc/hosts** on both nodes to have the correct information. This includes setting the local hostname to the cluster IP address:

```
# Do not remove the following line, or various programs# that require network functionality will fail.
127.0.0.1 localhost.localdomain localhost
172.16.1.5 openuc.sip.corp.ezuce.com openuc
```

Run the following command on the primary node:

```
sipxecs-setup
```

Enter all the necessary information and then when prompted, select **Exit to prompt.**

Disable sipXecs services from starting at bootup:

```
chkconfig sipxecs off chkconfig postgresql off chkconfig phonelogd off
```

Now we need to generate the default postgres configuration. On the primary server start postgresql:

```
service postgresql start
```

And now stop it (the cluster will manage this service):

```
service postgresql stop
```

You'll also need to change the IP address of your primary node's communication interface (eth0) back to the original settings by running:

```
/usr/local/bin/clusnet stop
```

DNS Caching Nameserver Configuration

Because running sipxecs-setup-system would have broken many things we'll have to perform a few steps involving DNS manually.

The first thing we need to do **on both nodes** is remove the file **/etc/named. caching-nameserver.conf** and add our own **/etc/named.conf** with the following contents:

```
// WARNING: Name server configuration is a sipX automatically generated file.
//Contents may be overwritten unless you change the mode to "Manual".
//Available modes:
//"Master"    - Master name server (on primary server).
//"Slave"     - Slave named server (on distributed server).
//"Caching"   - Caching only name server.
//"Manual"    - Blocks future automatic updates.
// DNS_MODE="Caching"
//
options {
        directory "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
forwarders {
                192.168.5.20;
                192.168.5.22;
        };
        forward only;
};
```

Change the **forwarders** section to match your DNS servers.

Restart the caching DNS server and enable it during startup:

```
service named restart chkconfig named on
```

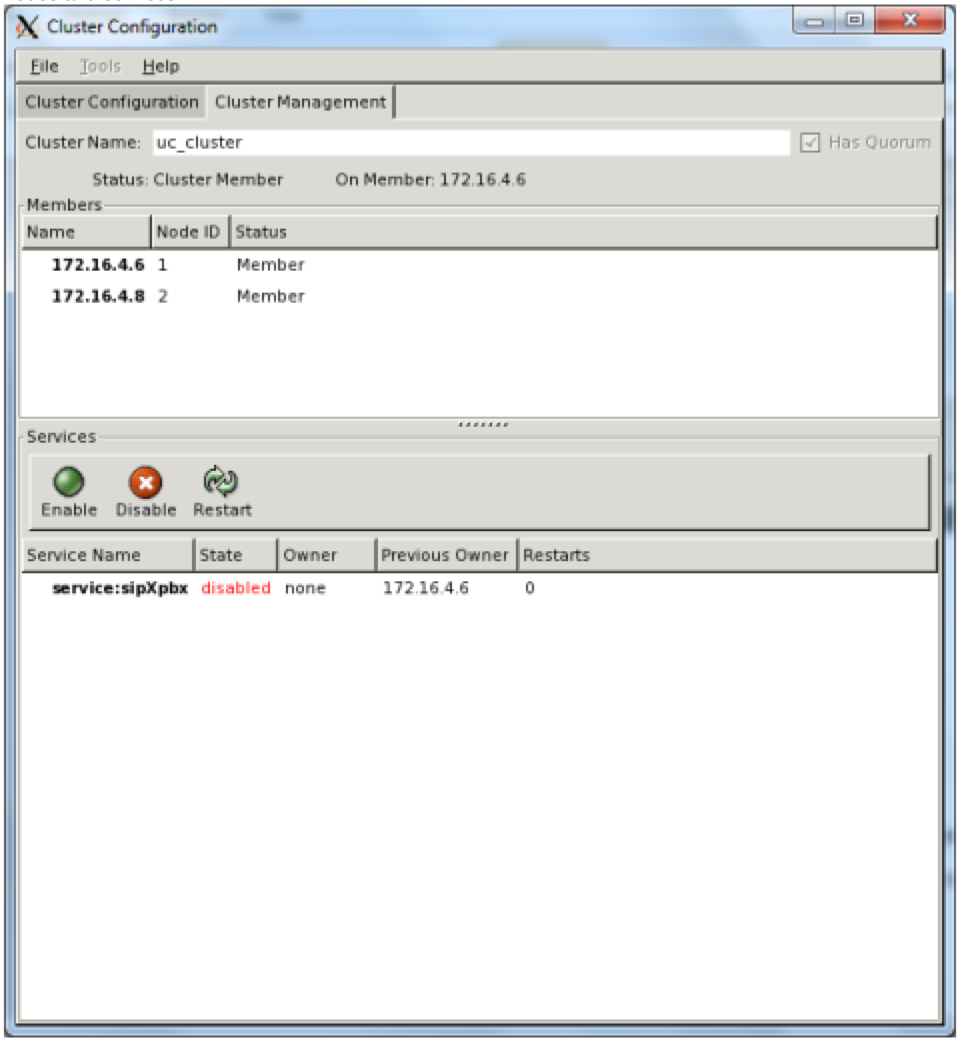
Modify **/etc/resolv.conf** to use the local caching DNS server first, changing the other values as needed:

```
search sip.corp.ezuce.com
nameserver 127.0.0.1
nameserver 192.168.5.20
nameserver 192.168.5.22
```

Reboot both nodes simultaneously for all configuration changes to take effect.

Starting sipXecs Cluster Service

To start sipXecs you'll need to start **system-config-cluster** then click on the **Cluster Management** tab at the top of the window. Here you'll see the active nodes and services:



Cluster Name: uc_cluster

Status: Cluster Member   On Member: 172.16.4.6

Name	Node ID	Status
172.16.4.6	1	Member
172.16.4.8	2	Member

Service Name	State	Owner	Previous Owner	Restarts
service:sipXpbx	disabled	none	172.16.4.6	0

To start the sipXpbx service, click on the service and then click the **Enable** button. This will start all the necessary services.