High Availability

CONTENTS

## Contents

VitalPBX High Availability

1.- Introduction

High availability is a characteristic of a system which aims to ensure an agreed level of operational performance, usually uptime, for a higher than normal period.

Make a high-availability cluster out of any pair of VitalPBX servers. VitalPBX can detect a range of failures on one VitalPBX server and automatically transfer control to the other server, resulting in a telephony environment with minimal down time.

Example
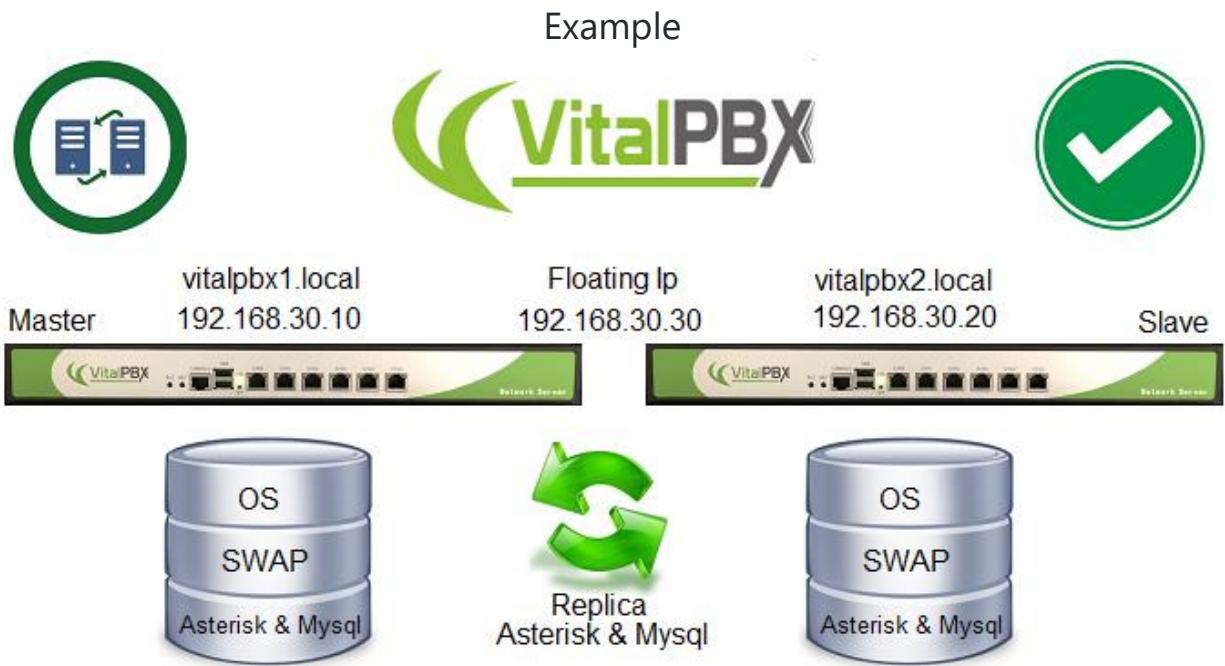
2.- Prerequisites

In order to install VitalPBX in high availability you need the following:

a.- 3 IP addresses.
b.- Install VitalPBX on two servers with similar characteristics.
c.- At the time of installation leave the largest amount of space on the hard drive to store the variable data on both servers.

3.- Installation

We are going to start by installing VitalPBX on two servers

a.- When starting the installation go to:
INSTALLATION DESTINATION (Custom partitioning selected)
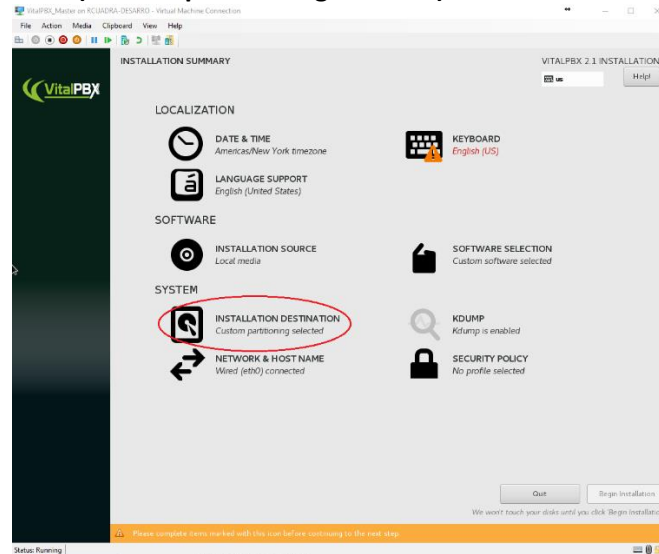
b.- Select:
I will configure partitioning
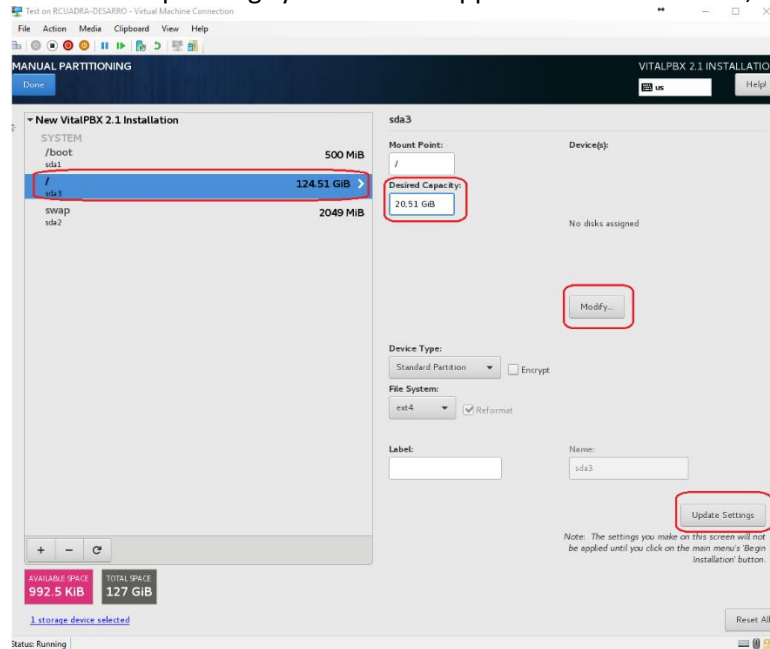And press the button
Done
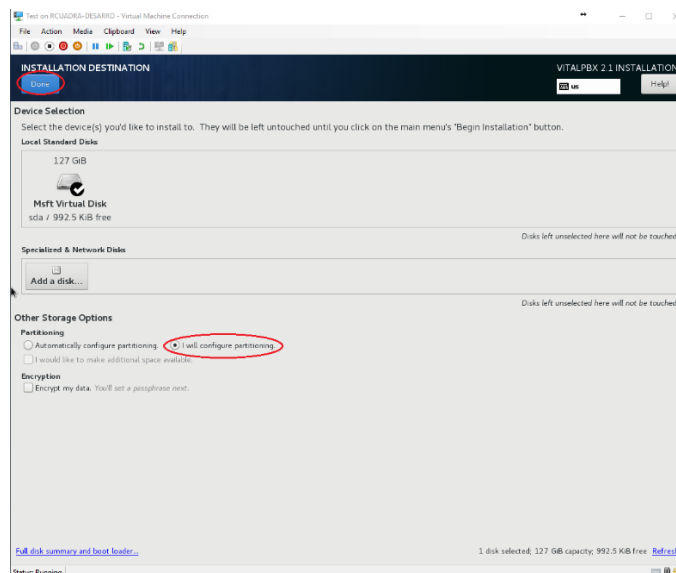
c.- Select the root partition:
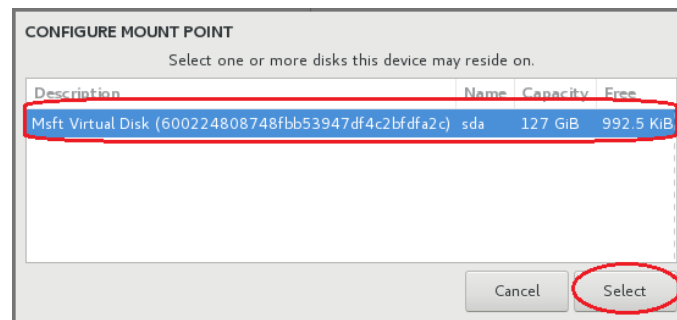**/**
Change the capacity to:
Desired Capacity: 20GB
We need enough space for the operating system and its applications in the future; then click

"Modify…" button
Select the disk and press
Select

Update Settings
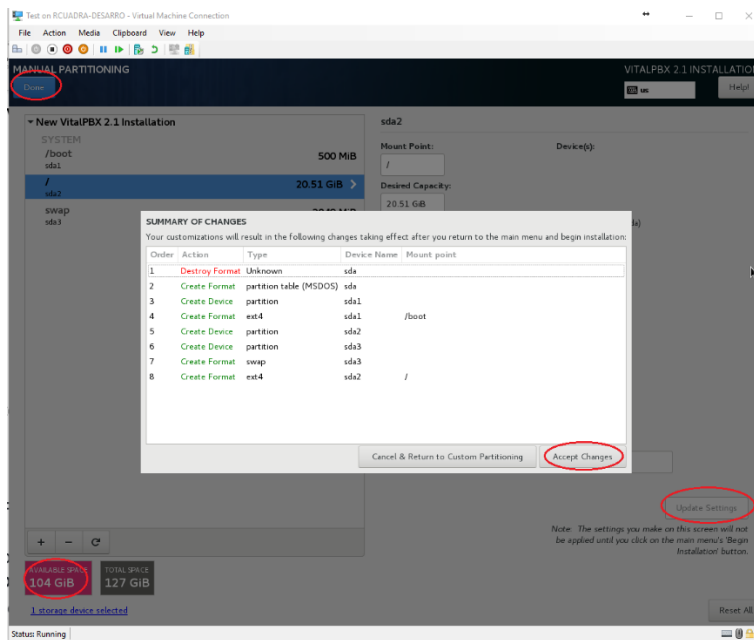
Finally, we press:
Done
And press:
Accept Changes
And continue with the installation.

4.- Configurations

4.1- IP and Hostname Configuration.

We will configure in each server the IP address and the host name.
Go to the web interface to:
Admin>System Settings>Network Settings

First, change the Hostname, and remember to press the **Check button**.
Disable the DHCP option and we will set these values

Name	Master	Slave
Hostname	vitalpbx-master.local	vitalpbx-slave.local
IP Address	192.168.30.10	192.168.30.20
Netmask	255.255.248.0	255.255.248.0
Gateway	192.168.24.1	192.168.24.1
Primary DNS	8.8.8.8	8.8.8.8
Secondary DNS	8.8.4.4	8.8.4.4

Master

Slave

4.2.- Hostname

Now we connect through ssh to each of the servers and we configure the hostname of each server in the /etc/hosts file, so that both servers see each other with the hostname.

```
[root@vitalpbx-master-slave ~]# vi /etc/hosts
192.168.30.10 vitalpbx-master.local
192.168.30.20 vitalpbx-slave.local
```

4.3.- Create the partition on both servers

Initialize the partition to allocate the available space on the hard disk. Do these on both servers.

```
[root@vitalpbx-master-slave ~]# fdisk /dev/sda
Command (m for help): n
Partition type:
   p   primary (3 primary, 0 extended, 1 free)
   e   extended
Select (default e): p
Selected partition 4 (take note of the assigned partition number as we will need it later)
First sector (35155968-266338303, default 35155968): [Enter]
Last sector, +sectors or +size{K,M,G} (35155968-266338303, default 266338303): [Enter]
Using default value 266338303
Partition 4 of type Linux and of size 110.2 GiB is set
Command (m for help): t
Partition number (1-4, default 4): 4
Hex code (type L to list all codes): 8e
Changed type of partition 'Linux' to 'Linux LVM'
Command (m for help): w
```

Then, restart the servers so that the new table is available.

```
[root@vitalpbx-master-slave ~]# reboot
```

4.4.- Format the partition

Now, we will proceed to format the new partition in both servers with the following command:

```
[root@vitalpbx-master-slave ~]# mke2fs -j /dev/sda4
[root@vitalpbx-master-slave ~]# dd if=/dev/zero bs=1M count=500 of=/dev/sda4; sync
```

4.5.- Firewall

Adjust the firewall using the following commands:

```
[root@ vitalpbx-master-slave ~]# firewall-cmd --permanent --add-service=high-
availability
```

Just in Server-Master

```
[root@ vitalpbx-master ~]# firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.30.20" port port="7789" protocol="tcp" accept'
```

Just in Server-Slave

```
[root@ vitalpbx-slave ~]# firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.30.10" port port="7789" protocol="tcp" accept'
```

In both Servers

```
[root@ vitalpbx-master-slave ~]# firewall-cmd --reload
```

4.-6.- Install Dependencies

Install the necessary dependencies on both servers

```
[root@ vitalpbx-master-slave ~]# yum -y install drbd90-utils kmod-drbd90 corosync pacemaker
pcs
```

4.7.- Configuring DRBD

Load the module and enable the service on both nodes, using the follow command:

```
[root@vitalpbx-master-slave ~]# modprobe drbd
[root@vitalpbx-master-slave ~]# systemctl enable drbd.service
```

Create a new global_common.conf file on both nodes with the following contents:

```
[root@ vitalpbx-master-slave ~]# mv /etc/drbd.d/global_common.conf
/etc/drbd.d/global_common.conf.orig
[root@ vitalpbx-master-slave ~]# vi /etc/drbd.d/global_common.conf
global {
  usage-count no;
}
  common {
net {
  protocol C;
  }
}
```

Next, we will need to create a new configuration file called /etc/drbd.d/drbd0.res for the new resource named drbd0, with the following contents:

```
[root@ vitalpbx-master-slave ~]# vi /etc/drbd.d/drbd0.res
resource drbd0 {
protocol C;
    on vitalpbx1.local {
        device /dev/drbd0;
        disk /dev/sda4;
        address 192.168.30.10:7789;
        meta-disk internal;
    }
    on vitalpbx2.local {
        device /dev/drbd0;
        disk /dev/sda4;
        address 192.168.30.20:7789;
        meta-disk internal;
    }
handlers {
    split-brain "/usr/lib/drbd/notify-split-brain.sh root";
    }
net  {
    after-sb-0pri discard-zero-changes;
    after-sb-1pri discard-secondary;
    after-sb-2pri disconnect;
    }
}
```

Initialize the meta data storage on each nodes by executing the following command on both nodes

```
[root@vitalpbx-master-slave ~]#  drbdadm create-md drbd0
Writing meta data...
New drbd meta data block successfully created.
```

Let's define the DRBD Primary node as first node "vitalpbx-master"

```
[root@vitalpbx-master ~]# drbdadm up drbd0
[root@vitalpbx-master ~]# drbdadm primary drbd0 --force
```

On the Secondary node "vitalpbx-slave" run the following command to start the drbd0

```
[root@vitalpbx-slave ~]# drbdadm up drbd0
```

You can check the current status of the synchronization while it's being performed. The *cat /proc/drbd* command displays the creation and synchronization progress of the resource, as shown here:

4.7.1.- Formatting and Test DRBD Disk

In order to test the DRBD functionality we need to Create a file system, mount the volume and write some data on primary node "vitalpbx-master" and finally switch the primary node to "vitalpbx-slave"

Run the following command on the primary node to create an xfs filesystem on /dev/drbd0 and mount it to the mnt directory, using the following commands

```
[root@ vitalpbx-master ~]# mkfs.xfs /dev/drbd0
[root@ vitalpbx-master ~]# mount /dev/drbd0 /mnt
```

Create some data using the following command:

```
[root@ vitalpbx-master ~]# touch /mnt/file{1..5}
[root@ vitalpbx-master ~]# ls -l /mnt/total 0
-rw-r--r-- 1 root root 0 Nov 17 11:28 file1
-rw-r--r-- 1 root root 0 Nov 17 11:28 file2
-rw-r--r-- 1 root root 0 Nov 17 11:28 file3
-rw-r--r-- 1 root root 0 Nov 17 11:28 file4
-rw-r--r-- 1 root root 0 Nov 17 11:28 file5
```

Let's now switch primary mode "vitalpbx-server" to second node "vitalpbx-slave" to check the data replication works or not.
First, we have to unmount the volume drbd0 on the first drbd cluster node "vitalpbx-master" and change the primary node to secondary node on the first drbd cluster node "vitalpbx-master"

```
[root@ vitalpbx-master ~]# umount /mnt
[root@ vitalpbx-master ~]# drbdadm secondary drbd0
```

Change the secondary node to primary node on the second drbd cluster node "vitalpbx-slave"

```
[root@ vitalpbx-slave ~]# drbdadm primary drbd0 --force
```

Mount the volume and check the data available or not.

```
[root@ vitalpbx-slave ~]# mount /dev/drbd0 /mnt
[root@ vitalpbx-slave ~]# ls -l  /mnt
total 0
-rw-r--r-- 1 root root 0 Nov 17 11:28 file1
-rw-r--r-- 1 root root 0 Nov 17 11:28 file2
-rw-r--r-- 1 root root 0 Nov 17 11:28 file3
-rw-r--r-- 1 root root 0 Nov 17 11:28 file4
-rw-r--r-- 1 root root 0 Nov 17 11:28 file5
```

Normalize
Server-Slave

```
[root@ vitalpbx-slave ~]# umount /mnt
[root@ vitalpbx-slave ~]# drbdadm secondary drbd0
```

Server-Master

```
[root@ vitalpbx-master ~]# drbdadm primary drbd0
[root@ vitalpbx-master ~]# mount /dev/drbd0 /mnt
```

4.8.- Configure Cluster

Create the password of the hacluster user on both nodes
```
[root@ vitalpbx-master-slave ~]# echo MyPassword | passwd --stdin hacluster
```

Start PCS on both servers
```
[root@ vitalpbx-master-slave ~]# systemctl start pcsd
```

Configure the start of services on both nodes
```
[root@ vitalpbx-master-slave ~]# systemctl enable pcsd.service
[root@ vitalpbx-master-slave ~]# systemctl enable corosync.service
[root@ vitalpbx-master-slave ~]# systemctl enable pacemaker.service
```

Server Authenticate in Master
```
[root@ vitalpbx-master ~]# pcs cluster auth vitalpbx-master.local vitalpbx-slave.local -u
hacluster -p MyPassword

vitalpbx-master.local: Authorized
vitalpbx-slave.local: Authorized
```

Create the cluster and configure parameters, perform only on the server-master
```
[root@ vitalpbx-master ~]# pcs cluster setup --name cluster_voip vitalpbx-master.local
vitalpbx-slave.local
```

Starting Cluster in Master
```
[root@ vitalpbx-master ~]# pcs cluster start --all
[root@ vitalpbx-master ~]# pcs cluster enable --all
[root@ vitalpbx-master ~]# pcs property set stonith-enabled=false
[root@ vitalpbx-master ~]# pcs property set no-quorum-policy=ignore
```

Create resource for the use of Floating IP
```
[root@ vitalpbx-master ~]# pcs resource create virtual_ip ocf:heartbeat:IPaddr2
ip=192.168.30.30 cidr_netmask=21 op monitor interval=30s on-fail=restart
[root@ vitalpbx-master ~]# pcs cluster cib drbd_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push drbd_cfg
```

Create resource for the use of DRBD
```
[root@ vitalpbx-master ~]# pcs -f drbd_cfg resource create DrbdData ocf:linbit:drbd
drbd_resource=drbd0 op monitor interval=60s
[root@ vitalpbx-master ~]# pcs -f drbd_cfg resource master DrbdDataClone DrbdData master-max=1
master-node-max=1 clone-max=2 clone-node-max=1 notify=true
[root@ vitalpbx-master ~]# pcs cluster cib-push drbd_cfg
```

Create FILESYSTEM resource for the automated mount point
```
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs -f fs_cfg resource create DrbdFS Filesystem device="/dev/drbd0"
directory="/mnt" fstype="xfs"
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add DrbdFS with DrbdDataClone
INFINITY with-rsc-role=Master
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order promote DrbdDataClone then start
DrbdFS
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add DrbdFS with virtual_ip
INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order virtual_ip then DrbdFS
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg
```

Stop and disable all services in both servers
```
[root@ vitalpbx-master-slave ~]# systemctl stop fail2ban
[root@ vitalpbx-master-slave ~]# systemctl disable fail2ban
[root@ vitalpbx-master-slave ~]# systemctl stop asterisk
[root@ vitalpbx-master-slave ~]# systemctl disable asterisk
```

```
[root@ vitalpbx-master-slave ~]# systemctl stop vpbx-monitor
[root@ vitalpbx-master-slave ~]# systemctl disable vpbx-monitor
[root@ vitalpbx-master-slave ~]# systemctl stop mariadb
[root@ vitalpbx-master-slave ~]# systemctl disable mariadb
[root@ vitalpbx-master-slave ~]# usermod -u 1000 asterisk
```

Create resource for the use of MariaDB in Master

```
[root@ vitalpbx-master ~]# mkdir /mnt/mysql
[root@ vitalpbx-master ~]# mkdir /mnt/mysql/data
[root@ vitalpbx-master ~]# cd /mnt/mysql
[root@ vitalpbx-master ~]# cp -aR /var/lib/mysql/* /mnt/mysql/data
[root@ vitalpbx-master-slave ~]# sed -i 's/var\/lib\/mysql/mnt\/mysql\/data/g' /etc/my.cnf
[root@ vitalpbx-master ~]# mv /etc/my.cnf /mnt/mysql/
[root@ vitalpbx-slave ~]# rm -rf /etc/my.cnf
[root@ vitalpbx-master-slave ~]# ln -s /mnt/mysql/my.cnf /etc/
[root@ vitalpbx-master ~]# pcs resource create mysql ocf:heartbeat:mysql
binary="/usr/bin/mysqld_safe" config="/etc/my.cnf" datadir="/mnt/mysql/data"
pid="/var/lib/mysql/mysql.pid" socket="/var/lib/mysql/mysql.sock" additional_parameters="--
bind-address=0.0.0.0" op start timeout=60s op stop timeout=60s op monitor interval=20s
timeout=30s on-fail=standby
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add mysql with virtual_ip
INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order DrbdFS then mysql
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
```

Create resource for DAHDI

```
[root@ vitalpbx-master ~]# pcs resource create dahdi service:dahdi op monitor interval=30s
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add dahdi with virtual_ip
INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order mysql then dahdi
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
```

Path Asterisk service

```
[root@ vitalpbx-master-slave ~]# sed -i 's/RestartSec=10/RestartSec=1/g'
/usr/lib/systemd/system/asterisk.service
[root@ vitalpbx-master-slave ~]# sed -i 's/Wants=mariadb.service/#Wants=mariadb.service/g'
/usr/lib/systemd/system/asterisk.service
[root@ vitalpbx-master-slave ~]# sed -i 's/After=mariadb.service/#After=mariadb.service/g'
/usr/lib/systemd/system/asterisk.service
```

Create resource for Asterisk

```
[root@ vitalpbx-master ~]# pcs resource create asterisk service:asterisk op monitor
interval=30s
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add asterisk with virtual_ip
INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order mysql then asterisk
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg --config
```

Copy folders and files the DRBD partition on the server1

```
[root@ vitalpbx-master ~]# tar -zcvf var-asterisk.tgz /var/log/asterisk
[root@ vitalpbx-master ~]# tar -zcvf var-lib-asterisk.tgz /var/lib/asterisk
[root@ vitalpbx-master ~]# tar -zcvf var-lib-ombutel.tgz /var/lib/ombutel
[root@ vitalpbx-master ~]# tar -zcvf var-share-ombutel.tgz /var/share/ombutel
[root@ vitalpbx-master ~]# tar -zcvf usr-lib64-asterisk.tgz /usr/lib64/asterisk
[root@ vitalpbx-master ~]# tar -zcvf var-spool-asterisk.tgz /var/spool/asterisk
[root@ vitalpbx-master ~]# tar -zcvf etc-asterisk.tgz /etc/asterisk

[root@ vitalpbx-master ~]# tar xvfz /mnt/var-asterisk.tgz -C /mnt/
[root@ vitalpbx-master ~]# tar xvfz /mnt/var-lib-asterisk.tgz -C /mnt/
```

```
[root@ vitalpbx-master ~]# tar xvfz /mnt/var-lib-ombutel.tgz -C /mnt/
[root@ vitalpbx-master ~]# tar xvfz /mnt/var-share-ombutel.tgz -C /mnt/
[root@ vitalpbx-master ~]# tar xvfz /mnt/usr-lib64-asterisk.tgz -C /mnt/
[root@ vitalpbx-master ~]# tar xvfz /mnt/var-spool-asterisk.tgz -C /mnt/
[root@ vitalpbx-master ~]# tar xvfz /mnt/etc-asterisk.tgz -C /mnt/


[root@ vitalpbx-master ~]# rm -rf /var/log/asterisk
[root@ vitalpbx-master ~]# rm -rf /var/lib/asterisk
[root@ vitalpbx-master ~]# rm -rf /var/lib/ombutel
[root@ vitalpbx-master ~]# rm -rf /var/share/ombutel
[root@ vitalpbx-master ~]# rm -rf /usr/lib64/asterisk/
[root@ vitalpbx-master ~]# rm -rf /var/spool/asterisk/
[root@ vitalpbx-master ~]# rm -rf /etc/asterisk

[root@ vitalpbx-master ~]# ln -s /mnt/var/log/asterisk /var/log/asterisk
[root@ vitalpbx-master ~]# ln -s /mnt/var/lib/asterisk /var/lib/asterisk
[root@ vitalpbx-master ~]# ln -s /mnt/var/lib/ombutel /var/lib/ombutel
[root@ vitalpbx-master ~]# ln -s /mnt/var/share/ombutel /var/share/ombutel
[root@ vitalpbx-master ~]# ln -s /mnt/usr/lib64/asterisk /usr/lib64/asterisk
[root@ vitalpbx-master ~]# ln -s /mnt/var/spool/asterisk /var/spool/asterisk
[root@ vitalpbx-master ~]# ln -s /mnt/etc/asterisk /etc/asterisk
```

Configure symbolic links on the server-slave

```
[root@ vitalpbx-slave ~]# rm -rf /var/log/asterisk
[root@ vitalpbx-slave ~]# rm -rf /var/lib/asterisk
[root@ vitalpbx-slave ~]# rm -rf /var/lib/ombutel
[root@ vitalpbx-slave ~]# rm -rf /var/share/ombutel
[root@ vitalpbx-slave ~]# rm -rf /usr/lib64/asterisk/
[root@ vitalpbx-slave ~]# rm -rf /var/spool/asterisk/
[root@ vitalpbx-slave ~]# rm -rf /etc/asterisk

[root@ vitalpbx-slave ~]# ln -s /mnt/var/log/asterisk /var/log/asterisk
[root@ vitalpbx-slave ~]# ln -s /mnt/var/lib/asterisk /var/lib/asterisk
[root@ vitalpbx-slave ~]# ln -s /mnt/var/lib/ombutel /var/lib/ombutel
[root@ vitalpbx-slave ~]# ln -s /mnt/var/share/ombutel /var/share/ombutel
[root@ vitalpbx-slave ~]# ln -s /mnt/usr/lib64/asterisk /usr/lib64/asterisk
[root@ vitalpbx-slave ~]# ln -s /mnt/var/spool/asterisk /var/spool/asterisk
[root@ vitalpbx-slave ~]# ln -s /mnt/etc/asterisk /etc/asterisk
```

Create VitalPBX Service

```
[root@ vitalpbx-master ~]# pcs resource create vpbx-monitor service:vpbx-monitor op monitor
interval=30s
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add vpbx-monitor with
virtual_ip INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order asterisk then vpbx-monitor
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg
```

Create fail2ban Service

```
[root@ vitalpbx-master ~]# pcs resource create fail2ban service:fail2ban op monitor
interval=30s
[root@ vitalpbx-master ~]# pcs cluster cib fs_cfg
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint colocation add fail2ban with virtual_ip
INFINITY
[root@ vitalpbx-master ~]# pcs -f fs_cfg constraint order asterisk then fail2ban
[root@ vitalpbx-master ~]# pcs cluster cib-push fs_cfg
```

Initialize the services of corosync and pacemaker in server-slave

```
[root@ vitalpbx-slave ~]# systemctl restart corosync.service
[root@ vitalpbx-slave ~]# systemctl restart pacemaker.service
```

Show the Cluster status

```
[root@ vitalpbx-master ~]# pcs status resources
virtual_ip      (ocf::heartbeat:IPaddr2):        Started vitalpbx-master.local
 Master/Slave Set: DrbdDataClone [DrbdData]
     Masters: [ vitalpbx-master.local ]
     Slaves: [ vitalpbx-slave.local ]
 DrbdFS (ocf::heartbeat:Filesystem):    Started vitalpbx-master.local
 mysql  (ocf::heartbeat:mysql): Started vitalpbx-master.local
 asterisk       (service:asterisk):     Started vitalpbx-master.local
 vpbx-monitor   (service:vpbx-monitor): Started vitalpbx-master.local
 fail2ban       (service:fail2ban):     Started vitalpbx-master.local

Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

4.9.- Create "bascul" command in both servers

```
[root@ vitalpbx-master-slave ~]# vi bascul
#!/bin/bash
set -e
# Authors:      Rodrigo Cuadra
#               with Collaboration of Jose Miguel Rivera
#               06-Nov-2019
# Support:      rcuadra@aplitel.com
#

#funtion for draw a progress bar
#You must pass as argument the amount of secconds that the progress bar will run
# progress-bar 10 --> it will generate a progress bar that will run per 10 seconds
progress-bar() {
        local duration=\${1}

    already_done() { for ((done=0; done<\$elapsed; done++)); do printf "█"; done }
    remaining() { for ((remain=\$elapsed; remain<\$duration; remain++)); do printf " "; done }
    percentage() { printf "| %s%%" \$(( ((\$elapsed)*100)/(\$duration)*100/100 )); }
    clean_line() { printf "\r"; }

        for (( elapsed=1; elapsed<=\$duration; elapsed++ )); do
                already_done; remaining; percentage
                sleep 1
                clean_line
        done
        clean_line
}

#Define some global variables
host_master=\`pcs status resources | awk '/Masters/ {print \$3}'\`
host_slave=\`pcs status resources | awk '/Slaves/ {print \$3}'\`
drbd_disk_status=\`drbdadm status | awk '/peer-disk/ {print \$1}'\`

#Perform some validations
if [ "\${host_master}" = "" ] || [ "\${host_slave}" = "" ]
then
    echo -e "\e[41m There are problems with high availability, please check with the command
*pcs status* (we recommend applying the command *pcs cluster unstandby* in both servers)
\e[0m"
    exit;
fi

if [ "\$drbd_disk_status" != "peer-disk:UpToDate" ]
then
        echo -e "\e[41m There are problems with high availability, please check with the
command *drbdadm status* (we recommend applying the command *drbdadm up* in both servers)
\e[0m"
        exit;
fi
```

14

```
# Print a warning message and ask to the user if he wants to continue
echo -e "************************************************************"
echo -e "*     Change the roles of servers in high availability     *"
echo -e "*\e[41m WARNING-WARNING-WARNING-WARNING-WARNING-WARNING-WARNING  \e[0m*"
echo -e "*All calls in progress will be lost and the system will be *"
echo -e "*     be in an unavailable state for a few seconds.        *"
echo -e "************************************************************"

#Perform a loop until the users confirm if wants to proceed or not
while [[ \$perform_bascul != yes && \$perform_bascul != no ]]; do
        read -p "Are you sure to switch from $host_master to $host_slave? (yes,no) > "
perform_bascul
done

if [[ "\${perform_bascul}" = "yes" ]]; then
        #Unstandby both nodes
        pcs cluster unstandby $host_master
        pcs cluster unstandby $host_slave

        #Do a loop per resource
        pcs status resources | grep "^\s.*\s(.*):\s.*" | awk '{print \$1}' | while read -r
resource ; do
                #Skip moving the virutal_ip resource, it will be moved at the end
                if [[ "\${resource}" != "virtual_ip" ]]; then
                        echo "Moving \${resource} from \${host_master} to \${host_slave}"
                        pcs resource move \${resource} \${host_slave}
                fi
        done

        sleep 5 && pcs cluster standby \$host_master & #Standby current Master node after five
seconds
        sleep 20 && pcs cluster unstandby \$host_master & #Automatically Unstandby current
Master node after 20 seconds

        #Move the Virtual IP resource to slave node
        echo "Moving virutal_ip from \${host_master} to \${host_slave}"
        pcs resource move virtual_ip \${host_slave}

        #End the script
        echo "Becoming \${host_slave} to Master"
        progress-bar 30
        echo "Done"
        pcs status resources
        drbdadm status
else
        echo "Nothing to do, bye, bye"
fi
```

Add permissions and move to folder /usr/local/bin

```
[root@ vitalpbx-master-slave ~]# chmod +x bascul
[root@ vitalpbx-master-slave ~]# mv bascul /usr/local/bin
```

Test bascul

```
[root@ vitalpbx-master-slave ~]# bascul
************************************************************
*     Change the roles of servers in high availability     *
* WARNING-WARNING-WARNING-WARNING-WARNING-WARNING-WARNING  *
*All calls in progress will be lost and the system will be *
*     be in an unavailable state for a few seconds.        *
************************************************************
Are you sure to switch from vitalpbx-master.local to vitalpbx-slave.local? (yes,no) > yes
Moving DrbdFS from vitalpbx-master.local to vitalpbx-slave.local
Moving mysql from vitalpbx-master.local to vitalpbx-slave.local
Moving dahdi from vitalpbx-master.local to vitalpbx-slave.local
Moving asterisk from vitalpbx-master.local to vitalpbx-slave.local
Moving vpbx-monitor from vitalpbx-master.local to vitalpbx-slave.local
Moving fail2ban from vitalpbx-master.local to vitalpbx-slave.local
Moving virutal_ip from vitalpbx-master.local to vitalpbx-slave.local
```

```
Becoming vitalpbx-slave.local to Master
Done████████████████████| 100%
 virtual_ip    (ocf::heartbeat:IPaddr2):       Started vitalpbx-slave.local
 Master/Slave Set: DrbdDataClone [DrbdData]
     Masters: [vitalpbx-slave.local ]
     Slaves: [ vitalpbx-master.local ]
 DrbdFS (ocf::heartbeat:Filesystem):    Started vitalpbx-slave.local
 mysql  (ocf::heartbeat:mysql): Started vitalpbx-slave.local
 dahdi  (service:dahdi):        Started vitalpbx-slave.local
 asterisk       (service:asterisk):     Started vitalpbx-slave.local
 vpbx-monitor   (service:vpbx-monitor): Started vitalpbx-slave.local
 fail2ban       (service:fail2ban):     Started vitalpbx-slave.local
drbd0 role:Primary
  disk:UpToDate
  vitalpbx-master.local role:Secondary
    peer-disk:UpToDate
```

4.10.- Create "role" command in both servers

```
[root@ vitalpbx-master-slave ~]# vi role
#!/bin/bash
set -e
# Authors:      Rodrigo Cuadra
#               with Collaboration of Jose Miguel Rivera
#               2019/11/06
# Support:      rcuadra@aplitel.com
#
host=\`hostname\`
server_mode=\`pcs status | grep ":\s\[\s\$host\s\]" | awk -F ":" '{print \$1}' | sed -e
's/^[[:space:]]*//'\`
echo ""
echo "Host Local: " \$host
echo "Role:       " \$server_mode
echo ""
echo "Disk Status"
drbdadm status
```

Add permissions and move to folder /usr/local/bin

```
[root@ vitalpbx-master-slave ~]# chmod +x role
[root@ vitalpbx-master-slave ~]# mv role /usr/local/bin
```

Test role

```
[root@ vitalpbx-master-slave ~]# role

Host Local:  vitalpbx-slave.local
Role:        Masters

Disk Status
drbd0 role:Primary
  disk:UpToDate
  vitalpbx-master.local role:Secondary
    peer-disk:UpToDate
```

4.11.- Create "drbdsplit" command in both servers

```
[root@ vitalpbx-master-slave ~]# vi drbdsplit
#!/bin/bash
set -e
# Authors:      Rodrigo Cuadra
#               with Collaboration of Jose Miguel Rivera
#               2019/11/06
# Support:      rcuadra@aplitel.com
#
```

```
drbdadm secondary drbd0
drbdadm disconnect drbd0
drbdadm -- --discard-my-data connect drbd0
ssh root@$ip_slave "drbdadm connect drbd0"
echo "Disk Status"
drbdadm status
EOF
chmod +x /usr/local/bin/drbdsplit
cat > /tmp/drbdsplit << EOF
#!/bin/bash
set -e
# Authors:      Rodrigo Cuadra
#               with Collaboration of Jose Miguel Rivera
#                               2019/11/06
# Support:      rcuadra@aplitel.com
#
drbdadm secondary drbd0
drbdadm disconnect drbd0
drbdadm -- --discard-my-data connect drbd0
ssh root@$ip_master "drbdadm connect drbd0"
echo "Disk Status"
drbdadm status
```

Add permissions and move to folder /usr/local/bin

```
[root@ vitalpbx-master-slave ~]# chmod +x drbdsplit
[root@ vitalpbx-master-slave ~]# mv drbdsplit /usr/local/bin
```

4.12.- Create "vitalwelcome" message in both servers

```
[root@ vitalpbx-master-slave ~]# vi vitalwelcome
#!/bin/bash
#Bash Colour Codes
green="\033[00;32m"
txtrst="\033[00;0m"

if [ -f /etc/redhat-release ]; then
        linux_ver=\`cat /etc/redhat-release\`
        vitalpbx_ver=\`rpm -qi vitalpbx |awk -F: '/^Version/ {print \$2}'\`
        vitalpbx_release=\`rpm -qi vitalpbx |awk -F: '/^Release/ {print \$2}'\`
elif [ -f /etc/debian_version ]; then
        linux_ver="Debian "\`cat /etc/debian_version\`
        vitalpbx_ver=\`dpkg -l vitalpbx |awk '/ombutel/ {print \$3}'\`
else
        linux_ver=""
        vitalpbx_ver=""
        vitalpbx_release=""
fi
vpbx_version="\${vitalpbx_ver}-\${vitalpbx_release}"
asterisk_version=\`rpm -q --qf "%{VERSION}" asterisk\`
host=\`hostname\`
server_mode=\`pcs status | grep ":\s\[\s\$host\s\]" | awk -F ":" '{print \$1}' | sed -e
's/^[[:space:]]*//'\`
logo='

 |¯|  |¯(_)_          |¯(_____ (_____ \¯\  /¯/
 | |  | |_| |_   ____| |_____) )___)  ) \/ /
  \ \/ /| |  _)/ _   | |  ____/  __   ( )  (
   \  / | | |_( ( |  | | |       | |_)  ) /\ \\
    \/  |_|\___)_||_|_|_|      |_____/_/   \_\\
'
echo -e "
\${green}
\${logo}
\${txtrst}

 Role           : \$server_mode
 Version        : \${vpbx_version//[[:space:]]}
```

```
 Asterisk       : \${asterisk_version}
 Linux Version  : \${linux_ver}
 Welcome to     : \`hostname\`
 Uptime         : \`uptime | grep -ohe 'up .*' | sed 's/up //g' | awk -F "," '{print \$1}'\`
 Load           : \`uptime | grep -ohe 'load average[s:][: ].*' | awk '{ print "Last Minute: "
\$3" Last 5 Minutes: "\$4" Last 15 Minutes: "\$5 }'\`
 Users          : \`uptime | grep -ohe '[0-9.*] user[s,]'\`
 IP Address     : \${green}\`ip addr | sed -En 's/127.0.0.1//;s/.*inet (addr:)?(([0-
9]*\.){3}[0-9]*).*/\2/p' | xargs\`\${txtrst}
 Clock          :\`timedatectl | sed -n '/Local time/ s/^[ \t]*Local time:\(.*$\)/\1/p'\`
 NTP Sync.      :\`timedatectl |awk -F: '/NTP sync/ {print \$2}'\`
"
```

Add permissions and move to folder /etc/profile.d/

```
[root@ vitalpbx-master-slave ~]# cp -rf vitalwelcome.sh /etc/profile.d/vitalwelcome.sh
[root@ vitalpbx-master-slave ~]# chmod 755 /etc/profile.d/vitalwelcome.sh
[root@ vitalpbx-master-slave ~]# rm -rf vitalwelcome.sh
```

Test vitalwelcome

```
[root@ vitalpbx-master-slave ~]# /etc/profile.d/vitalwelcome.sh
```

5.- Test

To execute the process of changing the role, we recommend using the following command:

```
[root@vitalpbx-master-slave /]# bascul
****************************************************************
*      Change the roles of servers in high availability      *
* WARNING-WARNING-WARNING-WARNING-WARNING-WARNING-WARNING  *
*All calls in progress will be lost and the system will be *
*      be in an unavailable state for a few seconds.        *
****************************************************************
Are you sure to switch from vitalpbx1.local to vitalpbx2.local? (yes,no) >
```

This action convert the vitalpbx1.local to Slave and vitalpbx2.local to Master. If you want to return to default do the same again.

6.- Turn on and turn off

When you must turn off the servers, when you turn it on always start with the Master, wait for the Master to start and then turn on the Slave.

7.- Update

To update VitalPBX to the latest version just follow the following steps:

1.- From your browser, go to ip 192.168.30.30
2.- Update VitalPBX from the interface
3.- Execute the following command in Master console

```
[root@ vitalpbx-master ~]# bascul
```

4.- From your browser, go to ip 192.168.30.30 again
5.- Update VitalPBX from the interface
6.- Execute the following command in Master console

```
[root@ vitalpbx-master ~]# bascul
```

CONGRATULATIONS, you have installed and tested the high availability in VitalPBX

8.- Some useful commands

- **bascul**, is used to change roles between high availability servers. If all is well, a confirmation question should appear if we wish to execute the action.
- **role**, shows the status of the current server. If all is well you should return Masters or Slaves.
- **pcs resource refresh --full**, to poll all resources even if the status is unknown, enter the following command.
- **pcs cluster unstandby host**, in some cases the bascul command does not finish tilting, which causes one of the servers to be in standby (stop), with this command the state is restored to normal.
- **drbdadm status**, shows the integrity status of the disks that are being shared between both servers in high availability. If for some reason the status of Connecting or Standalone returns to

us, wait a while and if the state remains it is because there are synchronization problems between both servers and you should execute the drbdsplit command.

- **drbdsplit**, solves DRBD split brain recovery.

9.- Credits

9.1 Sources of Information

- voip-info.org

- asterisk.org

- DRBD Website (https://www.linbit.com/en/)

- Pacemaker Website (https://clusterlabs.org/pacemaker/)