Post Installation Processes

Post Installation Configurations

Configure Control Center Access

After the installation is complete you need to configure the Control Center access. This is designed to give the customer a customized Control Center username. For more information on Control Center Access, refer to *CPS Operations Guide*.

Configure NTP on Cluster Manager

To configure NTP on Cluster Manager/Installer, perform the following steps:

Step 1 Install NTP package by executing the following commands:

```
/usr/bin/yum install ntp -y
```

```
chkconfig ntpd on
```

Step 2 Configure NTP Servers: Copy `/etc/ntp.conf` file from any deployed CPS VM.

```
scp pcrfclient01:/etc/ntp.conf /etc/ntp.conf
```

Step 3 Synchronize time with lb01 or lb02.

```
date -s "`ssh lb01 date`"
```

Note Manually enter `date -s "`ssh lb01 date`"` command in your system.

Step 4 Start NTPD service by executing the following command:

```
/usr/bin/systemctl start ntpd
```

IPv6 Support - VMware

Enable IPv6 Support

For VMware hypervisor, IPv6 needs to be enabled first.

Step 1 Select the blade from the left panel where you want to enable IPv6 support.

Step 2 Click **Configure** tab from the top menu from the right panel.

Step 3 Under **Networking**, click **Advanced** from the options available.

Step 4 Click **Edit...** in the upper right corner of the **Advanced** panel.

The **Edit Advanced Network Settings** window opens.

Step 5 From **IPv6 support** drop-down list, select **`Enabled`** to enable IPv6 support.

By performing above steps, IPv6 will be enabled on the blade. Rebooting the blade is required for this setting to take effect.

Note All CPS nodes support IPv4 and IPv6 addresses.

Set Up IPv4 and IPv6 Addresses for VMs

Any hosts in the CPS cluster can be configured to have IPv4 or IPv6 addresses. Currently, IPv6 is supported only for policy director (lb) external interfaces.

For more information on how to configure IPv6 addresses for VMs, refer to the section Hosts Configuration.

Converting IPv4 to IPv6 on Policy Director External Interfaces

To convert an existing CPS deployment from IPv4 to IPv6 (external IP addresses on lb* VM), perform the following steps:

Step 1 Log in to Cluster Manager.

Step 2 Backup the relevant files using the following commands:

```
mkdir /var/backup_ipv4
cp -rf /var/qps/config/deploy/csv /var/backup_ipv4
cp -rf /etc/puppet/modules/qps/templates/var/broadhop/init_pacemaker_res.sh /var/backup_ipv4
```

Step 3 Update the CSV files as per your IPv6 requirement.

The following sample configuration files for Hosts.csv, AdditionalHosts.csv, and Vlan.csv that use IPv6 address are shown:

- Hosts.csv:

```
cat /var/qps/config/deploy/csv/Hosts.csv
Hypervisor Name,Guest Name,Role,Alias,Datastore,Networks -->,Internal,Management
10.10.10.1,lb01,lb01,lb01,datastore8,,192.1.168.10,2003:3041::22:20
10.10.10.2,lb02,lb02,lb02,datastore9,,192.1.168.11,2003:3041::22:21
10.10.10.1,pcrfclient01,pcrfclient01,pcrfclient01,datastore8,,192.1.168.12,
10.10.10.2,pcrfclient02,pcrfclient02,pcrfclient02,datastore9,,192.1.168.13,
10.10.10.1,qns01,qps,qns01,datastore8,,192.1.168.14,
10.10.10.2,qns02,qps,qns02,datastore9,,192.1.168.15,
10.10.10.1,qns03,qps,qns03,datastore8,,192.1.168.16,
10.10.10.2,qns04,qps,qns04,datastore9,,192.1.168.17,
10.10.10.1,sessionmgr01,sm,sessionmgr01,datastore8,,192.1.168.18,
10.10.10.2,sessionmgr02,sm,sessionmgr02,datastore9,,192.1.168.19,
```

- AdditionalHosts.csv:

```
cat /var/qps/config/deploy/csv/AdditionalHosts.csv
Host,Alias,IP Address
ntp-primary,ntp,10.14.58.1
ntp-secondary,btp,10.14.58.2
lbvip01,lbvip01,2003:3041::22:22
lbvip02,lbvip02,192.1.168.20
arbitervip,arbitervip,192.1.168.250
sslvip01,sslvip01,10.12.12.18
qns-site-server-2,pcrf,10.12.12.24
snmp-trapdest,nms-destination,10.12.12.5
10.10.207.8,,10.10.207.8
10.10.207.9,,10.10.207.9
```

- Vlans.csv:

```
cat /var/qps/config/deploy/csv/VLANs.csv
VLAN Name,Network Target Name,Netmask,Gateway,VIP Alias
Internal,vlan467,255.255.255.0,NA,lbvip02
Management,vlan467,64,2003:3041::22:1,lbvip01
External,qps-vlan,255.255.255.0,NA,rtp-swag-vm204
```

Step 4 Execute the following commands to update the changes through puppet and redeploy the Policy Director (lb) VMs:

```
/var/qps/install/current/scripts/import/import_deploy.sh
cd /var/qps/install/current/scripts/deployer
deploy.sh lb01
deploy.sh lb02
```

Note Configure the appropriate firewall rules required for IPv6 or disable the same.

Step 5 After modifying the files, run the following commands:

```
/var/qps/install/current/scripts/build_all.sh
```

```
/var/qps/install/current/scripts/upgrade/reinit.sh
```

Security Enhanced Linux

This release provides support for Security Enhanced Linux (SELinux).

Note You must use htpasswd based authentication instead of PAM based authentication for SElinux.

To enable SElinux:

Step 1 Update `/var/qps/config/deploy/csv/Configuration.csv` file with the following information:

```
selinux,true,

selinux_state,enforcing,

selinux_type,targeted,
```

By default, SELinux is disabled. The following configuration shows SELinux disabled:

```
selinux,false,

selinux_state,disabled,

selinux_type,targeted,
```

Step 2 Import the new configuration by executing the following command:

```
/var/qps/install/current/scripts/import/import_deploy.sh
```

Step 3 Verify that the proper paths are available for custom puppet configuration:

```
mkdir -p /etc/puppet/env_config/nodes
```

Step 4 If `/etc/puppet/env_config/nodes/pcrfclient.yaml` does not exist, copy the existing OAM (PCRFCLIENT) node definition into the env_config nodes by executing the following command:

```
cp /etc/puppet/modules/qps/nodes/pcrfclient.yaml /etc/puppet/env_config/nodes
```

Step 5 Create your new custom manifest `/etc/puppet/env_config/modules/custom/manifests/selinux_httpd_config.pp` for SELinux settings by using below content:

```
cat modules/custom/manifests/selinux_httpd_config.pp
# == Class: custom::selinux_httpd_config
class custom::selinux_httpd_config (
) {
if ('vmware' == $virtual and $::selinux == 'true' and $::selinux_state != 'disabled') {
selboolean { "allow_httpd_mod_auth_pam":
persistent => true,
value => 'on',
}
selboolean { "httpd_setrlimit":
persistent => true,
value => 'on',
}
selboolean { "allow_httpd_anon_write":
persistent => true,
value => 'on',
}
selboolean { "httpd_can_network_connect":
persistent => true,
value => 'on',
}
}
}
```

Step 6 Validate the syntax of your newly created puppet script:

```
puppet parser validate /etc/puppet/env_config/modules/custom/manifests/selinux_httpd_config.pp
```

Step 7 Add a reference to your custom Puppet class 'custom::selinux_httpd_config' in `/etc/puppet/env_config/nodes/pcrfclient.yaml` file.

Step 8 Rebuild your Environment Configuration file by executing the following command:

```
/var/qps/install/current/scripts/build/build_env_config.sh
```

Step 9 For new installations which enable SELinux, after the deployment of all VMs, you must restart the VM for the changes to take effect.

Step 10 For an existing deployed VM, after changing selinux_state (such as, disabled to enforcing, enforcing to disabled), you need to re-initialize setup using reinit.sh and restart the VM for the changes to take effect.

Synchronize Time Between Nodes

To synchronize time between VM nodes, perform the following steps:

Step 1 Login to Cluster Manager VM.

Step 2 Execute the following command to synchronize the time between nodes:

```
/var/qps/bin/support/sync_times.sh ha
```

Note If this is a Geographic Redundancy (GR) installation with multiple sites, refer to *CPS Geographic Redundancy Guide*.

To check the current clock skew of the system, execute the following command:

```
diagnostics.sh --clock_skew -v
```

The output numbers are in seconds. Refer to the following sample output:

```
CPS Diagnostics Multi-Node Environment
--------------------------
Checking for clock skew...
Clock skew not detected between qns01 and lb01. Skew: 1...[PASS]
Clock skew not detected between qns02 and lb01. Skew: 0...[PASS]
Clock skew not detected between lb01 and lb01. Skew: 0...[PASS]
Clock skew not detected between lb02 and lb01. Skew: 0...[PASS]
Clock skew not detected between sessionmgr01 and lb01. Skew: 0...[PASS]
Clock skew not detected between sessionmgr02 and lb01. Skew: 0...[PASS]
Clock skew not detected between pcrfclient01 and lb01. Skew: 0...[PASS]
Clock skew not detected between pcrfclient02 and lb01. Skew: 0...[PASS]
```

Update the VM Configuration without Re-deploying VMs

Sometimes, certain configurations in the excel sheet need to be modified and updated to the deployed VMs. To update the configurations in the excel sheet, perform the following steps:

Step 1 Make the changes to the excel.

Step 2 Save them as CSV files.

Step 3 Upload the csv files to the Cluster Manager VM in `/var/qps/config/deploy/csv/`.

Step 4 Execute the following commands after uploading the csv files to Cluster Manager VM:

```
/var/qps/install/current/scripts/import/import_deploy.sh
```

```
/var/qps/install/current/scripts/upgrade/reinit.sh
```

Reserving Memory on the Virtual Machines (VMs)

To avoid performance impact, you must reserve all allocated memory to each CPS virtual machine.

It is recommended to allocate 8 GB memory for the Hypervisor. For example, suppose the total memory allocated on a blade/ESXi host is 48 GB then we should only allocate 40 GB to CPS VMs and keep 8 GB for the Hypervisor.

Note This is required only if your ESXi host is added to VCenter, if not then the deployment will take care of reservation.

Power OFF the virtual machine before configuring the memory settings.

Step 1 Log in to your ESXi host with the vSphere Client.

Step 2 In the vSphere Client, right-click a virtual machine from the inventory and select **Edit Settings...**.

Step 3 In the **Virtual Machine Properties** window, select **Resources** tab and select **Memory**.

Step 4 In the **Resource Allocation** pane, set the memory reservation to allocated memory.

Step 5 Click **OK** to commit the changes.

Step 6 Power ON the Virtual Machine.

Configure Custom Route

In lb01 and lb02, if needed, custom route should be configured to route diameter traffic to the PGWs.

Add a file called route-ethxx in the `./etc/sysconfig/network-scripts`.

For example, 172.20.244.5/32 via 172.16.38.18

Destination subnet via GW of the subnet.

TACACS+

This section covers the following topics:

TACACS+ Configuration Parameters

Basic instructions for enabling TACACS+ AAA in the system can be found in the section Configure System Parameters for Deployment. There are a number of advanced configuration options which allow administrators to tune their deployments for their specific needs. The following table list TACACS+ configuration parameters that can be added in the Configuration sheet:

Table 1: TACACS+ Configuration Parameters

Parameter	Description	Value Range
tacacs_enabled*	A boolean value indicating whether TACACS+ AAA must be enabled or not.	Values: 1, 0, true, false For example: tacacs_enabled,1
tacacs_server*	An ordered comma-separated list of *<ip>[:port]* pairs indicating which servers need to be queried for TACACS+ AAA.	Values: NA For example: tacacs_server“10.0.2.154:49 ,172.18.63.187:49” **Note** If multiple servers are defined, they must be separated by a comma and enclosed in double quotes, as shown in the example above. Port number with the IP address is optional.
tacacs_secret*	The 'secret' key string used for encrypting the TACACS+ protocol communications.	Values: NA For example: tacacs_secret,CPE1704TKS
tacacs_debug	An integer value indicating the debug level to run the software in. Currently, this is effectively boolean.	Value: 0 1 For example: tacacs_debug,1 Default: 0
tacacs_service	A string value indicating which service to be used when authorizing and auditing against the TACACS+ servers.	Value: NA For example: tacacs_servicepcrflinuxlogin Default: pcrflinuxlogin if no value is specified
tacacs_protocol	A string value indicating which protocol to be used when authorizing and auditing against the TACACS+ servers.	Value: NA For example: tacacs_protocol,ssh Default: ssh

Parameter	Description	Value Range
tacacs_timeout	An integer that represents how long the software needs to wait, in seconds, for the TACACS+ server to respond to the queries.	Value: in seconds For example: tacacs_timeout,2 Default: 5 seconds

The * mark indicates that the parameter is mandatory. * mark is not a part of the parameter.

AIO/Arbiter Configuration for TACACS+

Step 1 Create the following `yaml` file on Cluster Manager: `/etc/facter/facts.d/tacacs.yaml`.

tacacs_enabled: true

tacacs_server: ip address

tacacs_secret: password

Step 2 puppet apply.

```
puppet apply -v --modulepath "/etc/puppet/modules:/etc/puppet/env_config/modules" --pluginsync
/etc/puppet/manifests/init.pp --logdest /var/log/puppet.log
```

Note Manually enter **`puppet apply`** command in your system.

TACACS+ Enabler

The `enable_tacacs+` utility can be used to configure the Cluster Manager VM for TACACS+-based authentication. The utility achieves this by first validating if TACACS+ has been configured properly using the Configuration sheet of CPS Deployment Template (Excel spreadsheet). Assuming the required values are provided, the utility will then selectively apply several Puppet manifests to enable TACACS+ authentication on the target VM.

To use the utility:

Step 1 Get the `tacacs_enabler.tar.gz` package from Cisco Technical Representative.

Step 2 Copy the utility package to the target VM.

Step 3 Acquire shell access on the target VM with the ability to execute operations with 'root' privileges.

Step 4 Extract the utility package using the tar utility on the target VM:

```
tar -zxvf tacacs_enabler.tar.gz
```

```
tacacs_enabler/enable_tacacs+
tacacs_enabler/README.md
```

Step 5 (Optional) Copy the utility to the `/var/qps/bin/support` directory.

```
cp tacacs_enabler/enable_tacacs+ /var/qps/bin/support/
```

Note This step places the utility into a directory which should be in the PATH on the target VM. While not required, this simplifies execution of the script for the later steps.

Step 6 (Optional) Execute the script in 'check' mode to validate the configuration values:

```
Detected VM node type: clustermgr
Generating facts based on current deployment configuration
Validating TACACS+ configuration settings:
* Found required setting for 'tacacs_secret'
* Found optional setting for 'tacacs_debug'
* Found optional setting for 'tacacs_timeout'
* Found required setting for 'tacacs_server'
* Found required setting for 'tacacs_enabled'
Configuration appears to be complete. You should be able to enable TACACS+
on this 'clustermgr' node by executing this command without the '--check'
command-line option.
```

Step 7 Execute the script without the '--check' command-line option to apply the configuration:

```
enable_tacacs+ clustermgr --check
```

```
Detected VM node type: clustermgr
Generating facts based on current deployment configuration
Validating TACACS+ configuration settings:
* Found required setting for 'tacacs_secret'
* Found optional setting for 'tacacs_debug'
* Found optional setting for 'tacacs_timeout'
* Found required setting for 'tacacs_server'
* Found required setting for 'tacacs_enabled'
Executing Puppet to apply configuration:
... Puppet output ...
Notice: Finished catalog run in 34.57 seconds
```

Step 8 Validate that TACACS+ authenticated users are now available on the target VM:

`id -a <`*TACACS+ user*`>`

Adding Indexes for Geo Location (ANDSF)

To create indexes for Geo Location Lookups, perform the following steps:

Step 1 Login to the Cluster Manager with the valid username and credentials.

Step 2 Connect to the active sessionmgr VM.

```
ssh sessionmgr02
```

Step 3 Connect to the active ANDSF mongo database.

```
mongo -port 27717
```

Step 4 Once the mongo is connected successfully, check for the ANDSF dB being present in the list of dB's.

```
set01:PRIMARY> show dbs
```

```
admin (empty)
andsf 6.528GB
```

Step 5 Switch to the ANDSF database.

```
set01:PRIMARY> use andsf
```

Step 6 Check for the Geo Location and dmtl_Policy_EXT_GEO_LOC_STATIC table in the list of collections.

```
set01:PRIMARY> show collections
```

Step 7 Create the following 2 indexes on the Geo_Location Table and dmtl_Policy_EXT_GEO_LOC_STATIC using the following commands on the mongo prompt.

```
set01:PRIMARY> db.Geo_Location.createIndex({"coordinates": "2d"})

set01:PRIMARY> db.dmtl_Policy_EXT_GEO_LOC_STATIC.createIndex({"keys.GEO_Location":1})
```

Step 8 The mongo should return a success code on creation of index.

Configure Multiple Redis Instances

Note All the commands mentioned in the following section should be executed on Cluster Manager.

Before you begin

Redis instance must be enabled and running.

Step 1 To configure multiple redis instances, update *redis_server_count* parameter in `Configuration.csv` spreadsheet in `QPS_deployment_config_template.xlsm` deployment template file.

Step 2 After updating the `Configuration.csv`, execute the following command to import the new configuration file into Cluster Manager VM.

```
/var/qps/install/current/scripts/import/import_deploy.sh
```

Step 3 Edit `redisTopology.ini` file in `/etc/broadhop/` directory and add all redis endpoints:

Note By default, three redis instances are enabled.

For example, for three redis instances, the `redisTopology.ini` file will look like:

```
policy.redis.qserver.1=lb01:6379
policy.redis.qserver.2=lb02:6379
policy.redis.qserver.3=lb01:6380
policy.redis.qserver.4=lb02:6380
policy.redis.qserver.5=lb01:6381
policy.redis.qserver.6=lb02:6381
```

Note

- For every added redis instance, you need to add two lines in `redisTopology.ini` file.
- Redis instances are monitored by monit on lb VMs.
- Guidelines on number of redis instances running on each policy director (lb):

  If Diameter TPS < 15K, then the default number of redis instances (3) will be running on each policy director (lb).

  If Diameter TPS < 28K, then the number of redis instances running on each policy director (lb) should be 4.

  If Diameter TPS > 28K, then the number of redis instances running on each policy director (lb) should be 5.

Step 4 After modifying the configuration file, to make the changes permanent, user needs to rebuild `etc.tar.gz`.

```
/var/qps/install/current/scripts/build/build_etc.sh
```

Step 5 Reinitialize the environment:

```
/var/qps/install/current/scripts/upgrade/reinit.sh
```

Step 6 Restart the qns service.

```
/var/qps/bin/control/restartall.sh
```

Configure Redis Instances for Keystore

Currently, keystore is being used internally for features such as RAN NAS Retry, Holding Rx STR, and so on.

- Keystore is a temporary cache used by application to store temporary information. It stores information in the form of key-value pair.
- Keystore internally uses redis cache for storing the key-value pair.

Note By default, keystore uses redis running on lb01:6379 and lb02:6379 if redis instances is not configured for keystore.

```
-Dredis.keystore.connection.string=lb01:lb02:6379:6379
```

Before you begin

Redis instance must be installed and running on VMs.

If you want to add more redis instances for keystore, run the following OSGi command:

```
telnet qns01 9091
```

Note qns01 must be up and running.

```
setKeystoreConnectionString <start lb>:<end lb>:<start port>:<end port>
```

Range of lbs can be defined using <*start lb*>:<*end lb*>.

Range of redis ports can be defined using <*start port*>:<*end port*>.

For example, to use redis instance running on 6379, 6380 on lb01 to lb04, configure the parameter as follows:

```
telnet qns01 9091
osgi> setKeystoreConnectionString lb01:lb04:6379:6380
Keystore string updated successfully
```

Note The parameter `-Dredis.keystore.connection.string` has been deprecated from CPS 12.0.0 release and is only used to maintain backward compatibility.

The current keystore instances which are being used in application can be checked by the running the following command:

```
telnet qns01 9091
```

Note qns01 must be up and running.

```
listKeystoreShards
```

For example:

```
telnet qns01 9091
osgi> listKeystoreShards

Shard Id    Keystore Instances

1           lb01:6379
2           lb01:6380
3           lb02:6379
4           lb02:6380

Keystore Shards Status: BALANCED
```

Modify Configuration Files

Customers might need to change configurations in the `/etc/broadhop` directory on VMs. It is recommended not to change the configurations in the VMs. All changes must be done in the Cluster Manager VM. To modify configuration files, perform the following steps:

Step 1 In Cluster Manager, modify the files in `/etc/broadhop/`.

Step 2 (Optional) For HA system, we need to configure TCP no delay by modifying the set *Denable_tcp_nodelay* flag in `/etc/broadhop/qns.conf` file.

-Denable_tcp_nodelay=true

Step 3 After modifying the configuration file, to make the changes permanent for future use (when any VM is redeployed or restarted... etc.), user needs to rebuild `etc.tar.gz`.

```
/var/qps/install/current/scripts/build/build_etc.sh
```

Step 4 In Cluster Manager, execute the following command to synchronize the changes to the VM nodes.

```
SSHUSER_PREFERROOT=true copytoall.sh /etc/broadhop/qns.conf /etc/broadhop/qns.conf
```

Step 5 Restart the CPS service if necessary on cluster manager.

```
/var/qps/bin/control/restartall.sh
```

Convert the Cluster Manager VM to an All-in-One

If you want to create an “All-in-One” deployment, you should be able to update the Cluster Manager VM to an AIO.

To upgrade the Cluster Manager VM to an AIO, perform the following steps:

Step 1 Validate your system configuration is correct, with respect to this server's hostname:

```
vi /etc/sysconfig/network
```

Check the value of the “HOSTNAME” variable. Typically this value is set to 'lab'.

a) If you have modified this value, either restart your system, or manually reset the hostname at the command line:

```
hostname `grep HOSTNAME /etc/sysconfig/network | cut -d= -f2`
```

Step 2 Check node to be configured as AIO in `/etc/broadhop.profile` file. If it is not configured to be AIO, then explicitly configure this node to be an 'aio':

```
echo NODE_TYPE=aio > /etc/broadhop.profile
```

Step 3 Execute configuration script to apply the appropriate configurations to the system:

```
puppet apply -v --modulepath "/etc/puppet/modules:/etc/puppet/env_config/modules" --pluginsync
/etc/puppet/manifests/init.pp --logdest /var/log/puppet.log
```

Step 4 Execute the following commands to publish configuration and restart CPS.

```
/var/qps/bin/control/restartall.sh
```

```
restartall.sh script process will prompt for either Y/N to restart process. Enter Y to restart the
process.
```

Step 5 If you want to reset the password for Policy Server (QNS) Linux user on AIO, execute the `change_passwd.sh` script.

Note For fresh installation, before executing `change_passwd.sh` script, run source `/etc/profile.d/broadhop.sh` to source the broadhop scripts in the PATH. This is applicable for first time only.

```
change_passwd.sh
```

```
Enter username whose password needs to be changed: qns
Enter new password:
Re-enter new password:
Changing password on pcrfclient01...
Connection to pcrfclient01 closed.
Password for qns changed successfully on pcrfclient01
```

At this point the Cluster Manager node is properly configured to provide All-in-One service.

Scaling Existing Installation

There might be situations when customer would want to expand existing installation e.g. add more Policy Server (QNS) or session manager virtual machines.

To add more VMs, perform the following steps:

Step 1 Refer to the template file that were used to create earlier installation. These files are present under `/var/qps/config/deploy/csv`. For more information, refer to Import the csv Files into the Cluster Manager VM.

Step 2 Assuming that we are not adding any new VLANs in the scaling up of setup, modify the csv files in the following order to include additional changes:

- Update the Hosts.csv to include new hosts on appropriate ESXi hosts, with corresponding guest name, role, alias etc. For more information, refer to Hosts Configuration.
- For scaling up, if we want to add more Policy Server (QNS), say qns03 and qns04, those entries should get reflected appropriately in above Hosts file.

Note No changes are needed in rest of the template files.

Step 3 Validate the configurations using `jvalidate.py` script. For more information, refer to Validate Imported Data.

```
cd /var/qps/install/current/scripts/deployer/support/

python jvalidate.py
```

Step 4 Once Steps 2 and 3 are completed, import the modified csv file by executing the following command:

```
/var/qps/install/current/scripts/import/import_deploy.sh
```

This would convert updated csv into the JSON file, and also create new `/etc/hosts` file on the Cluster Manager with entries for new virtual machines.

Step 5 For each new hosts (VM) that is defined in the Hosts sheet, we need to run `deploy.sh` script. For more information, refer to Manual Deployment.

Note Make sure that we do not execute `deploy_all.sh` script as it would wipe out the existing deployment and recreate new VMs.

Step 6 Manually copy the new `/etc/hosts` file from cluster manager to all (new and existing) virtual machines.

Note Currently, there is no procedure to synchronize `/etc/hosts` to all the hosts.

What to do next

Important If existing four qns are not able to handle CC (Control Center) and API traffic, we need to make changes in `/etc/haproxy/haproxy.conf` file for additional Policy Server (QNS). If we do not add entries for additional Policy Server (QNS), (e.g. qns05 and qns06), then the CC and API traffic would be handled by existing four Policy Server (QNS) VMs i.e., qns01 to qns04. Also, no changes are required to be done in `/etc/broadhop/servers` for new VMs.

For Gx, no entries are required in `haproxy.conf` file.

Adding Member to Existing Replica Set

During above process you can add new session managers for databases and expand the existing replica-set. The procedure for the same is covered in Defining a Replica-set.

Configure Balance Shards

Balance database can be sharded logically to improve the performance of balance database. Internally it will create multiple balance dbs and distribute the data among each shards.

Prerequisites

This feature is available in 7.5.0 and higher releases. By default, there is one shard that gets created for balance.

- Adding or removing shards to the Balance database must be performed during a maintenance window.
- Back up the Balance database before adding or removing shards. Refer to the *CPS Backup and Restore Guide* for instructions.

Shard Configuration

Shard collection can be increased/decreased based on performance needs.

Add Shards to Balance Database

The following example increases the number of shards from 1 to 6.

Step 1 Log into the Control Center and note down the balance information for a few subscribers. This information is used to confirm the balance of these users after the shards have been added.

Step 2 Log into the Cluster Manger VM.

Step 3 Edit `/etc/broadhop/pcrf/qns.conf` to add the following parameter:

```
-Dcom.cisco.balance.dbs=6
```

Step 4 Run `copytoall.sh` to synchronize the configuration changes to all VMs in the CPS cluster.

Step 5 Run `restartall.sh` to restart all Policy Server (QNS) processes.

Step 6 After restart, connect to qns01 OSGi console and execute `rebalanceBalanceShard <newShardCount>` command.

where, <*newShardCount*> is the new shard count equal to the value configured for `com.cisco.balance.dbs` in `/etc/broadhop/pcrf/qns.conf` file.

Example: `rebalanceBalanceShard 6`

Caution This command may take time to complete. Monitor the rebalance shard and wait until the command finishes. Do not restart Policy Server (QNS) while rebalance is in progress.

This creates six logical shards.

Caution To terminate the OSGi session, use the disconnect command. Do not use the exit command, as this command restarts the process.

Step 7 Verify by connecting to the Balance database. A total of six entries for balance_mgmt must be listed, (balance_mgmt – balance_mgmt_5).

Step 8 Log into Control Center again and verify that the subscribers from Step 1 have the same balance.

Remove Shards from Balance Database

The following example decreases the number of shards from 6 to 1.

Step 1 Log into Control Center and note down the balance information for a few subscribers.

Step 2 Go to Policy Server (QNS) OSGi console and run `rebalanceBalanceShard <newShardCount>` command.

where, <*newShardCount*> is the new shard count equal to the value configured for `com.cisco.balance.dbs` in `/etc/broadhop/pcrf/qns.conf` file.

Caution This command may take time to complete. Monitor the rebalance shard and wait until the command finishes. Do not restart Policy Server (QNS) while rebalance is in progress.

This reduces the shards from six to one.

Caution To terminate the OSGi session, use the `disconnect` command. Do not use the `exit` command, as this command restarts the process.

Step 3 Log into the Cluster Manager VM.

Step 4 Edit the `/etc/broadhop/pcrf/qns.conf` file and add or modify the following parameter:

```
-Dcom.cisco.balance.dbs=1
```

Step 5 Run `copytoall.sh` to synchronize the configuration changes to all VMs in the CPS cluster.

Step 6 Run `restartall.sh` to restart all Policy Server (QNS) processes.

Step 7 Verify by connecting to the Balance database and see the count. Only one entry for balance_mgmt should now be listed.

Step 8 Log into Control Center again and verify that the subscribers from Step 1 have the same balance.

Secondary Key Ring Configuration

CPS provides a high availability solution for secondary key to primary key mappings. Rings are group of memcached servers processes running on different sessionmgr VMs (session cache) which stores the mapping of primary and secondary keys. This is used for secondary key lookup to optimize performance for Rx calls. Examples of secondary key lookups include framed IP Rx session ID IMSI MSISDN.

Architecturally the solution is divided into the following components

- Secondary Key Ring — A secondary key ring is a set of nodes that have a complete set of secondary key to primary key mappings. The secondary keys are partitioned using consistent hashing across the nodes within the ring to ensure an even distribution of the keys.
- Ring Set — Each node on a secondary key ring is called a ring set. A ring set can have 1 to many physical servers. Each server has an exact copy of the data stored for that node. Each additional server within a ring set increases the high availability capability of the system.

Using these component pieces the system supports parallel searching of key mappings across the physical servers to find a specific entry. If a physical server is shutdown or becomes unavailable the system automatically rebuilds the rings and remap the secondary keys to the primary keys when the server comes back online.

The system does not support the following scenario:

- Detecting if a ring is need of a rebuild due to issuing a `flush_all` command.

Why it is Required

- Secondary key (Rx) to primary key (Gx) lookups are cached into a set of *n* servers and failure of a server results in a loss of 1/*n* of the primary to secondary key mappings. Because of this failure number of additional queries continues to increase also the keys are not removed on session removal which ages out.
- Rings are used to handle this situation which allows the server endpoints to grow or shrink. Each key is written to multiple memcached servers within a ring.
- Keys are removed on session removal to keep the cache keys from expiring.
- Queries are parallely executed when search is done against multiple rings to allow for a ring and multiple servers within a ring.

Key Ring Commands

The following commands are provided to support this new functionality.

Note Before implementing any of these commands contact the Cisco AS team to discuss the optimal architecture for your CPS deployment.

All commands must be issued from Policy Server (QNS).

Telnet to any Policy Server (QNS) machine on port 9091 to enter the OSGi console.

Creating a New Ring

To create a new secondary key (sk) ring with a given id:

`createSkRing` *ringid*

Note The *ringid* must be numeric and the ring must initially be empty with no ring sets defined.

Example:

```
createSkRing 2
```

Adding a New Endpoint

This command assigns a set of servers to act as node on the cache ring. Each server will have an exact copy of the data. If a node exists in the ring with that id then it is replaced and the ring is automatically rebuilt.

`setSkRingSet` *ringid setid cacheserver1port[cacheserver2portcacherverNport]*

Example:

```
setSkRingSet 1 1 sessionmgr01:11211 sessionmgr02:11211
```

Removing an Endpoint

This command removes a ring set from a ring. This triggers an automatic rebuild of the ring.

`removeSkRingSet` *ringid setid*

Example:

```
removeSkRingSet 1 2
```

Removing a Ring

This command removes a ring.

Note You cannot remove the last ring from the system.

`removeSkRing` *ringid*

Example:

```
removeSkRing 2
```

Triggering a Ring Rebuild

To trigger a rebuild of a secondary key ring with a given id:

`rebuildSkRing` *ringid*

where, *ringid* is a numeric value.

Example:

rebuildSkRing 1

To track the progress of a ring rebuild refer to the following statistic:

```
skcache_ring[ring id]_entry_rebalance
```

Single Cluster Configuration

Log into pcrfcleint01 or 02 to create/update rings from Policy Server (QNS) OSGi console. Assuming, there are three session cache replica sets, by default, Ring-1 Set-1 gets configured automatically and remaining rings need to be configured manually.

```
osgi> setSkRingSet 1 2 sessionmgr03:11211,sessionmgr04:11211

Ring updated

osgi> setSkRingSet 1 3 sessionmgr05:11211,sessionmgr06:11211
```

Multi-Cluster Configuration

Log into pcrfcleint01 or 02 to create/update rings from Policy Server (QNS) OSGi console. Assuming there are three session cache replica sets by default Ring-1 Set-1 get configured automatically and remaining rings need to be configured manually.

- Configure cluster-1 (Ring-1):

```
osgi> setSkRingSet 1 2 sessionmgr03:11211,sessionmgr04:11211
Ring updated
osgi> setSkRingSet 1 3 sessionmgr05:11211,sessionmgr06:11211
```

- Configure cluster-2 (Ring-2):

```
telnet qns01 9091
osgi> createSkRing 2
Successfully added ring with ID: 2
osgi> setSkRingSet 2 1 sessionmgr01:11211,sessionmgr02:11211
osgi> setSkRingSet 2 2 sessionmgr03:11211,sessionmgr04:11211
osgi> setSkRingSet 2 3 sessionmgr05:11211,sessionmgr06:11211
```

  Log into admin database and verify. You should be able to see such entries in `cache_config` collection.

GR Configuration with Session Replication Across Sites

Login to pcrfclient01/02 to create/update rings from Policy Server (QNS) OSGi console. Assuming there are two session cache replica-sets. By default, Ring-1 Set-1 get configured automatically and remaining rings need to be configured manually.

Configure cluster-1 (Ring-1)

```
osgi> setSkRingSet 1 1 <hostname_primary_site_sessionmgr01>:11211,
<hostname_primary_site_sessionmgr02>:11211
Ring updated
osgi> setSkRingSet 1 2 <hostname_primary_site_sessionmgr03>:11211,
<hostname_primary_site_sessionmgr04>:11211
Ring updated
```

Configure cluster-2 (Ring-2)

```
telnet qns01 9091
osgi> createSkRing 2
Successfully added ring with ID: 2
osgi> setSkRingSet 2 1 <hostname_secondary_site_sessionmgr01>:11211,
<hostname_secondary_site_sessionmgr02>:11211
osgi> setSkRingSet 2 2 <hostname_secondary_site_sessionmgr03>:11211,
<hostname_secondary_site_sessionmgr04>:11211
```

An example configuration is given below:

- Configure cluster-1 (Ring-1):

```
osgi> setSkRingSet 1 1 L2-CA-PRI-sessionmgr01:11211, L2-CA-PRI-sessionmgr02:11211
Ring updated
osgi> setSkRingSet 1 2 L2-CA-PRI-sessionmgr03:11211, L2-CA-PRI-sessionmgr04:11211
Ring updated
```

- Configure cluster-2 (Ring-2):

```
telnet qns01 9091
osgi> createSkRing 2
Successfully added ring with ID: 2
osgi> setSkRingSet 2 1 L2-CA-SEC-sessionmgr01:11211, L2-CA-SEC-sessionmgr02:11211
osgi> setSkRingSet 2 2 L2-CA-SEC-sessionmgr03:11211, L2-CA-SEC-sessionmgr04:11211
```