GR Installation - OpenStack

GR Installation - OpenStack

The examples given in the steps is for your reference only. You need to modify them based on your GR deployments.

Important Copying YAML and environment files from this document is not recommended. The files are provided for your reference only.

Before you begin

- Download the latest ISO build.
- Create CPS VMs using Heat template or Nova boot commands on all GR sites. In the following section, heat template has been considered as an example to deploy GR (here examples are site1, site2 and arbiter) sites.

For more information, refer to *CPS Installation Guide for OpenStack*

Step 1 Create instances for site1, site2 and Arbiter. Wait till they are cluman ready.

Check the readiness status of the Cluster Manager VM on all the sites using the API: `GET http://<Cluster Manager IP>:8458/api/system/status/cluman`.

External replication VLAN information should be added for each VM in the `hot-cps.env` and `hot-cps.yaml` for communication between GR sites.

Refer to Sample Heat Environment File and Sample Heat Template File for sample configuration of site1. For site2 similar files need to be created by modifying hostname, IP addresses and so on.

For Arbiter, refer to Arbiter Installation on OpenStack, on page 5.

Step 2 Load CPS configuration files on each site: Refer to */api/system/config/* section in *CPS Installation Guide for OpenStack*.

GR Installation - OpenStack

In `CPS_system_config.yaml` file, give consideration to the following mentioned items:

- Under Additional Host section, add session manager information of other site (site1 or site2) and arbiter.
- Mongo replica members should include the site identifier to differentiate database host such as, sessionmgr01-site1 from sessionmgr01-site2. Database host names (such as sessionmgr01-site1) needs to be modified according to the your GR deployment in template file.
- Update `policyServerConfig:` section according to your GR deployment.
- Internal/management/external IPs need to modified in `hosts:` and `additionalhosts:` section according to your GR deployment.
- In `additionalhosts:` section, other site session manager host entry should be added with alias `psessionmgr`*xx*.

For sample configurations, refer to Sample YAML Configuration File - site1 and Sample YAML Configuration File - site2.

Note If you want to add the MongoDB authentication, refer to *Configuration Parameters - HA System* section in *CPS Installation Guide for OpenStack*. You need to mention password for all the sites separately using API and that must be same for all the sites.

If you want to enable mongo authentication you need to add the following parameters under `config:` section in YAML configuration file:

```
config:
  dbAuthenticationEnabled: "true"
  dbAuthenticationAdminPasswd: "XXXX"
  dbAuthenticationReadonlyPasswd: "YYYY"
  dbAuthenticationEncryption: "false"
```

where, *XXXX* and *YYYY* are encrypted passwords.

Step 3 (Optional) To confirm the configuration was loaded properly onto the Cluster Manager VM on each site, perform a GET with the API:

`GET http://<Cluster Manager IP>:8458/api/system/config/`

Step 4 Apply the configuration using the following API on each site:

`POST http://<Cluster Manager IP>:8458/api/system/config/apply`

Refer to *Apply the Loaded Configuration* section in *CPS Installation Guide for OpenStack* for more information.

This API applies the CPS configuration file, triggers the Cluster Manager VM to deploy and bring up all CPS VMs on each site, and performs all post-installation steps.

Important Wait for approx 15 minutes for the API to complete the all post-installation steps.

Step 5 In your mongo YAML file, add other site members as secondary-members and local site members as primary members for respective databases depending on your GR deployment.

For sample configuration, refer to Sample Mongo Configuration File - site1 and Sample Mongo Configuration File - site2.

Step 6 After updating the mongo YAML files, apply them using the */api/system/mongo/config* API on each site with their YAML file.

Refer to */api/system/mongo/config* section in *CPS Installation Guide for OpenStack*.

Note This step will not create replica-set for added members. It will create only new mongo configuration file on each site.

Step 7 Add remote site pcrfclient IPs in respective `gr_cluster.yaml` files.

For sample configuration, refer to Sample GR Cluster Configuration File - site1 and Sample GR Cluster Configuration File - site2.

Step 8 Execute below APIs from respective sites to update the GR cluster information and populate respective ADMIN host database.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/application-config -H "Content-Type:
application/yaml" --data-binary @gr_cluster.yaml
```

```
curl -i -X PATCH http://installer-site2:8458/api/system/config/application-config -H "Content-Type:
application/yaml" --data-binary @gr_cluster2.yaml
```

For sample configuration, refer to Sample GR Cluster Configuration File - site1 and Sample GR Cluster Configuration File - site2.

Verify whether:

- Remote pcrfclient IPs are populated correctly in `/etc/broadhop/gr_cluster.conf` file.
- ADMIN database has been populated correctly, run `mongo sessionmgr01-site1:27721/clusters --eval "db.hosts.find()"` and `mongo sessionmgr01-site2:27769/clusters --eval "db.hosts.find()"` on primary database member on site-1 and site-2 console.

Step 9 If all sites are deployed and configured, then create geo replica-sets between site1 and site2:

a) Combine both sites mongo YAML files to be used in your GR deployment.

For sample configuration, refer to Sample Mongo GR Configuration File.

b) After combining YAML files, post the combined file on both sites except arbiter. For more information, refer to */api/system/mongo/config* section in *CPS Installation Guide for OpenStack*.

For example:

```
curl -i -X PUT http://installer-site1:8458/api/system/mongo/config -H "Content-Type:
application/yaml" --data-binary @mongogr.yaml
```

```
curl -i -X PUT http://installer-site2:8458/api/system/mongo/config -H "Content-Type:
application/yaml" --data-binary @mongogr.yaml
```

c) Remove unused replica-sets from site2 using `/var/qps/bin/support/mongo/build_set.sh` script.

In the sample configuration file common SPR, Balance and ADMIN between site1 and site2 are being used, thus these replica-sets can be removed from site2.

d) Add members to replica-set. This API needs to be executed from primary site (site1) only.

For example:

```
curl -i -X POST http://installer-site1:8458/api/system/mongo/action/addMembers
```

e) Configure the priority using the following APIs:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/replica-sets -H "Content-Type:
application/yaml" --data-binary @setPriority-site1.yaml
```

GR Installation - OpenStack

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/replica-sets -H "Content-Type:
application/yaml" --data-binary @setPriority-site2.yaml
```

For sample configuration, refer to Sample Set Priority File - site1 and Sample Set Priority File - site2.

Step 10 Create appropriate clusters in Policy Builder such as, 'Cluster-SITE1' for site1 and 'Cluster-SITE2' for site2 and update Primary Database IP Address, Secondary Database IP Address and Database port number based on mongo configuration and publish to the respective sites depending on your GR deployment.

For more information, refer to Policy Builder Configuration.

Step 11 Run `diagnostics.sh` on both sites to display the current state of the system. Make sure there are no error on both the sites.

Step 12 Modify/add shard on respective sites. It contains each site session replication sets with backup database.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/replica-sets/ -H "Content-Type:
application/yaml" --data-binary @modify_shard.yaml
```

```
curl -i -X PATCH http://installer-site2:8458/api/system/config/replica-sets/ -H "Content-Type:
application/yaml" --data-binary @modify_shard2.yaml
```

For sample configuration, refer to Sample Shard Configuration File - site1 and Sample Shard Configuration File - site2.

Step 13 Modify/add ring: It contains only session replica-sets and not backup database. This API needs to be executed from primary site.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/replica-sets/ -H "Content-Type:
application/yaml" --data-binary @modify_ring.yaml
```

For sample configuration, refer to Sample Ring Configuration File.

Step 14 Add geo-site lookup for both sites.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/application-config -H "Content-Type:
application/yaml" --data-binary @geositelookup.yaml
```

```
curl -i -X PATCH http://installer-site2:8458/api/system/config/application-config -H "Content-Type:
application/yaml" --data-binary @geositelookup2.yaml
```

For sample configuration, refer to Sample Geo Site Lookup Configuration File - site1 and Sample Geo Site Lookup Configuration File - site2.

Note The pattern matching is supported for site lookup mapping. In case the incoming host/realm does not match any of the values configured under LookupValues, request is dropped with the following exception in log:

```
GeoHASiteMappingNotFound - No realm/host to site mapping matched for:
<incoming value>
```

Step 15 Add geo tags in replica-sets for both sites.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/replica-sets/ -H "Content-Type:
application/yaml" --data-binary @modify_geotag.yaml
```

For more information, refer to Sample Geo-tagging Configuration File - site1 and Sample Geo-tagging Configuration File - site2.

Step 16 Add monitor database for both sites.

For example:

```
curl -i -X PATCH http://installer-site1:8458/api/system/config/application-config -H "Content-Type: application/yaml" --data-binary @monitor_db.yaml
```

```
curl -i -X PATCH http://installer-site2:8458/api/system/config/application-config -H "Content-Type: application/yaml" --data-binary @monitor_db2.yaml
```

For sample configuration, refer to Sample Monitor Database Configuration File - site1 and Sample Monitor Database Configuration File - site2.

Arbiter Installation on OpenStack

Before you begin

- Latest ISO Image
- Latest base VMDK
- Glance images
- Cinder Volumes, only for ISO (SVN and mongo are not needed) are created
- Access and Security (22 and mongo port 27717 to 27720 are opened as per deployment)

Note For more information on the above mentioned prerequisites, refer to *CPS Installation Guide for OpenStack*.

Note If you want to add the MongoDB authentication, you need to enable *dbAuthenticationEnabled* parameter. For more information refer to *Configuration Parameters - HA System* section in *CPS Installation Guide for OpenStack*. You need to mention password for all the sites separately using API and that must be same for all the sites.

Step 1 Create flavors by executing the following command:

```
nova flavor-create --ephemeral 0 arbiter auto 4096 0 2
```

Step 2 Cloud init configuration for Arbiter: When Arbiter is launched, `arbiter-cloud.cfg` file needs to be passed via user-data. In order to pass `arbiter-cloud.cfg` file, it should be placed in the directory where the user executes `nova boot` command (likely the path is `/root/cps-install` directory).

Create `arbiter-cloud.cfg` file with the following content:

```
#cloud-config
write_files:
 - path: /etc/sysconfig/network-scripts/ifcfg-eth0
   encoding: ascii
   content: |
     DEVICE=eth0
     BOOTPROTO=none
     NM_CONTROLLED=none
     IPADDR=172.20.38.251           ---> update with your internal address
     NETMASK=255.255.255.0          ---> update with your netmask
     GATEWAY=172.20.38.1            ---> update with your gateway
     NETWORK=172.20.38.0            ---> update with your network
   owner: root:root
   permissions: '0644'
 - path: /var/lib/cloud/instance/payload/launch-params
   encoding: ascii
   owner: root:root
   permissions: '0644'
 - path: /root/.autoinstall.sh
   encoding: ascii
   content: |
     #!/bin/bash
     if [[ -d /mnt/iso ]] && [[ -f /mnt/iso/install.sh ]]; then
       /mnt/iso/install.sh << EOF
     arbiter
     y
     1
     EOF
     fi
     /root/.enable_firewall.sh
     /root/.mongo_auth.sh
     /root/.add_db_hosts.sh
     if [[ -x "/var/qps/install/current/scripts/upgrade/reinit.sh" ]]; then
         /var/qps/install/current/scripts/upgrade/reinit.sh
     fi
   permissions: '0755'
 - path: /root/.enable_firewall.sh
   encoding: ascii
   content: |
     #!/bin/bash
     mkdir -p /etc/facter/facts.d/
     cat <<EOF >/etc/facter/facts.d/qps_firewall.txt
     firewall_disabled=0            ---> change it to 1 if you do not want firewall enabled on this
 setup and remove below fields
     internal_address=172.20.38.251 ---> update with your internal address
     internal_device=0
     EOF
   permissions: '0755'
- path: /root/.mongo_auth.sh
  encoding: ascii
  content: |
    #!/bin/bash
    mkdir -p /etc/facter/facts.d/
    cat <<EOF >/etc/facter/facts.d/mongo_auth.txt
    db_authentication_enabled=FALSE     ---> if mongo-auth enable then make it to TRUE
    db_authentication_admin_passwd=     ---> provide admin user encrypted password if enable
    db_authentication_readonly_passwd=  ---> provide readonly user encrypted password if enable
    EOF
  permissions: '0755'
 - path: /root/.add_db_hosts.sh  ---> update db hosts IP as per requirement
   encoding: ascii
   content: |
     #!/bin/bash
     #Example if /etc/broadhop/mongoConfig.cfg:
```

```
    #[SESSION-SET1]
    #SETNAME=set01
    #OPLOG_SIZE=5120
    #ARBITER1=arbiter-site3:27717
    #ARBITER_DATA_PATH=/var/data/sessions.1/set01
    #PRIMARY-MEMBERS
    #MEMBER1=sessionmgr01-site1:27717
    #MEMBER2=sessionmgr02-site1:27717
    #SECONDARY-MEMBERS
    #MEMBER1=sessionmgr01-site2:27717
    #MEMBER2=sessionmgr02-site2:27717
    #DATA_PATH=/var/data/sessions.1/set01
    #[SESSION-SET1-END]
    #For above mongoConfig.cfg below hosts entries are needed in /etc/hosts, edit below list as per
 your requirement
    cat <<EOF >> /etc/hosts
    192.168.1.1 arbiter-site3
    192.168.1.2 sessionmgr01-site1
    192.168.1.3 sessionmgr02-site1
    192.168.1.4 sessionmgr01-site2
    192.168.1.5 sessionmgr02-site2
    EOF
  permissions: '0755'
mounts:
 - [ /dev/vdb, /mnt/iso, iso9660, "auto,ro", 0, 0 ]
runcmd:
 - ifdown eth0
 - echo 172.20.38.251 installer arbiter >> /etc/hosts ---> update this IP
 - ifup eth0
 - /root/.autoinstall.sh
```

Note Edit IPADDR/NETMASK/NETWORK/GATEWAY and remove the hint information while using the `cloud-config` file. For example, internal network information and so on.

Step 3 Create Arbiter VM:

Note As DHCP has been disabled in the prep script, the `arbiter-cloud.cfg` file needs to be passed to the arbiter to assign IP addresses to arbiter interfaces.

Before executing `nova boot` command, confirm that the cloud configuration file (`arbiter-cloud.cfg`) exists in the right directory.

Execute the following command to create arbiter VM with two NICs:

```
source ~/keystonerc_core
nova boot --config-drive true --user-data=arbiter-cloud.cfg  --file
/root/keystonerc_user=/root/keystonerc_core
--image "base_vm" --flavor "arbiter"
--nic net-id="9c89df81-90bf-45bc-a663-e8f80a8c4543,v4-fixed-ip=172.16.2.19"
--nic net-id="dd65a7ee-24c8-47ff-8860-13e66c0c966e,v4-fixed-ip=172.18.11.101"
--block-device-mapping "/dev/vdb=eee05c17-af22-4a33-a6d9-cfa994fecbb3:::0"
--availability-zone "az-2:os24-compute-2.cisco.com" arbiter
```

For example,

```
nova boot --config-drive true --user-data=arbiter-cloud.cfg
--file /root/keystonerc_user=/root/keystonerc_core
--image "base_vm" --flavor "arbiter" --nic net-id="<Internal n/w id>,v4-fixed-ip=<Interanl n/w private
 ip>"
--nic net-id="<Management n/w id>,v4-fixed-ip=<Management n/w public ip>"
--block-device-mapping "/dev/vdb=<Volume id of iso>:::0"
--availability-zone "<availability zone:Host info>" arbiter
```

The following examples can be used to get the internal and management IP addresses and volume IDs which are used to spin arbiter VM.

```
source ~/keystonerc_core
```

neutron net-list

id	name	subnet
9c89df81-90bf-45bc-a663-e8f80a8c4543	internal	682eea79-6eb4-49db-8246-3d94087dd487 172.16.2.0/24
8d60ae2f-314a-4756-975f-93769b48b8bd	gx	9f3af6b8-4b66-41ce-9f4f-c3016154e027 192.168.2.0/24
dd65a7ee-24c8-47ff-8860-13e66c0c966e	management	a18d5329-1ee9-4a9e-85b5-c381d9c53eae 172.18.11.0/24

nova volume-list

ID	Status	Display Name	Size	Volume Type	Attached to
146ee37f-9689-4c85-85dc-c7fee85a18f4	available	mongo2	60	None	
6c4146fa-600b-41a0-b291-11180224d011	available	mongo01	60	None	
181a0ead-5700-4f8d-9393-9af8504b15d8	available	snv02	2	None	
158ec525-b48b-4528-b255-7c561c8723a9	available	snv02	2	None	
eee05c17-af22-4a33-a6d9-cfa994fecbb3	available	cps-production-7.9.9-SNAPSHOT.iso	3	None	

For the kernel upgrade, once the Arbiter deployment is complete and the output of `diagnostics.sh` command displays no errors, execute the following command from Cluster Manager to ensure that the kernel version is upgraded.

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

This command prompts for reboot choice. Please select **Y** for the same and proceed.

Multiple Arbiter Installation - OpenStack

Step 1 Update the arbiter member information in YAML file.

Example:

```
- title: "SESSION-SET1"
  setName: "set01"
  oplogSize: "5120"
  arbiters:
   - "arbitervip:27717"
   - "sessionmgr13:27717"
   - "sessionmgr14:27717"
  arbiterDataPath: "/var/data/sessions.1"
```

```
members:
 - "sessionmgr01:27717"
 - "sessionmgr02:27717"
 - "sessionmgr03:27717"
dataPath: "/var/data/sessions.1/1"
hotStandBy: "true"
shardCount: "4"
seeds: "sessionmgr01:sessionmgr02:27717,sessionmgr02:sessionmgr03:27717"
```

Note Hotstandby replica sets must be on different ports.

Step 2 Load the updated YAML file in Cluster Manager.

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/mongo/config
- **Header:** Content-Type: application/yaml
- **Method:** PUT/GET (PATCH for creation and deletion of replica-set)
- **Payload:** Include the YAML configuration file in the PATCH request. The entire contents of the configuration (same as in Step 1, on page 8) must be included.

Step 3 Add arbiters from loaded YAML file.

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/mongo/action/addMembers
- **Header:** Content-Type: application/yaml
- **Method:** POST (PATCH for creation and deletion of replica-set)
- **Payload:** None

Note This API returns immediately and does not wait for the arbiters to be added.

Step 4 Check the configured replica-set in mongo.

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/config/replica-sets
- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** None

Step 5 Check the mongo configuration.

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/mongo
- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** None

Configuration Parameters - GR System

`grConfig` section under `applicationConfig` holds configuration for all GR related configurations. The following parameters can be defined in the CPS configuration file for GR system.

All parameters and values are case sensitive.

Note Before loading the configuration file to your CPS cluster, verify that the YAML file uses the proper syntax.

Various configuration files like, qns.conf, mon_db* related configuration files, gr_cluster.conf files have been modified to support GR installation using API.

- `policyServerConfig`
- `dbMonitorForQns`
- `dbMonitorForLb`
- `clusterInfo`

policyServerConfig

`policyServerConfig` holds configuration for `/etc/broahdop/qns.conf` file and supported parameters in it.

In `policyServerConfig`, a new parameter `deploymentType` has been added which is not a part of `qns.conf` file which is used for validation of `qns.conf` file parameters. It can have values for HA or GR deployments. By default, the value is set to GR. In case of GR, validation for required parameters in configuration will happen.

For the parameter descriptions, consult your Cisco Technical Representative.

Table 1: policyServerConfig Parameters

qns.conf Parameter	Corresponding Parameter in policyServerConfig
-DGeoSiteName	geoSiteName
-DSiteId	siteId
-DRemoteSiteId	remoteSiteId
-DheartBeatMonitorThreadSleepMS	heartBeatMonitorThreadSleepMS
-Dcom.mongodb.updaterConnectTimeoutMS	mongodbupdaterConnectTimeoutMS
-Dcom.mongodb.updaterSocketTimeoutMS	mongodbupdaterSocketTimeoutMS
-DdbConnectTimeout	dbConnectTimeout
-Dmongo.client.thread.maxWaitTime	threadMaxWaitTime

qns.conf Parameter	Corresponding Parameter in policyServerConfig
-DdbSocketTimeout	dbSocketTimeout
-DclusterFailureDetectionMS	clusterFailureDetectionMS
-Dremote.locking.off	remoteLockingOff
-DapirouterContextPath	apirouterContextPath
-Dua.context.path	uaContextPath
-Dcom.cisco.balance.dbs	balanceDbs
-DsprLocalGeoSiteTag	sprLocalGeoSiteTag
-DbalanceLocalGeoSiteTag	balanceLocalGeoSiteTag
-DsessionLocalGeoSiteTag	sessionLocalGeoSiteTag
-DclusterPeers	clusterPeers
-DgeoHASessionLookupType	geoHaSessionLookupType
-DisGeoHAEnabled	isGeoHaEnabled
-DmaxHash	maxHash
-DdbSocketTimeout.remoteBalance	dbSocketTimeoutRemoteBalance
-DdbConnectTimeout.remoteBalance	dbConnectTimeoutRemoteBalance
-Dmongo.connections.per.host.remoteBalance	mongoConnHostRemoteBalance
-Dmongo.threads.allowed.to.wait.for. connection.remoteBalance	waitThreadNumRemoteBalance
-Dmongo.client.thread.maxWaitTime.remoteBalance	threadWaitTimeRemoteBalance
-DdbSocketTimeout.remoteSpr	dbSocketTimeoutRemoteSpr
-DdbConnectTimeout.remoteSpr	dbConnectTimeoutRemoteSpr
-Dmongo.connections.per.host.remoteSpr	mongoConnHostRemoteSpr
-Dmongo.threads.allowed.to.wait.for.connection.remoteSp	waitThreadNumRemoteSpr
-Dmongo.client.thread.maxWaitTime.remoteSpr	threadWaitTimeRemoteSpr
-DenableReloadDictionary	enableReloadDict
-Dcom.broadhop.q.if	replicationIface
-DRemoteGeoSiteName	remoteGeoSiteName
-Dcom.broadhop.run.clusterId	clusterId

dbMonitorForQns and dbMonitorForLb

`dbMonitorForQns` holds configuration for `/etc/broadhop/mon_db_for_callmodel.conf` file and supported parameters in it.

Note The YAML file is used to update the `mon_db_for_callmodel.conf` file and not to overwrite/delete the existing entries.

`dbMonitorForLb` holds configuration for `/etc/broadhop/mon_db_for_lb_failover.conf` file and supported parameters in it.

```
applicationConfig:
dbMonitorForLb:
     setName:
       - "SPR-SET1"
       - "BALANCE-SET1"
       - "SESSION-SET1"
       - "ADMIN-SET1"
dbMonitorForQns:
     stopUapi: "true"
     setName:
       - "SPR-SET1"
       - "BALANCE-SET1"
       - "SESSION-SET1"
```

`monQnsLB: "true"` must be added under `config:` (`api/system/config/config`) section in YAML file to stop Policy Server (QNS) processes from lb01/lb02 when all the policy services are down (that is, qns01,02..n). Once policy server processes from lb01/lb02 go down, above configuration make sure that the traffic switchover takes place.

For `mon_db*` config, `setName` is an array of set names and corresponds to `title` in YAML for `replicaSet` configuration. The following is an example configuration:

```
---- title: "SESSION-SET1"
     setName: "set01"
     oplogSize: "1024"
     arbiters:
      - "arbiter-site3:27717"
     arbiterDataPath: "/var/data/sessions.1"
     primaryMembers:
```

clusterInfo

`clusterInfo` section under `grConfig` holds configuration for `gr_cluster.conf` file and supported parameters in it.

YAML will have following format for clusterInfo:

- remotePcrfclient01IP: Specifies remote sites pcrfclient01 IP. You can specify IPv4 or IPv6 address.
- remotePcrfclient02IP: Specifies remote sites pcrfclient02 IP. You can specify IPv4 or IPv6 address.

Note If you want to use IPv6 address of pcrfclient, then it has to be done in [] brackets.

For Example (for IPv6):

```
grConfig:
  clusterInfo:
    remotePcrfclient01IP: "[fd00:854::231]"
    remotePcrfclient02IP: "[fd00:854::232]”
```

When user specifies cluster info details, local site details are fetched from existing configuration and based on all information `gr_cluster.conf` is updated which populates admin database with cluster information.

Example Requests and Response

Retrieve Current Configuration

To retrieve (GET) the current configuration:

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/config/application-config

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, refer to *HTTPS Support for Orchestration API* section in *CPS Installation Guide for OpenStack*.

- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error
  - Example Response (YAML format):

```
---
policyServerConfig:
       geoSiteName: "SITE1"
       clusterId: "Cluster-SITE1"
       siteId: "SITE1"
       remoteSiteId: "SITE2"
       heartBeatMonitorThreadSleepMS: "500"
       mongodbupdaterConnectTimeoutMS: "1000"
       mongodbupdaterSocketTimeoutMS: "1000"
       dbConnectTimeout: "1200"
       threadMaxWaitTime: "1200"
       dbSocketTimeout: "600"
       remoteLockingOff: ""
       apirouterContextPath: ""
       uaContextPath: ""
       balanceDbs: ""
       clusterPeers: ""
       isGeoHaEnabled: "true"
       geoHaSessionLookupType: "realm"
       enableReloadDict: "true"
```

```
      sprLocalGeoSiteTag: "SITE1"
      balanceLocalGeoSiteTag: "SITE1"
      sessionLocalGeoSiteTag: "SITE1"
      deploymentType: "GR"
dbMonitorForQns:
 stopUapi: "true"
 setName:
  - "SESSION-SET1"
dbMonitorForLb:
 setName:
  - "SESSION-SET1"
```

Note In case there is an error in configuring `geoHaSessionLookupType`, CPS behaves incorrectly and drop messages. The following logs come continuously if there is an error in the configuration:

```
GeoHA is enabled, unknown lookuptype is configured:
<>. Possible values are…
```

Update Configuration

When this API call completes, the Cluster Manager configuration is updated and all new VMs are deployed asynchronously.

Note The amount of time needed to complete the process depends on the number of VMs being deployed.

Use this API to load an updated configuration on the CPS Cluster Manager: You can specify this configuration during fresh install time or also at a later stage once system is deployed using PATCH. The following information gives details about PATCH method:

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/mongo/config/application-config

  

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, refer to *HTTPS Support for Orchestration API* section in *CPS Installation Guide for OpenStack*.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:** Include the YAML configuration file in the PATCH request. The entire contents of the configuration must be included.
- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error

  Example Response (YAML format):

Note After using this API to load the updated configuration, you must apply the configuration.

```
curl -i -X PATCH http://installer:8458/api/system/config/application-config -H
"Content-Type: application/yaml" --data-binary @mondblb.yaml


HTTP/1.1 200 OK
Date: Fri, 19 Aug 2016 10:31:49 GMT
Content-Length: 0
```

`cat mondblb.yaml`: The following is an example request to change mon_db* script configuration:

```
dbMonitorForLb:
      setName:
        - ADMIN-SET1
        - BALANCE-SET1
```