Orchestration API

Important After the configuration is complete, take a backup of the configuration to be used in case there is an issue with configuration at a later stage. For more information on taking the backup, refer to *CPS Backup and Restore Guide*.

Installation APIs

Input and Output Formats

The CPS Orchestration API supports both YAML and JSON formats for both inputs (request payload) and outputs (response payloads).

The input format is specified by the "Content-Type" attribute in the header. The input format is mandatory if the request includes a message body; it must be specified in the header for any API such request.

The output format is specified by the "Accept" attribute in the header. The output format is optional.

The following formats are supported for Content-Type and Accept attributes:

- application/json
- application/yaml
- text/yaml

The default output format (if the Accept attribute is not specified) for all APIs is always application/json except for following APIs, for which the default output format is text/yaml:

- /api/system/config
- /api/system/config/additional-hosts

- /api/system/config/hosts
- /api/system/config/replica-sets
- /api/system/mongo/config

/api/system/status/cluman

Purpose

This API returns the readiness status of the Cluster Manager VM.

Cluster Manager VM Readiness

If `/mnt/iso/install.sh` is executing, the status is returned as 'not ready'.

If `/mnt/iso/install.sh` has completed executing, status is returned as 'ready'.

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** GET
- **Payload:** JSON
- **Response:** 200 OK: success

  The following example shows the status reported for a new CPS deployment:

```
{
    "status": "ready",
}
```

API logs are at written to: `/var/log/orchestration-api-server.log`

/api/system/config/

Purpose

This API is used to load an initial configuration or return (GET) the current CPS cluster configuration.

This API is also used to apply the loaded configuration to all VMs within the CPS cluster.

API logs are at written to: `/var/log/orchestration-api-server.log`

Retrieve the Current Configuration

To retrieve (GET) the current CPS cluster configuration that is loaded on the CPS Cluster Manager:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** There is no payload.
- **Response Codes:** 200: OK.

  Example Response (No Configuration Present) XML:

```
---
configVersion: null
hosts: null
vlans: null
additionalHosts: null
config: null
licenses: null
replicaSets: null
```

  For a response showing an example configuration file refer to Sample YAML Configuration File - HA Setup, on page 16.

Load a Configuration

Note This API can only be used once for initial deployment. Once a configuration has been applied (/system/config/action/apply) as described below, this API is no longer available.

Note Before loading the configuration file to your CPS cluster, verify that the YAML file uses the proper syntax. There are many publicly-available websites which you can use to validate your YAML configuration file.

Note When this API is issued, the following basic validations are performed on the consolidated configuration (YAML) file submitted in the payload:

- The replica set hosts are included in hosts or additionalHosts section
- Standard CPS aliases are present (lb01, lb02, and so on)
- Standard CPS vlan names are present (Internal, Management, and so on)
- Range checking (for example, IPv4/IPv6 IP address syntax validation)
- Cross-referencing of vlans with hosts

If a validation error is detected, an appropriate message is provided in the API response, and reported in `/var/log/orchestration-api-server.log`.

To load a new CPS cluster configuration on the CPS Cluster Manager:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** POST
- **Payload:** Include the YAML configuration file in the POST. Refer to Sample YAML Configuration File - HA Setup, on page 16 for more information about this configuration file.
- **Response:** 200: success; 400: malformed or invalid; 403: Configuration may not be changed at this time (for example, after it has been applied).

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/

Apply the Loaded Configuration

Note This API can only be used once for initial deployment. After a configuration has been applied, the API is no longer available.

Once a new configuration file has been uploaded to the Cluster Manager VM, you must apply the configuration. This triggers the Cluster Manager VM prepare and push out the new configurations to all VMs in the cluster, as well as perform any post-update steps.

During an initial deployment of a CPS cluster, the CPS VMs in the cluster will remain in an inactive/waiting state until this configuration file is applied.

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** POST
- **Payload:** There is no payload.
- **Response:** 200: success; 400: malformed or invalid; 403: Configuration may not be applied at this time; 500: System error. See logs.

To check the status of the CPS cluster after applying a configuration, refer to /api/system/config/status , on page 23.

Encrypt Administration Traffic Parameters

The administration traffic parameters (rysyslog, haproxy, SNMPv3) can be configured under the “config:” section which defines general global parameters used to deploy CPS.

Important For fresh installation, in case the parameters *rsyslog_tls* and *rsyslog_ca* are not set, they would be initialized to default values and feature would be enabled. If the user wants to disable the feature *rsyslog_tls* should be set to FALSE.

Similarly, for *haproxy_stats_tls*, if no value is set (TRUE or FALSE), the default value (TRUE) is used and the feature is enabled.

For SNMPv3, until the snmpv3 tag is not commented out, the feature would not be enabled.

Note For upgrade scenario, if parameters are not defined they are initialized to empty.

Table 1: Traffic Parameters

Parameter	Description
rsyslog_tls	This field is used to enable or disable encryption for rsyslog. Default: TRUE
rsyslog_cert	This field is used to define the path for trusted Certificate of server.
rsyslog_ca	This field is used to define the Path of certifying authority (CA). Default: `/etc/ssl/cert/quantum.pem`
rsyslog_key	This field is used to define the path of private key.
haproxy_stats_tls	This field is used to enable or disable the encryption for HAproxy statistics. Default: TRUE

Sample YAML format (for enabling SNMPv3):

```
config:
 # enable SNMP V3.
# If null, SNMP V3 is disabled.
# To enable add the following:
# v3User: The SNMP V3 user: REQUIRED
# engineId: hex value (ie, 0x0102030405060708): REQUIRED
# authProto: SHA or MD5: REQUIRED
# authPass: at least 8 characters: REQUIRED
# privProto: AES or DES: REQUIRED
# privPass: OPTIONAL
```

```
snmpv3:
v3User: "cisco_snmpv3" #---->Default value. You can change as per your deployment requirements
engineId: "0x0102030405060708"
authProto: "SHA"
authPass: "cisco_12345"
privProto: "AES"
privPass: ""rsyslogTls: "TRUE"
```

Sample YAML format (for rsyslog, haproxy):

```
config:
rsyslogCa: "/etc/ssl/certs/quantum.pem"
rsyslogCert: "/etc/ssl/cert/quantum.pem"
rsyslogKey: "/etc/ssl/cert/quantum.key"
haproxyStatsTls: "TRUE"
```

Configuration Parameters - HA System

The following parameters can be defined in the CPS configuration file. Refer also to: Sample YAML Configuration File - HA Setup, on page 16.

In this file, the Internal, Management and Gx networks must have an exact case match of "Internal", "Management" and " Gx" in the following sections:

- hosts: interfaces: value of "network"
- vlans: value of "name"

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. There are many publicly-available websites which you can use to validate your YAML configuration file.

Table 2: Configuration Parameters - HA System

Parameter	Description
`configVersion`	The version of the configuration file. This must be set to `configVersion: 1.0`.
`hosts:`	This section defines the host entries for each of the CPS VMs in the deployment.
`- name:`	Defines the host name of the VM. This name must be resolvable in the enterprise DNS environment. **Note** CPS host names must conform to RFC 952 and RFC 1123; characters such as "_" are not allowed.
`alias:`	Defines the internal host name used by each CPS VMs for internal communication, such as lb0x, pcrfclient0x, sessionmgr0x, or qns0x.
`interfaces`	This section defines the network details for each VM.
`network:`	The network name which must match a VLAN name (see below).

Parameter	Description
ipAddress:	The IP interface address.
vlans:	This section defines the separate VLANs to be configured. The "Internal" and "Management" VLANs are always needed. For additional networks, add more as needed.
- name:	Defines the name for a particular VLAN. It is recommended to use a name representing the network for certain traffic. The VLAN names defined here must be used in the `network` field in the `hosts` section above. The "Internal" VLAN Name is always needed. Names must consist only of alphanumeric characters and underscores, and must not start with a number.
vipAlias:	The hostname associated with virtual interfaces on the Policy Directors (LBs), typically "Internal", "Management", and "Gx".
vip:	The Virtual IP address used on this VLAN. The virtual addresses are used to distribute the traffic between two Policy Directors. If using IPv6, the address must be specified in canonical form as described in RFC5929.
guestNIC:	The Name of the interface specified in the host cloud config or Heat definition.
pcrfVipAlias:	The OAM (pcrfclient) vip alias.
additionalHosts	This section defines any hosts not configured in the hosts section above. **Note** Policy Director (LB) VIPs are defined in this section as '`lbvip01`' and '`lbvip02`', as well as the '`arbitervip`' which defines the prcfclient01 internal IP. In a CPS cluster which is configured with more than 2 Policy Directors (LBs), HAproxy and the VIPs are hosted only on LB01 and LB02. The additional LBs serve only as diameter endpoints to route diameter traffic. Any other hosts which CPS must interact with, such as NTP or NMS servers, must be defined in this section. Any hosts defined here are added to each CPS VM `/etc/hosts` file. **Note** The host names defined here do **not** need to conform to RFC 952 and RFC 1123.
- name:	The hostname of the host.
alias:	The internal host name used by CPS nodes for internal communication, such as qns01.
ipAddress:	The IP address to use in the `/etc/hosts` file.
config:	This section defines general global parameters used to deploy CPS.
qpsUser:	Do not change.
selinuxState:	Do not change. Security Enhanced Linux (SELinux) support: disabled | enforcing. Default: disabled

Parameter	Description
`selinuxType:`	Do not change.
`broadhopVar:`	Do not change. Default: broadhop
`tacacsEnabled:`	Enter TRUE to enable TACACS+ authentication. For more information, refer to the *CPS Installation Guide for VMware*. Default: FALSE
`tacacsServer:`	Defines the IP address of the TACACS+ server.
`tacacsSecret:`	Defines the password/secret of the TACACS+ server.
`tacacsService`	A string value indicating which service to be used when authorizing and auditing against the TACACS+ servers. Default: `pcrflinuxlogin` if no value is specified.
`tacacsProtocol`	A string value indicating which protocol to be used when authorizing and auditing against the TACACS+ servers. Default: `ssh`
`tacacsTimeout`	An integer that represents how long the software needs to wait, in seconds, for the TACACS+ server to respond to the queries. Default: `5`
`tacacsDebug`	An integer value indicating the debug level to run the software in. Currently, this is effectively boolean. Default: 0
`redisServerCount`	This value specifies the number of redis server instances running on each policy director (lb) VM. By default, redis is enabled with the number of instances as defined in *redisServerCount*. If the value for redis server count is not provided, default value of 3 for redisServerCount is considered. To disable redis explicitly, redis server count should have value 0. Default: 3 Value range: 0 to 64

Parameter	Description
redisForLdapRequired	This parameter is used only when dedicated LDAP instance is required. Default: false Possible values: true, false If you configure LDAP instance explicitly, first redis instance on policy director (lb) VMs running on port 6379 is used for LDAP and the remaining is used for diameter. **Note** If you configure redisForLdapRequired parameter, then the following changes are automatically added in configuration files. In /etc/broadhop/qns.conf file, an additional parameter -DldapRedisQPrefix=ldap is added. /etc/broadhop/redisTopology.ini file has the following content if redisForLdapRequired=true and redisServerCount=3: ldap.redis.qserver.1=lb01:6379 policy.redis.qserver.2=lb01:6380 policy.redis.qserver.3=lb01:6381 ldap.redis.qserver.4=lb02:6379 policy.redis.qserver.5=lb02:6380 policy.redis.qserver.6=lb02:6381 If a dedicated ldap instance is required, you many also want to consider increasing the total redis servers to accommodate the diameter traffic. For example, if redisForLdapRequired property was not configured, and redisServerCount=3 then after configuring redisForLdapRequired as true, you want to increase total redis server count to 4 by setting redisServerCount=4.
databaseNics	This parameter allows user to provide interface names on which the firewall is opened for replica-set on a VM. If databaseNics is not configured, firewall is opened only for internal interface for a replica-set. If databaseNics is configured, then firewall is opened for configured interfaces and internal interface as well (even if it is not mentioned in databaseNics). This field has comma (,) or semicolon (;) separated interface names for firewall ports to be opened for a replica-set on a VM. **Note** This field is effective only when the firewall is enabled.
freeMemPer:	By default, a low memory alert is generated when the available memory of any CPS VM drops below 10% of the total memory. To change the default threshold, enter a new value (0.0-1.0) for the alert threshold. The system generates an alert trap whenever the available memory falls below this percentage of total memory for any given VM. Default: 0.10 (10% free).

Parameter	Description
`syslogManagers:`	Entries are space separated tuples consisting of protocol:hostname:port. Only UDP is supported at this time. Default: 514. For example: udp:corporate_syslog_ip:514 udp:corporate_syslog_ip2:514
`syslogManagersPorts:`	A comma separated list of port values. This must match values in the syslog_managers_list.
`logbackSyslogDaemonPort:`	Port value for the rsyslog proxy server to listen for incoming connections, used in the rsyslog configuration on the Policy Director (lb) and in the logback.xml on the OAM (pcrfclient). Default: 6515
`logbackSyslogDaemonAddr:`	IP address value used in the `/etc/broadhop/controlcenter/logback.xml` on the OAM (pcrfclient). Default: lbvip02
`cpuUsageAlertThreshold:`	The following cpu_usage settings are related to the High CPU Usage Alert and High CPU Usage Clear traps that can be generated for CPS VMs. Refer to the *CPS SNMP, Alarms and Clearing Procedures Guide* for more details about these SNMP traps. Set the higher threshold value for CPU usage. The system generates an Alert trap whenever the CPU usage is higher than this value.
`cpuUsageClearThreshold:`	The lower threshold value for CPU usage. The system generates a Clear trap whenever the CPU usage is than this value and Alert trap is already generated.
`cpuUsageTrapIntervalCycle:`	The interval period to execute the CPU usage trap script. The interval value is calculated by multiplying five with the given value. For example, if set to one, then the script is executed every five seconds. The default value is 12, which means the script is executed every 60 seconds.
`snmpTrapCommunity:`	The SNMP trap community string. Default: broadhop
`snmpRoCommunity:`	This value is the SNMP read-only community string. Default: broadhop
`monQnsLb:`	Do not change.
`freeMemoryPerAlert:`	By default, a low memory alert is generated when the available memory of any CPS VM drops below 10% of the total memory. To change the default threshold, enter a new value (0.0-1.0) for the alert threshold. The system generates an alert trap whenever the available memory falls below this percentage of total memory for any given VM. Default: 0.10 (10% free)

Parameter	Description
`freeMemoryPerClear:`	Enter a value (0.0-1.0) for the clear threshold. The system generates a low memory clear trap whenever available memory for any given VM is more than 30% of total memory. Default: 0.3 (30% of the total memory)
`monitorReplicaTimeout:`	This value is used to configure the replica-set timeout value. The default value is 540 seconds considering four replica sets. The customer can set timeout value according to the number of replica sets in their network. To recover a single session replica-set, it takes approximately 120 sec and adding 20% buffer to it; we are using 540 sec for default (for four replica sets). Without any latency between sessionmgr VMs, one replica-set recovers in ~135 seconds. If latency (40 -100 ms) is present between sessionmgr VMs, add a 10% buffer to 135 seconds and set the timeout value for the required number of replica sets in the deployment.
`sctpEnabled:`	Enables (`TRUE`) or disables (`FALSE`) Stream Control Transmission Protocol (SCTP) support for Diameter interfaces. Default: `TRUE`
`firewallState:`	Enables or disables linux firewall (IPtables) on all VMs. Valid Options: enabled / disabled Default: enabled

Parameter	Description
`snmpv3:`	Enable SNMPv3 support within CPS by deleting `null` and uncommenting (removing #) the following snmpv3 object parameters: • `v3User`: Username to be used for SNMPv3 request/response and trap. This parameter is required. Default: cisco_snmpv3 • `engineId`: This value is used for SNMPv3 request/response and on which NMS manager can receive the trap. It must be a hex value. This parameter is required. Default: 0x0102030405060708 • `authProto`: SHA or MD5. This value specifies the authentication protocol to be used for SNMPv3. This parameter is required. Default: SHA • `authPass`: This value specifies the authentication password to be used for SNMPv3 requests. It should have minimum length as 8 characters. This parameter is required. Default: cisco_12345 • `privProto`: This value specifies Privacy/Encryption protocol to be used in SNMPv3 request/response and SNMP trap. User can use AES/DES protocol. This parameter is required. Default: AES • `privPass`: This value specifies Privacy/Encryption password to be used in SNMPv3. If it is blank then value specified in `authPass` is used as privPass. This parameter is optional. Default: *blank (no value)*
`snmpRouteLan:`	This field contains the value of a VLAN name which can be used to access the KPIs value provided by SNMP. Default: Management
`remoteClumanIp:`	This parameter is used for GR deployments to synchronize mongo configuration across sites. For more information, refer to /api/system/config/replica-sets/action/sync-mongo, on page 45.
`dbAuthenticationEnabled:`	This field is used to enable or disable MongoDB authentication. By default, MongoDB authentication is disabled. Possible value: true or false For MongoDB authentication process, refer to MongoDB Authentication Process, on page 22.
`dbAuthenticationAdminPasswd:`	This parameter is the plain or encrypted password for admin user depending on the value set in `dbAuthenticationEncryption` parameter.

Parameter	Description
`dbAuthenticationReadonlyPasswd:`	This parameter is the plain or encrypted password for readonly user depending on the value set in `dbAuthenticationEncryption` parameter.
`dbAuthenticationEncryption:`	If this parameter is false, then the `dbAuthenticationAdminPasswd` and `dbAuthenticationReadonlyPasswd` are in plain text. **Note** Make sure to remove the `dbAuthenticationAdminPasswd` and `dbAuthenticationReadonlyPasswd` fields from your input YAML file after configuring API. If this parameter is true, then the encrypted password needs to be configured. For encrypted passwords, you need to SSH to a Cluster Manager and execute the following command: `/var/qps/bin/support/mongo/encrypt_passwd.sh <Password>` Default: false
`enableSshLoginSecurity:`	This parameter allows user to enable or disable SSH login security. Default: disabled Possible values: enabled, disabled
`cpsAdminUserCluman:`	This parameter is used to configure Cluster Manager administrator user.
`cpsAdminPasswordCluman:`	This parameter is the encrypted password for administrator user.
`whitelistedHostsForSsh:`	Valid values are an array of whitelisted hosts specified in string for which SSH access needs to be allowed. This configuration is effective only when the SSH login security is enabled. If the hostname is mentioned then it should be resolvable by CPS VM's. No validation on hostname/IP addresses is provided. You can specify both IPv4/IPv6 address. **Note** New whitelisted host list overwrites the old list. If the new whitelist host configuration is empty then all old additional whitelisted hosts (apart from standard local CPS VM's host) are deleted.
`sysUsers:`	This section defines CPS system users.
`- name:`	The username of this user.
`password:`	The password must be encrypted for this user. Refer to the section *Password Encryption* in *CPS Installation Guide for VMware* for instructions to generate an encrypted password.
`groups:`	This section defines the groups to which this user belongs. **Note** User group can be qns-svn, qns-ro, qns-su, qns-admin and pwauth. pwauth group is valid only for `qns` username and no other username.
`- <group>`	List each group on a separate line.
`hvUsers`	This section defines the hypervisor users.

Parameter	Description
- name:	The username of a user with root access to the host/blade. If installing CPS to multiple blade servers, it is assumed that the same username and password can be used for all blades.
password:	The password for this user. To pass special characters, they need to be replaced with the "% Hex ASCII" equivalent. For example, "$" would be "%24" or "hello$world" would be "hello%24world".
additionalUsers:	This section defines additional CPS system users, such as those given access to Control Center.
- name:	The username of this user.
password:	The clear text password for this user.
groups:	This section defines the groups to which this user belongs.
- *<group>*	List each group on a separate line.
licenses:	This section is used to enter the CPS license information. Contact your Cisco representative to receive your CPS license key(s).
- feature:	The name of the feature license, for example: "MOBILE_CORE".
license:	The license key for this feature.
replicaSets:	This section defines the CPS MongoDB replica sets.
- title:	The database for which the replica set is being created.
setName:	The name of the replica set.
oplogSize:	MongoDB operations log (oplog) size, in MB. Default: 5120
arbiter:	The hostname and port of the arbiter.
arbiterDataPath:	The data directory on the arbiter VM.
members:	The list of members for the replica set. Each list element is a session manager hostname:port. For example, sessionmgr01:27718.
- *<member>*	List each member hostname:port on a separate line.
dataPath:	The data directory path on the Session Manager VM.
LDAP SSSD	For more information, refer to LDAP SSSD, on page 15.

LDAP SSSD

Note For LDAP SSSD routable IP is required. LDAP server must be accessible from CPS VMs (LDAP client).

Table 3: LDAP SSSD

Parameter	Description
`ldapOnAll:`	When set to true, it installs the LDAP SSSD on all CPS VMs. When set to false, it install the LDAP SSSD only on pcrfclient/policy directors (lb) VMs. **Note** true or false must be in small case.
`ldapEnabled:`	When set to true, applies the SSSD configuration as per input provided by user. When set to false, use the default configuration. **Note** true or false must be in small case.
`ldapServer:`	Contains server IP:port to configure LDAP. Format: ldaps://<*serverip*>:<*port*>
`ldapSearchBase:`	This is required for SSSD configuration. The default base DN to use for performing LDAP user operations. Format: ou=users,dc=cisco,dc=com
`ldapDefaultBindDn:`	The default bind DN to use for performing LDAP operations. Format: uid=admin,ou=system
`ldapSecret:`	The authentication token for the default bind DN. Currently, only clear text passwords are supported. For example, secret
`ldapDefaultUser:`	The default LDAP user to be configured in LDAP server. For example, admin
`ldapOuUser:`	The default LDAP user OU. For example, users
`ldapOuGroup:`	The default LDAP group user OU. For example, groups

Parameter	Description
`ldapDefaultGroup:`	The LDAP attribute that corresponds to the group name. For example, Admin
`ldapDefaultGroupEditor:`	This is a user group which has the editor access to Grafana. For example, User
`ldapDcName:`	This is a single entity of all domains. Format: dc=cisco,dc=com

After upgrading from CPS 13.x.x or CPS 14.x.x to CPS 18.0.0 release, LDAP SSSD configuration is installed on default VM (pcrfclient/lb) and not on all VMs. You need to configure LDAP SSSD on all the other VMs.

Once LDAP SSSD configuration is complete, you need to authenticate the LDAP certificate. For more information, refer to *LDAP SSSD Configuration* section in *CPS Installation Guide for VMware*.

If upgrading from a lower version such as CPS 13.x.x to CPS 18.x.x and do not want the LDAP SSSD, modify the LDAP parameters as follows in YAML file:

```
ldapOnAll=false
ldapEnabled=false
```

After the modification, run `import_deploy.sh` so that LDAP SSSD is not installed by default

For more information on LDAP SSSD certificate authentication and troubleshooting, refer to *LDAP SSSD Configuration* section in *CPS Installation Guide for VMware*.

Sample YAML Configuration File - HA Setup

Use the following file as a template to create the YAML configuration file for your CPS deployment. Refer to Configuration Parameters - HA System, on page 6 for a description of the available parameters.

Note RADIUS-based policy control is no longer supported in CPS 14.0.0 and later releases as 3GPP Gx Diameter interface has become the industry-standard policy control interface.

```
---
#
#  CPS system configuration
#
#  CPS configuration is a YAML file with all the configuration required
#  to bring up a new installation of CPS.
#
#  This example file lists all possible configuration fields.
#  Fields that are not marked as required can be left out of
#  the configuration. Fields that are not provided will use
#  the default value. If not default is indicated the default
#  is an empty string.

# The version of the configuration file. The installation documentation
# for the version of the CPS you are installing will indicate which
# configuration version you must use.
# REQUIRED
```

```
configVersion: 1.0

# Configuration section for CPS hosts
# REQUIRED
hosts:
  # The host section must specify all hosts that are members of the CPS
  # deployment. Host entries consist of the following REQUIRED fields
  #  name: the string to be used as a hostname for the VM
  #  alias: the string to be used in hostname lookup for the VM
  #  interfaces: Network details consisting of the following REQUIRED fields
  #    network: The network name which must match a VLAN name (see below)
  #    ipAddress: The interface address
  - name: "lb01"
    alias: "lb01"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.201"
      - network: "Management"
        ipAddress: "172.18.11.154"
      - network: "Gx"
        ipAddress: "192.168.2.201"
  - name: "lb02"
    alias: "lb02"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.202"
      - network: "Management"
        ipAddress: "172.18.11.155"
      - network: "Gx"
        ipAddress: "192.168.2.202"
  - name: "sessionmgr01"
    alias: "sessionmgr01"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.22"
      - network: "Management"
        ipAddress: "172.18.11.157"
  - name: "sessionmgr02"
    alias: "sessionmgr02"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.23"
      - network: "Management"
        ipAddress: "172.18.11.158"
  - name: "qns01"
    alias: "qns01"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.24"
  - name: "qns02"
    alias: "qns02"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.25"
  - name: "qns03"
    alias: "qns03"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.26"
  - name: "qns04"
    alias: "qns04"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.27"
  - name: "pcrfclient01"
    alias: "pcrfclient01"
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.20"
      - network: "Management"
        ipAddress: "172.18.11.152"
  - name: "pcrfclient02"
    alias: "pcrfclient02"
```

```
    interfaces:
      - network: "Internal"
        ipAddress: "172.16.2.21"
      - network: "Management"
        ipAddress: "172.18.11.153"

# Configuration section for CPS VLANs
# REQUIRED
vlans:
  # VLAN entries consist of the following REQUIRED fields
  #  name: The VLAN name. This name must be used in the "network" field
  #        host interfaces (see above)
  #  vipAlias: Hostname associated with the vip
  #  vip: Virtual IP used no this network, if any.
  #  guestNic: The name of the interface specified in the host cloud config
  #            or the Heat definition.
  #
  - name: "Internal"
    vipAlias: "lbvip02"
    vip: "172.16.2.200"
  - name: "Management"
    vipAlias: "lbvip01"
    vip: "172.18.11.156"
  - name: "Gx"
    vipAlias: "gxvip"
    vip: "192.168.2.200"

# Configuration section for hosts not configured in the hosts section above.
# REQUIRED
additionalHosts:
  # additionalHosts entries consist of the following REQUIRED fields
  #  name: The hostname
  #  alias: The string to be used in the etc/host file.
  #  ipAddress: The IP address to use in the etc/host file.
  #
  # the "arbitervip" to the pcrfclient01 internal ip is mandatory.
  #
  - name: "lbvip01"
    ipAddress: "172.18.11.156"
    alias: "lbvip01"
  - name: "lbvip02"
    ipAddress: "172.16.2.200"
    alias: "lbvip02"
  - name: "diam-int1-vip"
    ipAddress: "192.168.2.200"
    alias: "gxvip"
  - name: "arbitervip"
    ipAddress: "172.16.2.20"
    alias: "arbitervip"

# Configuration section for general configuration items.
# REQUIRED
config:
  # Do not change. See install documentation for details.
  # default: sys_user_0
  qpsUser: "sys_user_0"

  # Do not change. See install documentation for details.
  # default: disabled
  selinuxState: "disabled"

  # Do not change. See install documentation for details.
  # default: targeted
  selinuxType: "targeted"

  # See install documentation for details.
  # default: broadhop
  broadhopVar: "broadhop"

  # Set true to enable TACACS+ authentication.
  # default: FALSE
  tacacsEnabled: "FALSE"
```

```
# The IP Address of the TACACS+ server
tacacsServer: "127.0.0.1"

# The password/secret of the TACACS+ server.
tacacsSecret: "CPE1704TKS"

# A set of SNMP Network Management Stations.
# NMS can be specified as IP addresses or IP
# addresses. Entries are space separated.
# Hostnames must also be specified in Additional
# Host configuration.
# See install documentation for details.
nmsManagers:

# Low Memory alert threshold %.
# default: 0.1 (10% free)
freeMemPer: "0.1"

# A space separated set of protocol:hostname:port
# entries. UDP is the only supported protocol.
# Example:
# upd:corporate_syslog_ip:514 udp:corporate_syslog_ip2:514
syslogManagers:

# A comma separated set of port values.
# This must match values in the syslog_managers_list.
# default: 514
syslogManagersPorts: "514"

# Port value for the rsyslog proxy server to listen
# for incoming connections
# default: 6515
logbackSyslogDaemonPort: "6515"

# IP address value used in the
# /etc/broadhop/controlcenter/logback.xml
# on the pcrfclient.
# default: lbvip02
logbackSyslogDaemonAddr: "lbvip02"


# High CPU alert threshold.
# The system will alert whenever the usage is
# higher than this value.
# default: 80
cpuUsageAlertThreshold: "80"

# Clear High CPU Trap threshold.
# The system will generate a clear trap when a
# High CPU trap has been generated and the CPU
# usage is lower than this value.
# default: 40
cpuUsageClearThreshold: "40"

# The number of 5 sec intervals to wait between
# checking the CPU usage.
# default: 12 (60 seconds)
cpuUsageTrapIntervalCycle: "12"

# The SNMP trap community string.
snmpTrapCommunity: "broadhop"

#The SNMP read community string.
snmpRoCommunity: "broadhop"

#
monQnsLb:

# The memory alert threshold (0.1 is 10%)
freeMemoryPerAlert: "0.1"

# The memory clear threshold (0.3 is 30%)
freeMemoryPerClear: "0.3"
```

```
  #
  monitorReplicaTimeout: "540"

  # Enable SCTP
  # TRUE - feature enabled
  # FALSE - feature disabled
  sctpEnabled: "TRUE"


  # Enables or disables linux firewall on all VMs (IPtables).
  # default: disabled
  firewallState: "disabled"

  # enable SNMP V3.
  # If null, SNMP V3 is disabled.
  # To enabled add the following:
  #   v3User: The SNMP V3 user: REQUIRED
  #   engineId: hex value (ie, 0x0102030405060708):  REQUIRED
  #   authProto: SHA or MD5: REQUIRED
  #   authPass: at least 8 characters:  REQUIRED
  #   privProto: AES or DES: REQUIRED
  #   privPass: OPTIONAL
  snmpv3:
     null
  #  v3User: "cisco_snmpv3"
  #  engineId: "0x0102030405060708"
  #  authProto: "SHA"
  #  authPass: "cisco_12345"
  #  privProto: "AES"
  #  privPass: ""


  # Users
  # There are different categories of users specified for the CPS.
  # All users have the following fields:
  #
  #  name: The user name. REQUIRED
  #  password: The password for the user. REQUIRED
  #            The password will need to be either in cleartext or
  #            encrypted. Please refer to Install documentation for details.
  #  groups: The groups for the user. Groups are specified as a list
  #          of group names.

  # System Users
  # Note that there must be a system use named sys_user_0
  sysUsers:
    - name: "qns"
      password: "$6$HtEnOu7S$8kkHDFJtAZtJXnhRPrPFI8KAlHFch41OJ405OnCCqO0CFuRmexvCRTk"
      groups:
        - pwauth

    - name: "qns-svn"
      password: "$6$HtEnOu7S$8kkHDFJtAZtJXnhRPrPFI8KAlHFch41OJ405OnCCqO0CFuRmexvCRTk"
    - name: "qns-ro"
      password: "$6$HtEnOu7S$8kkHDFJtAZtJXnhRPrPFI8KAlHFch41OJ405OnCCqO0CFuRmexvCRTk"

  # Hypervisor Users
  hvUsers:
    - name: "root"
      password: "cisco123"

  # Other Users for the CPS
  # e.g. Control Center Users
  additionalUsers:
    - name: "admin"
      password: "qns123"
      groups:
        - qns

# Configuration section for feature licenses
# REQUIRED
licenses:
```

```
  # Licenses have the following required fields:
  # feature: The name of the feature license.
  # license: The license key for the feature.
  # - feature: "feature 1  Name"
  #   license: "license 1 key string"
   - feature: "MOBILE_CORE"
     license: "xxxxxxx"
   - feature: "RADIUS_AUTH"
     license: "xxxxxxx"

# Configuration section for mongo replica sets
# REQUIRED
replicaSets:
  #
  # Mongo replica sets have the following REQUIRED fields
  # <Mongo Set Identifier> : The database for which the replica
  #                          set is being created.
  #   setName: The name of the replica set
  #   oplogSize: Mongo Oplog size
  #   arbiter: The Arbiter hosthame and port
  #   arbiterDataPath: The data directory on the arbiter VM
  #   members: List of members for the replica set. Each list element
  #            will be a session manager hostname:port
  #   dataPath: The data directory path on the session manager VMs
  - title: SESSION-SET1
    setName: set01
    oplogSize: 5120
    arbiter: pcrfclient01:27717
    arbiterDataPath: /var/data/sessions.1
    members:
      - sessionmgr01:27717
      - sessionmgr02:27717
    dataPath: /var/data/sessions.1/1
- title: SESSION-SET2
    setName: set08
    oplogSize: 5120
    arbiter: pcrfclient01:37717
    arbiterDataPath: /var/data/sessions.1/2
        members:
      - sessionmgr01:37717
      - sessionmgr02:37717
    dataPath: /var/data/sessions.1/2
    seeds: "sessionmgr01:sessionmgr02:37717"
  - title: BALANCE-SET1
    setName: set02
    oplogSize: 5120
    arbiter: pcrfclient01:27718
    arbiterDataPath: /var/data/sessions.2
    members:
      - sessionmgr01:27718
      - sessionmgr02:27718
    dataPath: /var/data/sessions.2
  - title: REPORTING-SET1
    setName: set03
    oplogSize: 5120
    arbiter: pcrfclient01:27719
    arbiterDataPath: /var/data/sessions.3
    members:
      - sessionmgr01:27719
      - sessionmgr02:27719
    dataPath: /var/data/sessions.3
  - title: SPR-SET1
    setName: set04
    oplogSize: 3072
    arbiter: pcrfclient01:27720
    arbiterDataPath: /var/data/sessions.4
    members:
      - sessionmgr01:27720
      - sessionmgr02:27720
    dataPath: /var/data/sessions.4
  - title: AUDIT-SET1
    setName: set05
    oplogSize: 3072
```

```
    arbiter: pcrfclient01:27725
    arbiterDataPath: /var/data/sessions.5
    members:
      - sessionmgr01:27725
      - sessionmgr02:27725
    dataPath: /var/data/sessions.5
  - title: ADMIN-SET1
    setName: set06
    oplogSize: 3072
    arbiter: pcrfclient01:27721
    arbiterDataPath: /var/data/sessions.6
    members:
      - sessionmgr01:27721
      - sessionmgr02:27721
    dataPath: /var/data/sessions.6
  - title: ADMIN-SET2
    setName: set07
    oplogSize: 3072
    arbiter: pcrfclient01:27731
    arbiterDataPath: /var/data/sessions.7
    members:
      - sessionmgr01:27731
      - sessionmgr02:27731
    dataPath: /var/data/sessions.7

# Configuration section for LDAP/SSSD
   ldapEnabled: “true”
   ldapOnAll:true
   ldapServer: "ldaps://<serverip>:10648"
   ldapSearchBase: "ou=users,dc=cisco,dc=com"
   ldapDefaultBindDn: "uid=admin,ou=system"
   ldapSecret: “secret”
   ldapDefaultUser: “admin”
   ldapOuUser: “users”
   ldapOuGroup: “groups”
   ldapDefaultGroup: “Admin”
   ldapDefaultGroupEditor: “User”
   ldapDcName: "dc=cisco,dc=com"
```

MongoDB Authentication Process

- Change mongo user password (Application downtime is involved):
  - Modify password using config PATCH API.
  - Wait for the process to complete.
  - Execute change password script (`/var/qps/install/current/scripts/modules/mongo_change_password.py`) and enter the old password.

    Syntax:

    `/var/qps/install/current/scripts/modules/mongo_change_password.py <old password>`
  - Restart all the JAVA processes.
- Disable mongo authentication (No application downtime is involved):
  - Modify mongo authentication configuration using config PATCH API.
  - Wait for the process to complete.
  - Execute disable mongo authentication script: `/var/qps/install/current/scripts/modules/mongo_auth_upgrade.py`

- Restart all the JAVA processes.

- Enable mongo authentication (Mongo and application downtime is involved).
  - Modify mongo authentication configuration using config PATCH API.
  - Wait for the process to complete.
  - Execute enable mongo authentication script:
    `/var/qps/install/current/scripts/modules/mongo_auth_upgrade.py`
  - Restart all the JAVA processes.

/api/system/config/status

Purpose

This API retrieves the status of individual install and deploy tasks run when a new or updated configuration is applied on the Cluster Manager VM.

This API can be called while the installation and deployment tasks are actively running.

The status reports:

- timestamp: timestamp in milliseconds.
- taskname: name of the individual task.
- status:
  - START: start of task.
  - INFO: general information about the task.
  - WARNING: error information about the task.
  - SUCCESS: task was successfully completed.
  - FAILURE: task failed and deployment failed.
- details: information about this task.

Retrieve Deployment Status

To retrieve the deployment status:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** GET

- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success.

  Example Response:

```
---
[
{"timestamp":"1454367943000","taskName":"CPS
Installation","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"Cluman Setup","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"Cluman Setup","status":"SUCCESS","details":"Wait
 for Puppet to complete"},
{"timestamp":"1454367943000","taskName":"Post Install","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"SyncSvn","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"SyncSvn","status":"WARNING","details":"Failed
 to sync SVN."},
{"timestamp":"1454367943000","taskName":"SyncSvn","status":"SUCCESS","details":""},
{"timestamp":"1454367943000","taskName":"build_set","status":"START","details":"Building
 replica sets"},
{"timestamp":"1454367943000","taskName":"build_set","status":"INFO","details":"Wrote
mongo config"},
{"timestamp":"1454367943000","taskName":"build_set","status":"INFO","details":"Syncing
 mongo config to other hosts"},
{"timestamp":"1454367943000","taskName":"build_set","status":"SUCCESS","details":"Replica
 sets have been created successfully"},
{"timestamp":"1454367943000","taskName":"SetPriority","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"SetPriority","status":"SUCCESS","details":""},
{"timestamp":"1454367943000","taskName":"AddAdditionalUsers","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"AddAdditionalUsers","status":"SUCCESS","details":""},
{"timestamp":"1454367943000","taskName":"Licenses","status":"START","details":""},
{"timestamp":"1454367943000","taskName":"Licenses","status":"SUCCESS","details":""},
{"timestamp":"1454367943000","taskName":"Post Install","status":"SUCCESS","details":""}
]
```

The deployment process is complete when the following response is received: `"Post Install","status":"SUCCESS"`

Note The amount of time needed to complete the entire deployment process depends on the number of VMs being deployed, as well as the hardware on which it is being deployed. A typical deployment can take 45 minutes or more.

Startup status logs are written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log`

Refer to the /api/system/status/cps, on page 24 to determine the readiness status of the CPS cluster.

/api/system/status/cps

Purpose

This API returns the readiness status of CPS cluster.

Cluster Readiness

This API returns the "readiness" status of the CPS cluster.

The cluster is deemed "ready" when Puppet has run to completion on all VMs and the Replica set creation is complete on the Session Manager VMs. The Orchestrator can use this API to check when the cluster is ready so that it can then invoke the Service Creation APIs.

This API reports an aggregate status of MongoDB replica sets, qns processes, and the cluster (Puppet) for all VMs.

This API will timeout after 150 seconds.

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** GET
- **Payload:** JSON
- **Response:**

  The following example shows the readiness status for a CPS cluster:

  ```
  {
      "clusterStatus": "ready",
      "mongoStatus": "ready",
      "qnsStatus": "ready"
  }
  ```

  mongoStatus and clusterStatus can report "ready", "not ready", or "error". qnsStatus can report "ready" or "not ready". If mongoStatus reports an "error" status, the clusterStatus also will report an "error" status.

  If the any database replica-sets are reporting "ok", but members are "off-line", mongoStatus will report "not ready".

  If any of the replica-sets are down or in an error state, mongoStatus will report "error".

- **Error Codes:**
  - 200 OK: success
  - 404: Unknown entity
  - 500: Script config not found
  - 500: CPS status job interrupted
  - 500: CPS status job timeout
  - 500: CPS status job termination interrupted
  - 500: Failed retrieval of CPS status job results

API logs are at written to: `/var/log/orchestration-api-server.log`

/api/system

Purpose

This API is to used to determine the current state of the CPS system, and if necessary, override it in the event the reported state does not match the actual system state.

Many CPS orchestration APIs are accepted only when the CPS system is in a particular state. This API provides a method of overriding the reported API system state. It does not rectify or correct the underlying issue. For example setting the state to pre_deploy does not un-deploy the CPS deployment.

API logs are at written to: `/var/log/orchestration-api-server.log`

Retrieve the Current API State

To determine the current system state:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** GET
- **Payload:** There is no payload.
- **Response Codes:** 200: OK.

  Example Response:

  ```
  {
    "state": "pre_config"
  }
  ```

  This API can be used at any time.

The following states can be reported:

- **pre_config:** no configuration has been loaded onto the system (/api/system/config).
- **pre_deploy:** a configuration has been loaded, but not applied (api/system/config/action/apply).
- **deploying:** the system is in the process of being deployed.
- **deployed:** the system has finished the installation/deployment.
- **upgrading:** the system is in the process of being upgraded.
- **busy:** the system is currently processing an operation.

Override the Current API State

Caution This API should only be used as directed by a Cisco representative. Improper use can cause irreparable harm to the CPS deployment.

To override the current system state:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** POST
- **Payload:** JSON payload with the new state specified as one of the following options: `pre_config`, `pre_deploy`, `deploying`, `deployed`, or `upgrading`.

  For example:

```
{
  "state": "pre_config"
}
```

- **Response Codes:** 400: Invalid state, please use: [pre_config, pre_deploy, deploying, deployed, upgrading]; 500: System error. See logs.

  Example Response:

```
{
  "state": "pre_config"
}
```

Upgrade APIs

Caution The Upgrade API can trigger kernel upgrade if kernel version is updated in new CPS version. So all the necessary precautions prior to kernel upgrade of CPS VMs must be taken before an upgrade is triggered through orchestration API. If kernel is upgraded then VMs will be rebooted.

Upgrade API Prerequisites

The following sequence of commands should be executed in OpenStack before running the CPS upgrade APIs.

Note These commands are for illustration purpose only and do not override any setup specific constraints. The specific commands may differ on your environment.

Step 1 Create a glance image of the new CPS ISO.

```
glance image-create --name <name of CPS ISO> --disk-format iso --container-format bare --is-public
True --file <Absolute path to new CPS ISO>
```

Step 2 Create a cinder volume based on the glance image of the new CPS ISO.

```
cinder create --image-id <glance image id of new CPS ISO> --display-name <name of new CPS ISO volume>
--availability-zone <optional zone> <size of ISO in GBs>
```

Step 3 Detach the existing CPS ISO volume from the Cluster Manager VM.

```
nova volume-detach <nova instance ID of cluman> <cinder volume ID of old CPS ISO volume>
```

Step 4 Attach the new CPS ISO volume to the Cluster Manager VM. This will require either the name of device at which volume is attached to the Cluster Manager, or "auto" to attach the volume as any available device name. In either case, the following command will output name of device to which new CPS ISO volume is attached.

```
nova volume-attach <nova instance ID of cluman> <cinder volume ID of new CPS ISO volume> <Name of
device, e.g. /dev/vdb or auto for autoassign>
```

/api/system/upgrade

Purpose

The following APIs are used to mount and unmount an ISO image to the Cluster Manager VM, trigger an out-of-service upgrade of a CPS deployment, and view the status of the upgrade.

Note Before invoking any of these APIs, refer to Upgrade API Prerequisites, on page 27.

Logs are at written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

Unmount ISO

To unmount an existing CPS ISO image from `/mnt/iso` directory on the Cluster Manager:

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/upgrade/action/unmount

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json

- **Method:** POST
- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success; 400: The mount parameters are invalid; 500: System Error. See logs.

Note After invoking this API, it is recommended to detach the ISO image from the Cluster Manager VM using relevant command in OpenStack.

Mount ISO

Note Before invoking this API:

- A new cinder volume must be created in OpenStack based on the CPS ISO, and then attached to the Cluster Manager VM using relevant command in OpenStack. Refer to Upgrade API Prerequisites, on page 27 for more details.
- Run the **lsblk** command on the Cluster Manager VM to check the device name before running mount API. This needs to be checked after the CPS ISO volume has been attached to the Cluster Manager VM.

To mount the CPS ISO image onto /mnt/iso directory on the Cluster Manager:

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/upgrade/action/mount

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** POST
- **Payload:**

```
{
"deviceName": "<filename of the block device at which the cinder volume is attached"
Ex: "/dev/vdb>"
}
```

`/dev/vdb` is for illustration only. Replace with the device name to which the CPS ISO volume is attached on your Cluster Manager VM.

Example:

```
{
 "deviceName": "/dev/vdb"
}
```

- **Response Codes:** 200 OK: success; 400: The mount parameters are invalid; 500: System Error. See logs.

Upgrade CPS

Caution This API must only be used during a planned maintenance window. This API does not perform an in-service software upgrade. CPS processes will be restarted during this process and traffic will be affected.

This API can only be used once the CPS has been deployed and is in a ready state. Prior to that time this API will not be available.

To upgrade CPS using the mounted ISO:

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/upgrade/action/apply

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** POST
- **Payload:**

  type: Only `"OUT_OF_SERVICE"` is supported.

  config: The SVN/policy repository configuration to back up prior to upgrade. This repository will be backed up and restored during the upgrade.

  installType: The type of CPS deployment. Only `mobile` is supported.

  Example:

```
{
"config": "run",
"installType": "mobile",
"type": "OUT_OF_SERVICE"
}
```

- **Response Codes:** 200 OK: success; 400: The input parameters are malformed or invalid.

The upgrade logs are at written to: /var/log/install_console_<*date_time*>.log on the Cluster Manager VM.

Upgrade Status

To view the status of an upgrade:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** GET
- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success; 500: Script config not found

Example Response:

```
{
        "status": "In-Progress"
    }
```

- Not-Started - No upgrade has been initiated
- In-Progress - Upgrade is currently underway
- Completed - Upgrade has completed
- Error - There is a problem with the upgrade

This API is only valid after the operator has issued an upgrade.

System Configuration APIs

/api/system/mongo/config

Purpose

This API is used to retrieve the contents of `/etc/broadhop/mongoConfig.cfg`. This API is also used to add members to existing Mongo replica sets.

Important This API does **not** support modifications to any other parameters within the Mongo configuration. It will only add members to existing Mongo replica sets.

Important While choosing mongo ports for replica-sets, consider the following:

- Port is not in use by any other application. To check it, login to VM on which replica-set is to be created and execute the following command:

  `netstat -lnp | grep <port_no>`

  If no process is using same port then port can be chosen for replica-set for binding.

- Port number used should be greater than 1024 and not in ephemeral port range i.e, not in between following range :

  net.ipv4.ip_local_port_range = 32768 to 61000

API logs are at written to: `/var/log/orchestration-api-server.log`

Workflow

1. Retrieve Current Mongo Configuration, on page 32
2. Manually edit the YAML file retrieved in step 1 to add members to the existing replica sets.
3. Load Updated Configuration, on page 33

4 Apply Loaded Configuration, on page 34

Retrieve Current Mongo Configuration

To retrieve (GET) the current configuration:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

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

```
---
- title: "SESSION-SET1"
  setName: "set01"
  oplogSize: "5120"
  arbiter: "pcrfclient01:27717"
  arbiterDataPath: "/var/data/sessions.1"
  members:
  - "sessionmgr01:27717"
  - "sessionmgr02:27717"
  dataPath: "/var/data/sessions.1"
- title: "BALANCE-SET1"
  setName: "set02"
  oplogSize: "5120"
  arbiter: "pcrfclient01:27718"
  arbiterDataPath: "/var/data/sessions.2"
  members:
  - "sessionmgr01:27718"
  - "sessionmgr02:27718"
  dataPath: "/var/data/sessions.2"
- ...
```

  - Example Response (YAML format): GR Setup

```
- title: "SESSION-SET1"
  setName: "set01"
  oplogSize: "1024"
  arbiter: "arbiter-site3:27717"
  arbiterDataPath: "/var/data/sessions.1"
  siteId: "SITE1"
  members:
    - sessionmgr02-site1:27717
    - sessionmgr01-site1:27717
  dataPath: /var/data/sessions.1/set1
  primaryMembersTag: "SITE1"
  secondaryMembersTag: "SITE2"
  shardCount: "4"
  hotStandBy: "false"
  seeds: "sessionmgr01:sessionmgr02:27717"
```

OR

```
- title: "SESSION-SET1"
  setName: "set01"
  oplogSize: "1024"
```

```
arbiter: "arbiter-site3:27717"
arbiterDataPath: "/var/data/sessions.1"
primaryMembers:
  - "sessionmgr02-site1:27717"
  - "sessionmgr01-site1:27717"
secondaryMembers:
  - "sessionmgr02-site2:27717"
  - "sessionmgr01-site2:27717"
dataPath: "/var/data/sessions.1"
hotStandBy: "false"
shardCount: "4"
seeds: "sessionmgr01:sessionmgr02:27717"
primaryMembersTag: "SITE1"
secondaryMembersTag: "SITE2"
siteId: "SITE1"
```

Note The response will include the complete Mongo configuration in YAML format.

Load Updated Configuration

Note This API can only be used once CPS has been deployed and is in a ready state. Prior to that time this API will not be available.

Use this API to load an updated Mongo configuration on the CPS Cluster Manager:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

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

  Example Response:

  The updated contents of `/etc/broadhop/mongoConfig.cfg` is returned in the response in YAML format.

  Note After using this API to load the updated mongo configuration, you must apply the configuration. Refer to Apply Loaded Configuration, on page 34.

Apply Loaded Configuration

Note This API can only be used once the CPS has been deployed and is in a ready state. Prior to that time this API will not be available.

Use this API to apply the updated Mongo configuration that you loaded using Load Updated Configuration, on page 33:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/json
- **Method:** POST
- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error

  This API returns immediately and does not wait for the members to be added. Refer to the log file to check the status.

  Example Response:

```
{
    "logfile": "/var/log/broadhop/scripts/orch_api_03122016_203220.log"
}
```

/api/system/config/hosts

Purpose

This API is used to retrieve the current list of deployed CPS hosts, and to add or remove Policy Server (QNS), SessionMgr, and Policy Director (Load Balancer) hosts to the CPS cluster. This enables an orchestrator to increase (scale up) or decrease (scale down) the session processing capacity of the CPS cluster.

Important To scale up, you must create VMs using heat or nova boot commands. However, already existing stacks cannot be used to scale up using heat.

Note Only Policy Server (QNS) and SessionMgr hosts can be scaled down. Policy Director (Load Balancer) hosts cannot be scaled down.

Retrieve Current List of Deployed Hosts

To retrieve (GET) the current list of hosts deployed in the CPS cluster:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **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):

```
---
- name: "lb01"
  alias: "lb01"
  interfaces:
  - network: "Internal"
    ipAddress: "172.16.2.201"
  - network: "Management"
    ipAddress: "172.18.11.154"
  - network: "Gx"
    ipAddress: "192.168.2.201"
- ...
```

Note The example response shown above is abbreviated. The response includes the complete list of configured hosts.

Add New Policy Server (QNS), Session Manager, and Policy Director (Load Balancer) Hosts

This API adds additional Policy Server (QNS), SessionMgr, and/or Policy Director (Load Balancer) hosts to an existing deployment. The API uses the PATCH method, which adds new hosts without affecting the existing configured hosts.

Policy Server (QNS), SessionMgr, and/or Policy Director (Load Balancer) VMs must be added in pairs (for example, qns05, qns06 and sessionmgr03, sessionmgr04). Attempts to add odd numbers of VMs are rejected.

Before issuing this API, you must create the additional VMs using Heat or Nova boot commands. For example, to create two additional Policy Server VMs (qns05, qns06):

```
nova boot --config-drive true --user-data=qns05-cloud.cfg --image "base_vm" --flavor "qps"
 --nic
net-id="2544e49e-0fda-4437-b558-f834e73801bb,v4-fixed-ip=172.16.2.28" --availability-zone
"az-2:os8-compute-2.cisco.com" "qns05"

nova boot --config-drive true --user-data=qns06-cloud.cfg --image "base_vm" --flavor "qps"
 --nic
net-id="2544e49e-0fda-4437-b558-f834e73801bb,v4-fixed-ip=172.16.2.29" --availability-zone
"az-2:os8-compute-2.cisco.com" "qns06"
```

Note To add SessionMgr VMs, refer to /api/system/config/replica-sets, on page 38 to configure additional replica sets on newly deployed Session Mgr VMs.

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.

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:** Include the YAML configuration file in the PATCH request. Use the `op: add` parameter to add a host. Only the new hosts should be defined in the YAML configuration file submitted in the API request.

  Sample Payload:

```
---
- op: add
  name: “qns05"
  alias: “qns05"
  interfaces:
    - network: "Internal"
      ipAddress: "172.16.2.28"
- op: add
  name: “qns06"
  alias: “qns06"
  interfaces:
    - network: "Internal"
      ipAddress: "172.16.2.29"
```

- **Response Codes:** 200 OK: success; 400: Invalid data; 500: System error

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/hosts

After issuing this API, /api/system, on page 26 will report a "busy" state. Once the operation is complete, it will report a "deployed" state.

Additionally, the /api/system/config/status , on page 23 can be used to monitor the progress of individual steps of the operation.

Status logs are also written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

In case of any errors, check the API log file `/var/log/orchestration-api-server.log` and do the following:

- Verify if puppet on the new Policy Director (Load Balancer) VM is completed successfully.
- In case of diameter calls issue, verify if puppet on lb01/02 VMs is completed successfully and haproxy-diameter configuration is updated. Also, verify if Policy Builder configuration for the new LB VMs is properly updated.
- Verify if `diagnistics.sh` status is clean after Policy Builder update.

Remove Policy Server (QNS) and Session Manager Hosts

This API removes Policy Server (QNS) and/or SessionMgr hosts from an existing deployment.

Note Only Policy Server (QNS) and SessionMgr hosts can be removed from an existing deployment. Policy Director (Load Balancer) hosts cannot be removed.

Caution Before removing any SessionMgr hosts, you must remove the replica-sets configured on those hosts using the /api/system/config/replica-sets, on page 38

Policy Server (QNS) VMs and SessionMgr VMs must be removed in pairs (for example qns05, qns06 and sessionmgr03, sessionmgr04). Attempts to remove odd numbers of VMs are rejected.

This API removes the specified VMs from the Cluster Manager configuration only. After issuing this API, the orchestrator terminates the VMs in OpenStack.

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:** Include the YAML configuration file in the PATCH request. Use the `op: remove` parameter to remove a host. Only the hosts which are to be removed should be defined in the YAML configuration file submitted in the API request.

  Important Using `op: remove` parameter, only the hosts configuration is removed and not the actual VMs. You need to use nova commands to remove the VMs. For more information on nova commands, refer to OpenStack commands.

  Sample Payload:

```
---
- op: remove
  name: “qns05"
  alias: “qns05"
 - op: remove
  name: “qns06"
  alias: “qns06"
```

After issuing this API, /api/system, on page 26 will report a "busy" state. Once the operation is complete, it will report a "deployed" state.

Additionally, the /api/system/config/status , on page 23 can be used to monitor the progress of individual steps of the operation.

Status logs are also written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

Configuration Parameters - Hosts

The following parameters can be defined in the Hosts YAML configuration file:

Table 4: Configuration Parameters - Hosts

Parameter	Description
`- op:`	The operation to be performed for this host, either `add` or `remove`.
`name:`	Defines the hostname of the VM. This name must be resolvable in the enterprise DNS environment.
`alias:`	Defines the internal host name used by each CPS VMs for internal communication, such as sessionmgr03 or qns05.
`interfaces:`	This section defines the network interface details for the VM.
`- network:`	Defines the CPS VLAN network name for the VM. QNS VMs are typically assigned to the "Internal" VLAN, and SessionMgrs are typically assigned both to "Internal" and "Management" VLANs.
`ipAddress:`	Defines the IP address of the VM.

/api/system/config/replica-sets

Purpose

This API is used to retrieve the current list of replica-sets for the Session database, to add additional replica-sets, or remove replica-sets.

Retrieve Current Replica-sets

To retrieve (GET) the current list of replica-sets configured for the Session database:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **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 Payload (YAML format): HA Setup

```
---
- title: "SESSION-SET1"
  setName: "set01"
  oplogSize: "5120"
  arbiter: "pcrfclient01:27717"
  arbiterDataPath: "/var/data/sessions.1"
  members:
  - "sessionmgr01:27717"
  - "sessionmgr02:27717"
  dataPath: "/var/data/sessions.1"
 - ...
```

Example Payload (YAML format): GR Setup

```
---
 - title: SESSION-SET1
    setName: set01
    oplogSize: 1024
    arbiter: arbiter-site3:27717
    arbiterDataPath: /var/data/sessions.1
    siteId: "SITE1"
    members:
      - sessionmgr02-site1:27717
      - sessionmgr01-site1:27717
    dataPath: /var/data/sessions.1/set1
    primaryMembersTag: "SITE1"
    secondaryMembersTag: "SITE2"
    shardCount: "4"
    hotStandBy: "false"
    seeds: "sessionmgr01:sessionmgr02:27717"
```

If the user has configured `primaryMembersTag:` and `secondaryMembersTag:` parameters, then only these parameters will be visible in case of API GET is called to fetch configuration details. There will be single tag specified for SPR/balance/session geo tagging. The value will be matched with any one of the parameters mentioned in `qns.conf` for geo site tagging.

Note The response will include the complete list of configured replica-sets.

Important While choosing mongo ports for replica-sets, consider the following:

- The port must not be in use by any other application. To check the port number, login to VM on which replica-set is to be created and execute the following command:

  ```
  netstat -lnp | grep <port_no>
  ```

  If no process is using the port, then the port number can be chosen for replica-set for binding.

- The port number used should be greater than 1024 and not be in the ephemeral port range i.e, not in between following range: 32768 to 61000.

- While configuring mongo ports in a GR environment, there should be a difference of 100 ports between two respective sites. For example, consider there are two sites: Site1 and Site2. For Site1, if the port number used is 27717, then you can configure 27817 as the port number for Site2. This is helpful to identify a mongo member's site. By looking at first three digits, one can decide where the mongo member belongs to. However, this is just a guideline. You should avoid having mongo ports of two different sites to close to each other (for exampl, 27717 on Site-1 and 27718 on Site2).

  Reason: The reason is that the `build_set.sh` script fails when you create shards on the site (for example, Site1). This is because the script calculates the highest port number in the `mongoConfig` on the site where you are creating shards. This creates clash between the replica-sets on both sites. Since the port number which it allocates might overlap with the port number of `mongoConfig` on other site (for example, Site2). This is the reason why there should be some gap in the port numbers allocated between both the sites.

Add Replica-sets

This API configures additional replica-sets on newly deployed SessionMgr VMs. This API uses the PATCH method, which adds replica-sets without affecting the existing configured replica-sets.

When this API call completes, the Cluster Manager configuration is updated and all new replica-sets are created asynchronously.

Note The amount of time needed complete the process depends on the number of replica-sets being deployed.

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

  

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH

- **Payload:** Include the YAML configuration file in the PATCH request. Use the `op: add` parameter to add a replica-set. Only the new replica-sets should be defined in the YAML configuration file submitted in the API request.

  Sample Payload (YAML format): HA Setup

```
---
- op: add
   title: SESSION
   arbiter: pcrfclient01
   instances: 2
   members:
    - sessionmgr03
    - sessionmgr04
```

  Sample Payload (YAML format): GR Setup

```
---
- op: add
   title: SESSION
   arbiter: pcrfclient01
   instances: 2
   members:
    - sessionmgr03
    - sessionmgr04
       -primaryMembersTag: “sitename”
       -secondaryMembersTag: “sitename”
```

- **Response Codes:** 200 OK: success; 400: Invalid data; 500: System error

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/replica-sets

The status of this API call is reported in http://:<*Cluster Manager IP*>8458/api/system/config/status

Status logs are also written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

Remove Replica-sets

This API removes replica-sets from deployed SessionMgr VMs. This API uses the PATCH method.

This API must be issued before removing any Session Manager VMs during a scale down of the CPS Cluster using the /api/system/config/hosts, on page 34 API.

After issuing this API, the /api/system/config/status , on page 23 API can be used to monitor the removal of the ring-sets and the replica-sets. After the operation has completed, this API will return a SUCCESS status for the operation.

While the operation is ongoing, performing a GET with the /api/system/config/, on page 2 API will return a BUSY status for the operation. No other API operations are allowed while the system is in this state.

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH

- **Payload:** Include the YAML configuration file in the PATCH request. Only the replica-sets which are to be removed should be defined in the YAML configuration file submitted in the API request.

  Sample Payload (YAML format):

```
---
- op: remove
   title: SESSION
   setName: set01
   arbiter: pcrfclient01
   instances: 2
   members:
    - sessionmgr03
    - sessionmgr04
```

- **Response Codes:** 200 OK: success; 400: Invalid data; 500: System error

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/replica-sets

The status of this API call is reported in http://:<*Cluster Manager IP*>8458/api/system/config/status

Status logs are also written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

Adding/Updating Shard Count

Use this API to create shards. This API also supports existing scaling session replica-set and adding shards to existing session replica-sets.

Shards must be created during installation after the qns restart process (post install step).

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:**

  Sample Payload (YAML format) for scaling new replica-set:

Note hotStandBy, shardCount and seeds are optional parameters.

```
---
- op: "add"
  title: "SESSION"
  instances: "1"
  arbiter: "pcrfclient01"
  members:
  - "sessionmgr01"
  - "sessionmgr02"
 hotStandBy: “true”
 shardCount: “4”
seeds: “sessionmgr01:sessionmgr02”
```

Sample Payload (YAML format) for modifying the replica-set configuration:

Note hotStandBy, shardCount and seeds are required parameters.

```
---
- op: "modify-shards"
  setName: "set10"
  hotStandBy: "true"
  shardCount: "5"
  seeds: "sessionmgr01:sessionmgr02:27820"
```

- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/replica-sets

Update Priority

Priorities can be set in descending order using PATCH request.

In HA environment, priorities can be set for all replica sets of a particular replica database like session, admin, and so on. Also, you can set a particular replica-set under specific replica database.

In GR environment, priorities can be set for particular site and all replica-sets of a particular replica database like session, SPR, and so on. Also, you can set a particular replica-set under specific replica database. siteId parameter is mandatory in GR scenario.

Note It is required that replica-set are created before priority can be set. During installation, priority is added for all replica sets. In case a member is added using `addMember` API. it is required to execute set-priority API to set priority for given replica-set.

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:** To change the priority of more than one database type you must include another block in the request.

  Example Payload (YAML format): HA Setup

```
---
- op: "set-priority"
  title: "SESSION"
  setName: “set01”

- op: "set-priority"
  title: "SPR"
```

  Example Payload (YAML format): GR Setup

```
---
 - op: "set-priority"
  title: "SESSION"
  siteId: "SITE1"
```

Note For HA, *title* parameter is mandatory. For GR, *title* and *siteId* are mandatory parameters. *setName* is optional parameter for both HA and GR deployments.

- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error

Configuration Parameters - Replica-set

The following parameters can be defined in the ReplicaSet YAML configuration file:

Table 5: Configuration Parameters - Replica-set

Parameter	Description
`- op:`	The operation to be performed for this replica set, either `add` or `remove`.
`title:`	The database for which the replica set is being created. The only option supported is SESSION.
`arbiter:`	The hostname of the arbiter, typically pcrfclient01.
`instances:`	The number of replica set instances to create. For each replica set, the API will automatically generate the next available port, for example 27737, 27757 and so on.
`members:`	The list of members for the replica set. Each list element will be a session manager hostname, for example sessionmgr03.
`- <member>`	List each member hostname on a separate line, for example: sessionmgr03 sessionmgr04 The port for each Session Manager is automatically generated by the API.
`siteId:`	This parameter can be either local or remote site.
`title:`	This parameter is used to represent replica-set of a particular type. For example, session, SPR, and so on.
`hotStandBy:`	This parameter is used to defined whether the created shard is to be used for primary or backup database. If set to true, then created shard will be used for backup database. If the parameter is not configured, then the created shard will be used for non backup database. By default, this parameter is not configured.

Parameter	Description
`shardCount:`	This parameter is used to defined the number of shards to be created. In modify request, shards can only be increased.
`seeds:`	This parameter is used to defined sharding for multiple sessionmgr VMs. Enter the sessionmgr VM name with port separated by a colon (:) with each pair separated by a colon (:). Example: sessionmgr01:sessionmgr02:27717, sessionmgr03:sessionmgr04:27717
`primaryMembersTag:`	This parameter is used to define the sitename for primary members of a replica set for geo tagging. This is an optional parameter.
`secondaryMembersTag:`	This parameter is used to define the sitename for secondary members of a replica set for geo tagging. This is an optional parameter.
`siteId:`	In GR setup, this parameter is used to define the replica-set corresponding to the given site.

Note The ReplicaSet API automatically generates values for the following parameters: `setname`, `oplogSize`, and `dataPath`. The default oplogSize is 5120 MB.

/api/system/config/replica-sets/action/sync-mongo

Purpose

This API is used to copy the `/etc/broadhop/mongoConfig.cfg` file from one site to another. API can be called on local cluman which in turn calls the remote cluman and update its data. The parameter `remoteClumanIp` needs to be configured using /api/system/config/config, on page 46. This is required before syncing operation can be started.

- **Endpoint and Resource:** http://<*Cluster Manager IP*>:8458/api/system/config/replica-sets/action/sync-mongo

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** POST

- **Payload:** There is no payload.
- **Response Codes:** 200 OK: success; 400: The request is invalid; 500: Server Error

/api/system/config/config

Purpose

This API is used to retrieve or update the 'config' section of the CPS cluster configuration.

API logs are at written to: `/var/log/orchestration-api-server.log`

Retrieve Current Configuration

To retrieve (GET) the 'config' section of the configuration currently loaded on the CPS cluster:

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

Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** There is no payload.
- **Response Codes:** 200: OK.

  Example Response (YAML format):

```
---
config:
  qpsUser: "sys_user_0"
  selinuxState: "disabled"
  selinuxType: "targeted"
  ...
  sysUsers:
     ...
  hvUsers:
     ...
  additionalUsers:
     ...
```

Note The example response shown above is abbreviated. The response will include the complete list of parameters from the 'config' section of the consolidated configuration.

Update Configuration

This API modifies the parameters within the 'config' section of the consolidated configuration on an existing deployment. This API uses the PATCH method, which enables you to modify specific parameters without needing to submit the entire configuration.

Note Only new sysUsers and additionalUsers can be added.

Modifying existing sysUsers and additionalUsers is not supported.

Adding new or modifying existing hvUsers is not supported.

When this API call completes, the Cluster Manager configuration is updated and the new configuration is then pushed to all CPS VMs.

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PATCH
- **Payload:** Include the YAML configuration file in the PATCH request. Only the modified parameters should be defined in the YAML file.

  For a list of parameters which can be defined in this file, refer to the parameters defined in the config, sysUsers, hvUsers, and additionalUsers sections listed in Configuration Parameters - HA System, on page 6.

  Sample Payload (YAML format):

```
---
  selinuxType: "permissive"
  firewallState: "enabled"
  selinuxState: "enabled"
  snmpv3:
    v3User: "cps-snmp"
    engineId: "4321"
    authPass: "snmp123"
    privPass: "snmp321"
  tacacsEnabled: "TRUE"
  firewallState: "enabled"
  additionalUsers:
    - name: orchuser
      password: cisco123
      groups:
       - qns
```

  Note `root` user group is not authorized group for Control Center.

  To add new TACACS configuration to an existing CPS deployment, use the PATCH method:

  Sample Payload (YAML format):

```
---
tacacsEnabled = "TRUE"
tacacsServer = "127.0.0.1"
tacacsSecret = "CPE1704TKS"
```

  Note The PATCH method will re-run puppet on all the VMs.

'config' section also supports the following "extra" TACACS parameters:

Sample Payload (YAML format):

```
tacacsService = "pcrflinuxlogin"
tacacsProtocol = "ssh"
tacacsTimeout = "5"
tacacsDebug = "0"
```

- **Response Codes:** 200 OK: success; 400: Invalid data; 500: System error

To verify the configuration was properly loaded, perform another GET to http://<*Cluster Manager IP*>:8458/api/system/config/config

The status of this API call is reported in http://:<*Cluster Manager IP*>8458/api/system/config/status

Status logs are also written to: `/var/log/startupStatus.log` on the Cluster Manager VM.

API logs are written to: `/var/log/orchestration-api-server.log` on the Cluster Manager VM.

/api/system/config/additional-hosts

Purpose

This API enables you to configure new peer nodes such as PCEF, NTP, NMS, and so on, by modifying the `/etc/hosts` files on all CPS VMs.

The API logs are written in the `/var/log/orchestration-api-server.log` and `/var/log/startupStatus.log` files.

Note This API does not add a CPS VM to the CPS cluster.

Retrieve AdditionalHosts Configuration

To retrieve (GET) the AdditionalHosts configuration from the CPS Cluster Manager VM:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** GET
- **Payload:** There is no payload
- **Response Codes:** 200 OK: success

  Example Response (YAML format):

```
---
 - name: "Host1 name"
   alias: "Host1 internal name"
   ipAddress: "Host1 IP address"
```

```
- name: "Host2 name"
  alias: "Host2 internal name"
  ipAddress: "Host2 IP address"
- name: "Host3 name"
  alias: "Host3 internal name"
  ipAddress: "Host3 IP address"
```

Add or Update AdditionalHosts Entry

This API adds or updates a new AdditionalHosts entry in the configuration file.

When this API call completes, the Cluster Manager is configured with the new `/etc/hosts` file. All the other deployed VMs are then updated asynchronously and the status is reported in http://:<*Cluster Manager IP*>8458/api/system/config/status.

To add or update an AdditionalHosts configuration:

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

  Note If HTTPS is enabled, the Endpoint and Resource URL changes from HTTP to HTTPS. For more information, see HTTPS Support for Orchestration API.

- **Header:** Content-Type: application/yaml
- **Method:** PUT
- **Payload:** Include the YAML configuration file in the PUT request.

  Sample Payload (YAML format):

```
---
 - name: "Host name"
   alias: "Host internal name"
   ipAddress: "Host IP address"
 - name: "NewHost name"
   alias: "NewHost internal name"
   ipAddress: "NewHost IP address"
```

  Important
  - To add or update AdditionalHosts, update new payload with existing additional hosts information along with new or updated additional hosts. This request replaces all the additional hosts with new additional hosts information.
  - To modify or delete AdditionalHosts, update new payload with modified or deleted additional hosts and perform PUT request. This request replaces additional hosts information in the `/etc/hosts` file of both Cluster Manager and CPS VMs.
  - To verify that the AdditionalHosts configuration is properly loaded, perform another GET request to http://<*Cluster ManagerIP*>:8458/api/system/config/additional-hosts.

- **Response Codes:** 200 OK: success; 400: malformed or invalid; 500: system error

Configuration Parameters - AdditionalHosts

The following parameters can be defined in the AdditionalHosts YAML configuration file:

Parameter	Description
`- name:`	Defines the hostname of the VM. This name must be resolvable in the enterprise DNS environment.
`alias:`	Defines the internal host name used by CPS nodes for internal communication, such as qns01.
`ipAddress:`	Defines the IP address to use in the `/etc/hosts` file.

Secondary Key Ring Configuration

You can create ring during installation for HA or GR systems. If the ring creation fails during installation, you can use APIs to recreate the ring.

The following APIs can be used to create ring configuration:

- During fresh install you can use `http://<cluman-ip>:8458/api/system/config` and `http://<cluman-ip>:8458/api/system/config/action/apply` to create replica-set configuration for all replica-sets and apply it.
- Updated replica-sets (for example, used in scale up of replica-sets) using PATCH method: `http://<cluman-ip>:8458/api/system/config/replica-sets`
- Load updated configuration using PUT method: `http://<cluman-ip>:8458/api/system/mongo/config`

Note You cannot disable ring configuraton.

- Example for HA: replicaSet YAML changes during add replica-set

```
- op: "add"
  title: "SESSION"
  instances: "1"
  arbiter: "pcrfclient01"
  members:
  - "sessionmgr01"
  - "sessionmgr02"
  shardCount: "4"
```

  Use PATCH API `http://<cluman-ip>:8458/api/system/config/replica-sets` to create ring and replica-set.

  Verify ring configuration by executing the following command:

  `echo "db.cache_config.find()" | mongo sessionmgr01:27721/sharding` <-- Change host name and port according to your deployment

- Example for GR:

```
- op: "add"
  title: "SESSION"
  instances: "1"
  arbiter: "arbiter-site3"
  primaryMembers:
  - "sessionmgr01-site1"
  - "sessionmgr02-site1"
```

```
  secondaryMembers:
  - "sessionmgr01-site2"
  - "sessionmgr02-site2"
  seeds: "sessionmgr01:sessionmgr02"
  shardCount: "4"
```

Use PATCH API `http://<cluman-ip>:8458/api/system/config/replica-sets` to create ring and replica-set.

Verify ring configuration by executing the following command:

`echo "db.cache_config.find()" | mongo sessionmgr01:27721/sharding` <-- Change host name and port according to your deployment

- Configure ring in case creation of the replica-set fails:
  - Modify ring operation

```
---
- op: "modify-rings"
  setName: "set09"
```

Call PATCH API `http://<cluman-ip>:8458/api/system/config/replica-sets` to create ring and replica-set.

Note This operation will re-create ring if they are not configured before.

Active-Active Geo HA Support

As an Active-Active GR user you can use an API to configure OSGi commands for distributing traffic across different databases depending upon site-name or host-name.

For the manual steps to configure Active/Active Geo HA, refer to *CPS Geographic Redundancy Guide*.

By default, Geo HA feature is not installed and is not enabled. To install and enable the Geo HA, perform the following steps:

Step 1 Add *isGeoHAEnabled*, *geoHaSessionLookupType*, *enableReloadDict*, *geoSiteName*, *siteId*, and *remoteSiteId* lines in YAML file to install and enable Geo HA feature:

```
---
policyServerConfig:
  geoSiteName: "SITE1"
  siteId: "SITE1"
  remoteSiteId: "SITE2"
  heartBeatMonitorThreadSleepMS: "500"
  mongodbupdaterConnectTimeoutMS: "1000"
  mongodbupdaterSocketTimeoutMS: "1000"
  dbConnectTimeout: "1200"
  threadMaxWaitTime: "1200"
  dbSocketTimeout: "600"
  geoHaSessionLookupType: "realm"
  isGeoHaEnabled: "true"
  enableReloadDict: "true"
  remoteGeoSiteName: “SITE2”
```

```
deploymentType: "GR"
sessionLocalGeoSiteTag: “SITE1”
```

- *isGeoHAEnabled* as `true` installs and enables the Geo HA feature.
- *geoHaSessionLookupType* as `realm` or `host` configures the lookup type.
- *enableReloadDict* is used to enable dictionary reload flag (Only for GR).
- *geoSiteName*, *siteId* and *remoteSiteId* are used to configure site information.

To verify whether Geo HA feature has been enabled or not, execute the following command:
`list_installed_features.sh | grep geo`

Output should be: `com.broadhop.policy.geoha.feature=XXXX`

`grep geoha /etc/broadhop/pcrf/features`

Output should be: `com.broadhop.policy.geoha.feature`

Step 2 Call PATCH API to load the updated configuration:
`curl -i -X PATCH http://<clumanIP>:8458/api/system/config/application-config -H "Content-Type: application/yaml" --data-binary @<yaml file name>`

Step 3 Configure geo lookup information: geoLookupConfig changes can be done during new installation or at a later time.

Important Currently, deleting of lookup keys is not supported

```
grConfig:
  geoLookupConfig:
    - siteId: "SITE1"
      lookupKey:
        - "site-gx-client1.com"
        - "site-gx-client2.com"
```

where,

- *siteId* is the ID of the site for which lookup keys need to be generated.
- *lookupKey* can be `realm` or `host`. This should have same value as configured for *geoHaSessionLookupType* in Step Step 1, on page 51.

a) In case, you update lookup key configuration, you can call the PATCH API:
`curl -i -X PATCH http://<clumanIP>:8458/api/system/config/application-config -H "Content-Type: application/yaml" --data-binary @<yaml file name>`

b) To verify site lookup, use the following OSGi commands:
`nc qns01 9091`

`listsitelookup <SITE-ID>`

Step 4 Add replica-set to add each primary (active) site with its secondary (remote) site ID: For more information, refer to /api/system/config/replica-sets, on page 38.
After adding replica-sets, update using PATCH API: `curl -i -X PATCH http://<clumanIP>:8458/api/system/config/replica-sets -H "Content-Type: application/yaml" --data-binary @<yaml file name>`

Step 5 Add shards for each site: For more information, refer to /api/system/config/replica-sets, on page 38.

After adding shards, update using PATCH API: `curl -i -X PATCH http://<clumanIP>:8458/api/system/config/replica-sets -H "Content-Type: application/yaml" --data-binary @<yaml file name>`