Configure Custom Database Fragmentation Threshold Percentage in CPS

Contents

[Introduction](#)
[Prerequisites](#)
[Requirements](#)
[Components Used](#)
[Background Information](#)
[Configure](#)
[Configurations](#)
[Approach for CPS Hosted in OpenStack](#)
[Approach for CPS Hosted in VMWare](#)
[Verify](#)
[Troubleshoot](#)

Introduction

This document describes how to configure custom database fragmentation threshold percentages in Cisco Policy Suite (CPS).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Linux
- CPS
- MongoDB

Note: Cisco recommends that you have privilege root user access to CPS CLI.

Components Used

The information in this document is based on these software and hardware versions:

- CPS 20.2
- Unified Computing System (UCS)-B
- MongoDB v3.6.17

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

CPS uses MongoDB to constitute its basic database (DB) structure.

Fragmentation is a characteristic of MongoDB. By this alert, you proactively monitor MongoDB fragmentation and thus avoid possible higher resource (disk and memory) utilization due to MongoDB.

CPS generates a Simple Network Management Protocol (SNMP) alarm when MongoDB fragmentation percentage exceeds a specified value.

The **/etc/collectd.d/dbMonitorList.cfg** file present on sessionmgr Virtual Machines (VMs) contains the list of databases and their respective fragmentation threshold percentage values.

Configure

Configurations

Approach for CPS Hosted in OpenStack

Step 1. From the Cluster Manager VM, run this command to create a backup of the current configuration file.

```
#cp /etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg.bkp
```

Step 2. Run this command from Cluster Manager to get the current configuration from sessionmgr VMs (to compare and validate post-change).

```
#for host in $(hosts-all.sh | grep 'sessionmgr'); do echo checking in $host; ssh $host "cat
/etc/collectd.d/dbMonitorList.cfg"; done
```

Sample output:

```
checking in sessionmgr01
session_cache|session|40
sk_cache|secondary_key|40
diameter|endpoints|40
spr|subscriber|40
balance_mgmt|account|40
checking in sessionmgr02
session_cache|session|40
sk_cache|secondary_key|40
diameter|endpoints|40
spr|subscriber|40
balance_mgmt|account|40
```

Step 3. Modify the default threshold (40) to the recommended value (for example; 60). Run this command from Cluster Manager.

> **Note**: This command changes the threshold for all DBs. If the requirement is to update the threshold for a specific DB, update the file manually.

```
#sed -i 's/40/60/g'
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg
```

Step 4. Run this command to compare the files in Cluster manager to validate the change.

```
#diff /etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg.bkp
```

Sample output:

```
4c4
<session_cache|session|60
---
>session_cache|session|40
9c9
<sk_cache|secondary_key|60
---
>sk_cache|secondary_key|40
14c14
<diameter|endpoints|60
---
>diameter|endpoints|40
19c19
<spr|subscriber|60
---
>spr|subscriber|40
24c24
<balance_mgmt|account|60
---
>balance_mgmt|account|40
```

Step 5. Run this command to build the change in Cluster Manager.

```
[root@installer ~]# /var/qps/bin/build/build_puppet.sh
Copying /etc/puppet to /var/qps/images/puppet.tar.gz...
Creating MD5 Checksum...
[root@installer ~]#
```

Step 6. Run this command from Cluster Manager to apply the change in sessionmgr VMs.

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo starting vm-init in
$host; ssh $host "/etc/init.d/vm-init > /dev/null 2>&1 &"; done
starting vm-init in sessionmgr01
starting vm-init in sessionmgr02
[root@installer ~]#
```

Step 7. Wait for the puppet to be completed. Run this command from Cluster Manager to view the progress in Puppet configuration.

```
#for host in $(hosts-all.sh | grep 'sessionmgr' | tail -1); do echo checking in $host; ssh $host
"tail -f /var/log/puppet.log"; done

2022-11-08 06:32:23 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:32:23 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:32:23 +0000 /Stage[main]/Whisper/Service[whisper] (notice): Triggered 'refresh'
from 1 event
```

```
2022-11-08 06:32:27 +0000 Stage[main] (info): Unscheduling all events on Stage[main]
2022-11-08 06:32:28 +0000 Puppet (notice): Applied catalog in 83.52 seconds
[Tue Nov 08 06:32:30 +0000 2022] * Completed puppet configuration for dc1-sessionmgr02...
[Tue Nov 08 06:32:30 +0000 2022] - NTP sync started, check the logs in vm-init.log
```

Approach for CPS Hosted in VMWare

Step 1. Update the **/var/qps/config/deploy/csv/Configuration.csv** file in Cluster Manager with the required database name and their respective threshold percentagevalue. The format to provide the custom threshold percentage value is as this (where XX is the numeric value of percentage...for example; 60).

```
session_cache,XX,
sk_cache,XX,
diameter,XX,
spr,XX,
balance_mgmt,XX,
```

Sample configuration:

```
session_cache,60,
sk_cache,60,
diameter,60,
spr,60,
balance_mgmt,60,
```

Step 2. Run these commands to update the **/etc/collectd.d/dbMonitorList.cfg** file so that it has the new threshold values from the Configuration.csv file:

```
[root@installer ~]# /var/qps/install/current/scripts/import/import_deploy.sh
Filenames that will be processed

AdditionalHosts.csv Configuration.csv DBConfigServer.csv Definitions.csv Hosts.csv
ReplicationSets.csv SessionCache.csv VLANs.csv VMSpecification.csv SecureConfig.csv
VipProxyConfiguration.csv DSCPConfig.csv CriticalFiles.csv

The CSV files in /var/qps/config/deploy/csv are converted to json files in
/var/qps/config/deploy/json..
build the hosts file to /var/www/html/hosts...
build the /etc/hosts file from the json configuation... /etc/hosts is backed to /etc/hosts.back
Skipping backup of '/etc/hosts' -- no changes detected.
Redis by default disabled -DenableQueueSystem=false in /etc/broadhop/qns.conf
Removing feature configs moved to core
Removing ws feature from pb and pcrf feature file
Building /etc/broadhop...
Copying to /var/qps/images/etc.tar.gz...
Creating MD5 Checksum...
Generating /etc/broadhop/servers.all
Rebuilding facts for: 'installer' (aka 'installer')
Creating md5sum for hosts file to validate later
Rebuilding facts for: 'dc1-lb01' (aka 'lb01')
Rebuilding facts for: 'dc1-sessionmgr01' (aka 'sessionmgr01')
Rebuilding facts for: 'dc1-lb02' (aka 'lb02')
Rebuilding facts for: 'dc1-qns01' (aka 'qns01')
Rebuilding facts for: 'dc1-qns02' (aka 'qns02')
Rebuilding facts for: 'dc1-pcrfclient01' (aka 'pcrfclient01')
Rebuilding facts for: 'dc1-sessionmgr02' (aka 'sessionmgr02')
Rebuilding facts for: 'dc1-pcrfclient02' (aka 'pcrfclient02')
No file for VipProxyConfiguration found
Copying /etc/puppet to /var/qps/images/puppet.tar.gz...
```

```
Creating MD5 Checksum...
[root@installer ~]#
```

Step 3. Run this command from Cluster Manager to apply the change in sessionmgr VMs.

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo starting vm-init in
$host; ssh $host "/etc/init.d/vm-init > /dev/null 2>&1 &"; done
starting vm-init in sessionmgr01
starting vm-init in sessionmgr02
[root@installer ~]#
```

Step 4. Wait for the puppet to be completed. Run this command from Cluster Manager to view the progress in Puppet configuration.

```
#for host in $(hosts-all.sh | grep 'sessionmgr' | tail -1); do echo checking in $host; ssh $host
"tail -f /var/log/puppet.log"; done

2022-11-08 06:48:34 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:48:34 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:48:34 +0000 /Stage[main]/Whisper/Service[whisper] (notice): Triggered 'refresh'
from 1 event
2022-11-08 06:48:39 +0000 Stage[main] (info): Unscheduling all events on Stage[main]
2022-11-08 06:48:40 +0000 Puppet (notice): Applied catalog in 93.27 seconds
[Tue Nov 08 06:48:42 +0000 2022] * Completed puppet configuration for dc1-sessionmgr02...
[Tue Nov 08 06:48:42 +0000 2022] - NTP sync started, check the logs in vm-init.log
```

Verify

Use this section in order to confirm that your configuration works properly.

Validate the latest configuration in sessionmgr VMs and compare it with the output of Step 2. Run this command from Cluster Manager.

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo checking in $host;
ssh $host "cat /etc/collectd.d/dbMonitorList.cfg"; done
checking in sessionmgr01
session_cache|session|60
sk_cache|secondary_key|60
diameter|endpoints|60
spr|subscriber|60
balance_mgmt|account|60
checking in sessionmgr02
session_cache|session|60
sk_cache|secondary_key|60
diameter|endpoints|60
spr|subscriber|60
balance_mgmt|account|60
[root@installer ~]#
```

Troubleshoot

This section provides information you can use in order to troubleshoot your configuration.

This MongoDB fragmentation alert was introduced in 20.1 and it was not measured in earlier releases. By default, the fragmentation threshold value is 40%. This threshold value must be

changed based on your deployment size, traffic patterns (call-models), and other traffic patterns factors. Otherwise, CPS throws no/unwanted database fragmentation threshold breached alerts.