Configure CPS Fragmentation Threshold on Mongo Balance Database

Contents

Introduction

This document describes how to configure the threshold for fragmentation of the Balance database (DB).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Linux
- Cisco Policy Suite (CPS)
- MongoDB

Components Used

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

- CPS 20.2.0
- MongoDB

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

Two application patterns cause increased Fragmentation, Document growth, and Document removal. The two operations are usual at the MongoDB level. The best way to defragment is to resync the data across replica members. However, there is still a fragmentation alarm.

- Fragmentation percent threshold values are configured in **/etc/collectd.d/dbMonitorList.cfg** file (present on sessionmgr Virtual Machines) for all the databases. The default threshold value for all the databases is configured as 40%. The default fragmentation threshold value can be changed as required. For more information, refer to Configure Custom Database Fragmentation Threshold Percentage section in the CPS Operations Guide.
- Additionally, verify the current fragmentation percentage of primary members of session_cache, sk_cache, diameter, and Subscriber Profile Repository (SPR) Database with the **diagnostics.sh --get_frag_status** command.
- The **diagnostics.sh --get_frag_status** supports fragmentation percentage calculation of session_cache, sk_cache, diameter, and the SPR Databases.
- The diagnostics.sh --get_frag_status like diagnostics.sh --get_session_shard_health is supported with root user only for multi-cluster High Availability (HA) and Geo-Redundant (GR) setups.

Steps

Pre-check

> **Note**: All the commands are run from cluman.

Get the current Balance DB Fragmentation threshold, that is used in the verification stage.

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

Apply the Change

Add this entry in **/var/qps/config/deploy/csv/Configuration.csv** with the threshold value (change 50 to the required value, for example, 60 on normal fragmentation level).

balance_mgmt,50,

echo -e "\nbalance_mgmt,50," >> /var/qps/config/deploy/csv/Configuration.csv

```
[root@cps194cluman ]# echo -e "\nbalance_mgmt,50," >>
/var/qps/config/deploy/csv/Configuration.csv [root@cps194cluman ]# grep balance_mgmt
Configuration.csv balance_mgmt,50,
```

Import the csv Changes

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

```
[root@cps194cluman csv]# /var/qps/install/current/scripts/import/import_deploy.sh Filenames to
be processed are listed here. 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 Warning: The first
build is not processed. We process only the list of files mentioned. Warning: old files are not
processed. We process only list of files mentioned. 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 Rotate backup '/etc/hosts.back' to '/etc/hosts.back.0' Backed up
'/etc/hosts' to '/etc/hosts.back' ‘/etc/hosts’ -> ‘/etc/hosts.back’ 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: 'casant01-ps01' (aka 'qns01') Rebuilding facts for: 'casant01-pd02'
(aka 'lb02') Rebuilding facts for: 'casant01-sessionmgr01' (aka 'sessionmgr01') Rebuilding facts
for: 'casant01-sessionmgr02' (aka 'sessionmgr02') Rebuilding facts for: 'casant01-oam01' (aka
'pcrfclient01') Rebuilding facts for: 'casant01-ps02' (aka 'qns02') Rebuilding facts for:
'casant01-pd01' (aka 'lb01') Rebuilding facts for: 'casant01-oam02' (aka 'pcrfclient02') Copying
/etc/puppet to /var/qps/images/puppet.tar.gz... Creating MD5 Checksum... [root@cps194cluman
csv]#
```

Build Updated Virtual Machine Images

```
# /var/qps/bin/build/build_all.sh
```

Update the Threshold in sessionmgr Virtual Machines

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

Verification

Verify the threshold has been changed for the Balance DB fragmentation to the configured value.

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

Verify the alarm generation script contains the updated threshold.

```
# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo checking in $host; ssh $host "cat
/var/log/broadhop//scripts/gen-frag-trap.log | grep DEBUG | tail -5"; done
```

Note: Test this in a laboratory before you attempt production setup.