Procedure to Resolve MongoPrimaryDB Fragmentation Alert in CPS

Contents

Introduction

This document describes the procedure to resolve the MongoPrimaryDB Fragmentation alert in Cisco Policy Suite (CPS).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Linux
- CPS
- MongoDB

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

Components Used

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

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

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 where mongod processes that runs on Sessionmgr Virtual Machines (VMs) constitute its basic DataBase structure.

When documents move around or are removed they leave holes. MongoDB tries to reuse these holes for new documents whenever possible, but over time it finds itself possess a lot of holes slowly and steadily, that cannot be reused because documents cannot fit in them. This effect is called fragmentation and is common in all systems that allocate memory, that includes your operating system (OS).

The effect of fragmentation is to waste space. Due to the fact that MongoDB uses memory-mapped files, any fragmentation on disk reflects in fragmentation in RAM as well. This results in the creation of less of the 'Working Set' fit in RAM and causes the disk to swap more.

CPS supports KPIs to monitor MongoDB level fragmentation by the use of Grafana and generates a Simple Network Management Protocol (SNMP) alarm when the MongoDB fragment percentage exceeds a specified value.

The **/etc/collectd.d/dbMonitorList.cfg** file present on sessionmgr virtual machines contains the list of databases and their respective fragmentation threshold percentage values. By default, the fragmentation threshold value is 40 %. The default fragmentation threshold value can be changed as required.

The Fragmentation statistics for session_cache, sk_cache, diameter, and Subscriber Profile Repository (SPR) databases (by the use of primary members) can be checked with this command:

```
[root@installer ~]# diagnostics.sh --get_frag
CPS Diagnostics HA Multi-Node Environment
---------------------------

Ping check for qns03 Adding to IGNORED_HOSTS...[FAIL]
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| Mongo:v3.6.17 DATABASE LEVEL FRAGMENTATION STATUS INFORMATION Date : 2022-09-17 07:19:29 |
| SET TYPE : HA [MEMBER_ROLE : PRIMARY] |
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| setname dbName storageSize(MB) datasize(MB) indexSize(MB) fileSize(MB) derivedFS(MB) frag% |
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| ADMIN:set06 |
| Status via sessionmgr01:27721 |
| set06 diameter 9.56 0.04 0.05 64.00 0 NoFrag |
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| BALANCE:set02 |
| Status via sessionmgr01:27718 |
| set02 balance_mgmt db not found - - - - - - |
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| SESSION:set01 |
| Status via sessionmgr01:27717 |
| set01 session_cache 0.02 0.00 0.02 16.00 0 NoFrag |
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
| SESSION:set01 |
| Status via sessionmgr01:27717 |
| set01 sk_cache 0.02 0.00 0.01 16.00 0 NoFrag |
|-----------------------------------------------------------------------------------------------
----------------------------------------|
| SPR:set04 |
| Status via sessionmgr01:27720 |
| set04 spr 0.04 0.00 0.13 64.00 0 NoFrag |
|-----------------------------------------------------------------------------------------------
----------------------------------------|
[root@installer ~]#
```

Problem

When the fragmentation percentage of the primary member for the replica set exceeds the configured threshold fragmentation value, this alarm is generated. If the threshold value is not configured, then the alarm is raised if the fragmentation percent breaches the default value (40%).

Sample "MongoPrimaryDB fragmentation exceeded the threshold value" alert:

```
id=7100,values={sub_id=7107, event_host=sessionmgr01, status=down, msg=MongoPrimaryDB
fragmentation exceeded the threshold value, CURR_FRAG = 40%, THRESHOLD = 40% at
sessionmgr01:27717 for session_cac
```

Procedure to Resolve MongoPrimaryDB Fragmentation Alert

In order to reduce the fragmentation percentage, shrink the database when an alarm is generated. Once the database is shrunk (fragmentation percentage decreases), a clear alarm is sent.

This procedure is to resolve the MongoPrimaryDB fragmentation alert in the sample provided.

Step 1. Run this command from either Cluster Manager or pcrfclient in order to verify the status of primary and secondary members in the replica set.

```
#diagnostics.sh --get_r

|-----------------------------------------------------------------------------------------------
----------------------------------------|
|SESSION:set01a|
|Status via sessionmgr01:27717 sessionmgr02:27717 |
|Member-1-27717 : 192.168.29.14-ARBITER-pcrfclient01- ON-LINE--0| --------|
|Member-2-27717 : 192.168.29.35-PRIMARY-sessionmgr01- ON-LINE--3| --------|
|Member-3-27717 : 192.168.29.36-SECONDARY-sessionmgr02- ON-LINE--2| 1 sec|
|-----------------------------------------------------------------------------------------------
----------------------------------------|
```

Step 2. Run this command from either Cluster Manager or pcrfclient to change the priority of sessionmgr01 and to make it a secondary member.

```
#sh set_priority.sh --db session --replSet set01a --asc

Expected output in #diagnostics.sh --get_r
|-----------------------------------------------------------------------------------------------
----------------------------------------|
|SESSION:set01a|
|Status via sessionmgr02:27717 sessionmgr01:27717 |
|Member-1-27717 : 192.168.29.14-ARBITER-pcrfclient01- ON-LINE--0| --------|
|Member-2-27717 : 192.168.29.35-PRIMARY-sessionmgr02- ON-LINE--3| --------|
|Member-3-27717 : 192.168.29.36-SECONDARY-sessionmgr01- ON-LINE--2| 1 sec|
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
```

> **Note**: Ensure sessionmgr01 is not primary anymore (diagnostics.sh --get_r) and there is a primary member available for the replica set.

Step 3. Run this command from Sessionmgr01 to stop the AIDO client.

```
#monit stop aido_client
```

Step 4. Run this command from Sessionmgr01 to stop the respective Mongo instance (portNum is the port number of the fragmented member).

```
Command syntax:
#/etc/init.d/sessionmgr-<portNum> stop

Example:
#/etc/init.d/sessionmgr-27717 stop
```

Step 5. In order to clean the database directory in sessionmgr01, remove the data directory from the path mentioned against the --dbpath attribute of the mongo command. Run this command from Sessionmgr01 in order to retrieve the value (use the portNum of the fragmented member).

> **Note**: Since the port number and directories associated with other sessionmgr dbs are different, ensure you have the right directories to clean up other sessionmgr dbs.

```
Command syntax:
#grep -w DBPATH= /etc/init.d/sessionmgr-<portNum>

Example:
#grep -w DBPATH= /etc/init.d/sessionmgr-27717

Sample Output: DBPATH=/var/data/sessions.1/a

Copy the DBPATH from output.

Command syntax:
#rm -rf <DBPATH>/*

Example:
#rm -rf /var/data/sessions.1/a/*
```

Step 6. Run this command from Sessionmgr01 to start the respective Mongo instance.

```
Command syntax:
#/etc/init.d/sessionmgr-<portNum> start

Example:
#/etc/init.d/sessionmgr-27717 start
```

Step 7. Run this command from Sessionmgr01 to start the AIDO client.

```
#monit start aido_client
```

Step 8. Run this command from either Cluster Manager or pcrfclient to reset the priorities of replica set members.

```
#sh set_priority.sh --db session --replSet set01a
```

Step 9. Run this command from either Cluster Manager or pcrfclient to verify the status of primary and secondary members in the replica set.

```
#diagnostics.sh --get_r

|-----------------------------------------------------------------------------------------------
-----------------------------------------|
|SESSION:set01a|
|Status via sessionmgr01:27717 sessionmgr02:27717 |
|Member-1-27717 : 192.168.29.14-ARBITER-pcrfclient01- ON-LINE--0| --------|
|Member-2-27717 : 192.168.29.35-PRIMARY-sessionmgr01- ON-LINE--3| --------|
|Member-3-27717 : 192.168.29.36-SECONDARY-sessionmgr02- ON-LINE--2| 1 sec|
|-----------------------------------------------------------------------------------------------
-----------------------------------------|
```