CPSでセッションレプリカがダウンした場合のサブスクライバセッションの処理

内容

はじめに

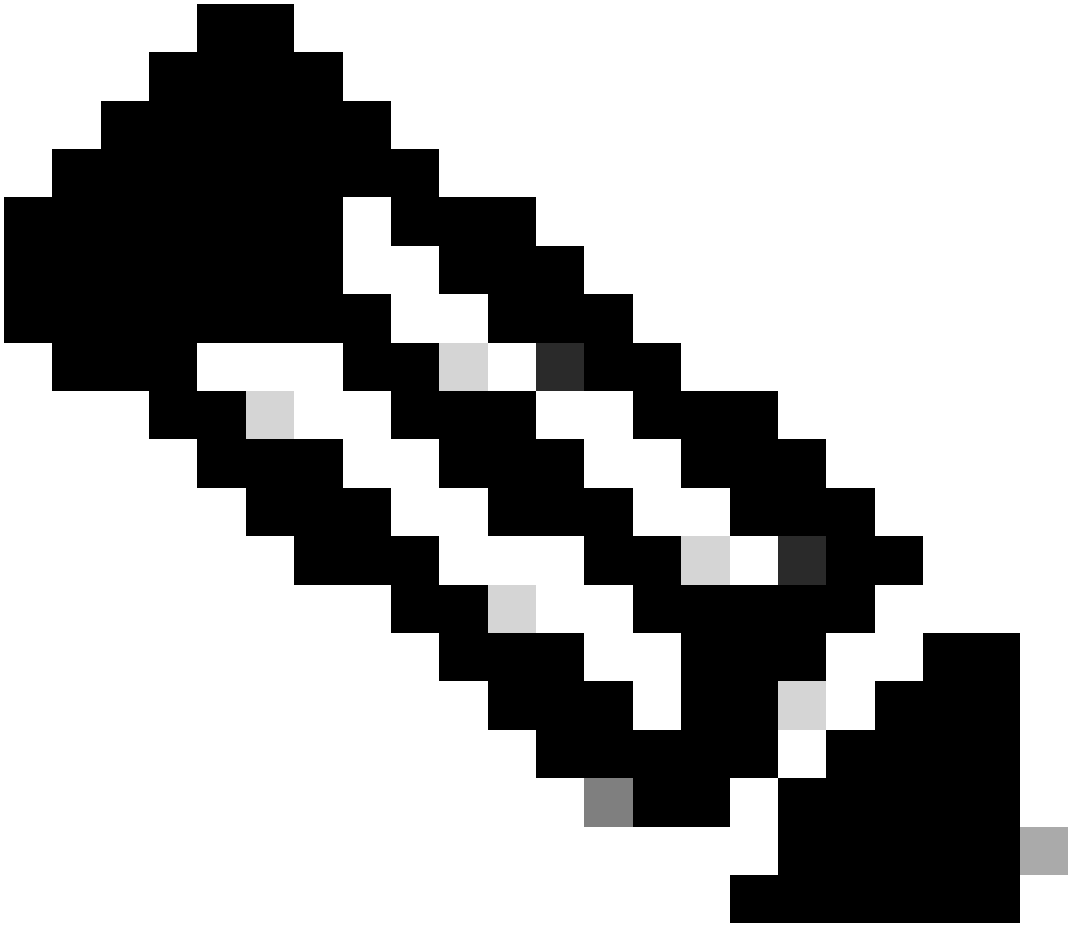
このドキュメントでは、Cisco Policy Suite(CPS)でセッションレプリカセットがダウンした場合のサブスクライバセッションの処理について説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- Linux
- CPS
- MongoDB



注:CPS CLIへのルートアクセス権限が必要です。

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- CPS 20.2
- ユニファイドコンピューティングシステム(UCS)-B
- MongoDB-v3.6.17

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

背景説明

CPSでは、Mongodプロセスがsessionmgr仮想マシン(VM)で実行されるMongoDBを使用して、基本的なデータベース構造を構成します。

レプリカセットのハイアベイラビリティを利用するために推奨される最小限の設定は、3つのデータベアリングメンバ(1つのプライマリメンバと2つのセカンダリメンバ)を持つ3つのメンバレプリカセットです。状況によっては（プライマリとセカンダリがあるものの、コスト制約により別のセカンダリの追加が禁止されている場合など）、アービターを含めることを選択できます。アービターは選択に参加しますが、データを保持しません（つまり、データの冗長性を提供しません）。 CPSの場合、通常はセッションDBをこのように構成する。

CPS設定のレプリカセット設定は、/etc/broadhop/mongoConfig.cfgで確認できます。

```
[SESSION-SET1]
SETNAME=set01a
OPLOG_SIZE=5120
ARBITER1=arbitervip:27717
ARBITER_DATA_PATH=/var/data/sessions.27717
MEMBER1=sessionmgr01:27717
MEMBER2=sessionmgr02:27717
DATA_PATH=/var/data/sessions.1/d
SHARD_COUNT=4
SEEDS=sessionmgr01:sessionmgr02:27717
[SESSION-SET1-END]

[SESSION-SET7]
SETNAME=set01g
OPLOG_SIZE=5120
ARBITER1=arbitervip:37717
ARBITER_DATA_PATH=/var/data/sessions.37717
MEMBER1=sessionmgr02:37717
MEMBER2=sessionmgr01:37717
DATA_PATH=/var/data/sessions.1/g
SHARD_COUNT=2
SEEDS=sessionmgr02:sessionmgr01:37717
[SESSION-SET7-END]
```

MongoDBには、クラスタの冗長性と速度を向上させるShardingという別の概念があります。シャードは、データベースをインデックス付きセットに分割します。これにより、書き込みの速度が大幅に向上し、データベース全体のパフォーマンスが向上します。シャーディングされたデータベースは、各シャードがレプリカセットになるようにセットアップされることがよくあります。

- セッションシャーディング：セッションシャードシードとそのデータベース：

```
osgi> listshards

Shard Id Mongo DB State Backup DB Removed Session Count

1 sessionmgr01:27717/session_cache online false false 109306
2 sessionmgr01:27717/session_cache_2 online false false 109730
3 sessionmgr01:27717/session_cache_3 online false false 109674
4 sessionmgr01:27717/session_cache_4 online false false 108957
```

- セカンダリキーの共有：セカンダリキーはシードとそのデータベースを共有します。

```
osgi> listskshards

Shard Id Mongo DB State Backup DB Removed Session Count

2 sessionmgr02:37717/sk_cache online false false 150306
3 sessionmgr02:37717/sk_cache_2 online false false 149605
```

問題

問題 1.シングルメンバ障害によるデータメンバのメモリ消費の着実な増加

データベアリングメンバが3つのメンバ（2つのデータベアリング+1アービター）でダウンすると、残りのデータベアリングメンバだけがプライマリセットとレプリカセットの役割を引き続き担いますが、DBの冗長性がなく、負荷とレプリカセットの負荷が大きくなります。 3メンバのPSAアーキテクチャでは、いずれかのデータベアリングノードがダウンすると、キャッシュの負荷が増加します。 その結果、残りのデータベアリングノード（プライマリ）のメモリ消費が着実に増加し、放置すると使用可能なメモリが枯渇してノード障害を引き起こし、最終的にはレプリカセット障害を引き起こす可能性があります。

```
----------------------------------------------------------------------------------------------
|[SESSION:set01a]|
|[Status via sessionmgr02:27717 ]|
|[Member-1 - 27717 : 192.168.29.100 - ARBITER - arbitervip - ON-LINE
|[Member-2 - 27717 : 192.168.29.35  - UNKNOWN - sessionmgr01 - OFF-LINE 19765 days
|[Member-3 - 27717 : 192.168.29.36  - PRIMARY - sessionmgr02 - ON-LINE
----------------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------------
|[SESSION:set01g]|
|[Status via sessionmgr02:37717 ]|
|[Member-1 - 37717 : 192.168.29.100 - ARBITER - arbitervip - ON-LINE
|[Member-2 - 37717 : 192.168.29.35  - UNKNOWN - sessionmgr01 - OFF-LINE 19765 days
|[Member-3 - 37717 : 192.168.29.36  - PRIMARY - sessionmgr02 - ON-LINE
----------------------------------------------------------------------------------------------
```

問題 2：二重メンバー障害によるセッション処理への影響

このようなレプリカセットでデータベアリングメンバ（Sessionmgr01とSessionmgr02）の両方がダウンすると（二重障害）、レプリカセット全体がダウンし、基本データベース機能が損なわれます。

```
Current setup have problem while connecting to the server on port : 27717
```

```
Current setup have problem while connecting to the server on port : 37717
```

セッションレプリカセットの場合、CPSコール処理プロセス(Quantum Network Suite(qns)プロセス)は、障害が発生したレプリカセットにすでに格納されているセッションにアクセスできないため、このレプリカセットの障害によりコールが失敗します。

解決方法

アプローチ1:シングルメンバ障害の場合。

障害が発生したレプリカ・セット・メンバーをPSA (プライマリ/セカンダリ・アービター) アーキテクチャに短時間で戻すことができるようにする必要があります。PSAアーキテクチャで障害が発生したデータベアリングメンバの復元に時間がかかる場合は、障害を取り除く必要があります。

ステップ 1.1：3メンバのPSAアーキテクチャを使用する特定のレプリカセットで障害が発生したデータベアリングメンバを特定します。Cluster Managerからこのコマンドを実行します。

```
#diagnostics.sh --get_r
```

ステップ 1.2：特定のレプリカセットから、障害が発生したデータベアリングメンバを削除します。

```
Syntax:

Usage: build_set.sh <--option1> <--option2> [--setname SETNAME] [--help]

option1: Database name
option2: Build operations (create, add or remove members)

Example:

#build_set.sh --session --remove-failed-members --setname set01a --force
#build_set.sh --session --remove-failed-members --setname set01g --force
```

ステップ 1.3：失敗したメンバがレプリカセットから削除されたことを確認します。

```
#diagnostics.sh --get_r
```

アプローチ2:ダブルメンバの障害の場合。

これは、3つのPSAレプリカセットで両方のデータベアリングメンバがダウンした場合の恒久的な回避策ではありません。むしろ、コール障害を回避または軽減し、シームレスなトラフィック処理を保証するための一時的な回避策です。この方法では、対応するレプリカセット、セッションシャード、およびskから障害のあるメンバーを削除し、適切に共有します。これ以上の望ましくない影響を避けるために、できるだけ早く失敗したメンバーの復元に取り組む必要があります。

ステップ 2.1：Sessionmgr09とSessionmgr10がダウンすると、セッション・レプリカ・セットもダウンするため、OSGIコンソールからセッション・シャードとスケードから次のレプリカ・セットのエントリを削除する必要があります。

```
#telnet qns0x 9091

osgi> listshards

Shard Id Mongo DB State Backup DB Removed Session Count

1 sessionmgr01:27717/session_cache online false false 109306
2 sessionmgr01:27717/session_cache_2 online false false 109730
3 sessionmgr01:27717/session_cache_3 online false false 109674
4 sessionmgr01:27717/session_cache_4 online false false 108957

osgi> listskshards

Shard Id Mongo DB State Backup DB Removed Session Count
2 sessionmgr02:37717/sk_cache online false false 150306
3 sessionmgr02:37717/sk_cache_2 online false false 149605
```

ステップ 2.2：次のセッションシャードを削除します。

```
 osgi> removeshard 1
 osgi> removeshard 2
 osgi> removeshard 3
 osgi> removeshard 4
```

ステップ 2.3：次のSkshardを削除します。

```
osgi> removeskshard 2
osgi> removeskshard 3
```

ステップ 2.4：リバランスを実行する前に、管理DBを確認します（インスタンスバージョンがすべてのQNS VMに対して一致していることを確認します）。

<#root>

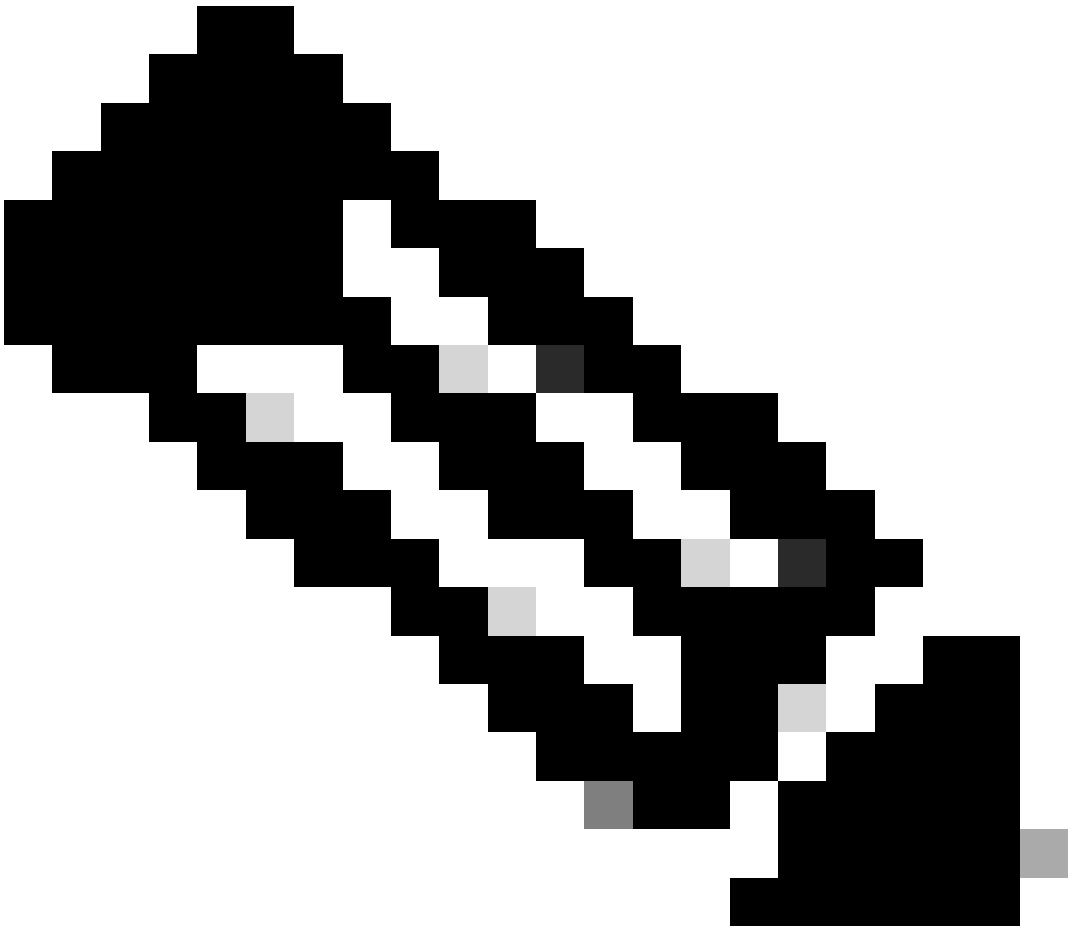
```
#mongo sessionmgrxx:xxxx/sharding.     [Note: use the primary sessionmgr hostname and respective port

#set05:PRIMARY>  db.instances.find()
{ "_id" : "qns02-1", "version" : 961 }
{ "_id" : "qns07-1", "version" : 961 }
{ "_id" : "qns08-1", "version" : 961 }
{ "_id" : "qns04-1", "version" : 961 }
{ "_id" : "qns08-1", "version" : 961 }
{ "_id" : "qns05-1", "version" : 961 }
```

Note: if the sharding versions (previous output) are different for some QNS instances. For example, if y

```
{ "_id" : "qns08-1", "version" : 961 }
{ "_id" : "qns04-1", "version" : 962 }
```

admin sharding DBで（適切なホスト名を使用して）次のコマンドを実行します。



注：セカンダリメンバ上にいる場合は、コマンドを実行できるようにするために rs.slaveOk()を使用します。

```
[root@casant01-cm csv]#  mongo sessionmgr01:27721/shardin
set05:PRIMARY>
set05:PRIMARY> db.instances.remove({ “_id” : “$QNS_hostname”})


Example:

set05:PRIMARY> db.instances.remove({ “_id” : “qns04-1”})
set05:PRIMARY> exit
```

ステップ 2.5：次に、セッションシャードリバランスを実行します。

```
Login to osgi console.

#telnet qns0x 9091

osgi>listshards

osgi>rebalance

osgi>rebalancestatus

Verify shards:

osgi>listshards
```

ステップ 2.6：sk shard rebalanceを実行します。

```
Login to osgi console.

#telnet qns0x 9091

osgi>listskshard

osgi>rebalancesk

osgi>rebalanceskstatus

Verify shards:

osgi>listshards
```

ステップ 2.7：複製セットset01aとset01gを削除します（ 列で実行 ）。

```
#build_set.sh --session --remove-replica-set --setname set01a --force
#build_set.sh --session --remove-replica-set --setname set01g --force
```

ステップ 2.8：qnsサービスを再起動します（ 列で実行 ）:

```
#restartall.sh
```

ステップ 2.9： mongoConfig.cfgファイルからset01aとset01gの行を削除します。Cluster Managerで次を実行します。

```
#cd /etc/broadhop/
#/bin/cp -p mongoConfig.cfg mongoConfig.cfg_backup_


#vi mongoConfig.cfg

[SESSION-SET1]
SETNAME=set01a
OPLOG_SIZE=5120
ARBITER1=arbitervip:27717
ARBITER_DATA_PATH=/var/data/sessions.27717
MEMBER1=sessionmgr01:27717
MEMBER2=sessionmgr02:27717
DATA_PATH=/var/data/sessions.1/d
SHARD_COUNT=4
SEEDS=sessionmgr01:sessionmgr02:27717
[SESSION-SET1-END]

[SESSION-SET7]
SETNAME=set01g
OPLOG_SIZE=5120
ARBITER1=arbitervip:37717
ARBITER_DATA_PATH=/var/data/sessions.37717
MEMBER1=sessionmgr02:37717
MEMBER2=sessionmgr01:37717
DATA_PATH=/var/data/sessions.1/g
SHARD_COUNT=2
SEEDS=sessionmgr02:sessionmgr01:37717
[SESSION-SET7-END]
```

ステップ 2.10： 行を削除したら、保存して終了します。

Cluster Managerでbuild_etcを実行します。

```
#/var/qps/install/current/scripts/build/build_etc.sh
```

ステップ 2.11： Diagnosticsでレプリカセット01dが削除されたことを確認します。

```
#diagnostics.sh --get_r
```

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。