Tratamento de Sessão de Assinante no Caso de Réplica de Sessão Definida no CPS

Contents

Introdução

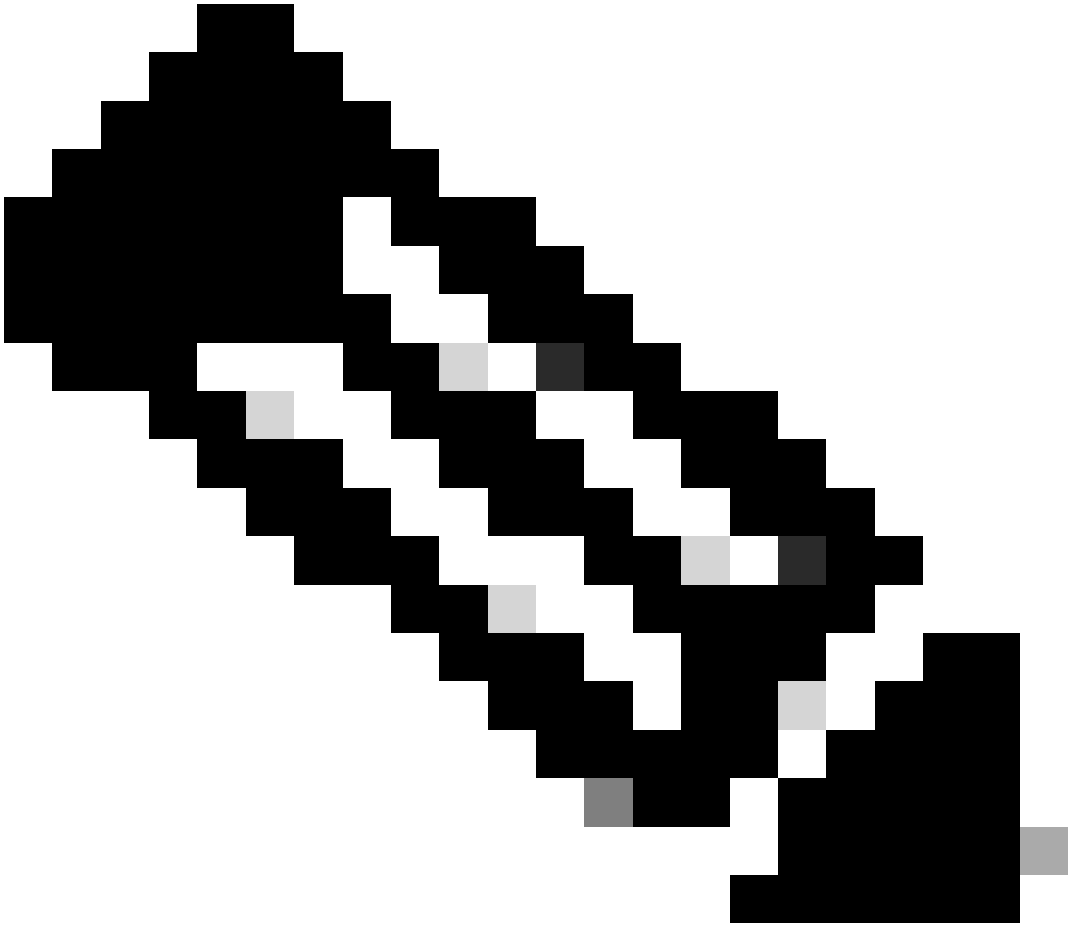
Este documento descreve o Tratamento de Sessão de Assinante no caso de Réplica de Sessão Definida no Cisco Policy Suite (CPS).

Pré-requisitos

Requisitos

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Linux
- CPS
- MongoDB



Note: A Cisco recomenda que você tenha acesso de raiz privilegiado à CLI do CPS.

Componentes Utilizados

As informações neste documento são baseadas nestas versões de software e hardware:

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

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

O CPS usa o MongoDB, onde os processos mongod são executados em máquinas virtuais (VMs) do sessionmgr para constituir sua estrutura básica de banco de dados.

A configuração mínima recomendada para disponibilizar alta disponibilidade para um conjunto de réplicas é um conjunto de réplicas de três membros com três membros que suportam dados: um membro principal e dois membros secundários. Em algumas circunstâncias (como se você tivesse um primário e um secundário, mas as restrições de custo proibirem a adição de outro secundário), você pode optar por incluir um intermediário. Um árbitro participa de eleições, mas não mantém dados (ou seja, não fornece redundância de dados). No caso do CPS, normalmente os BDs da sessão são configurados dessa maneira.

Você pode verificar a configuração do conjunto de réplicas para a instalação do CPS em /etc/broadhop/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]
```

MongoDB tem outro conceito chamado Sharding que ajuda a redundância e velocidade para um cluster. Os compartilhamentos separam o banco de dados em conjuntos indexados, o que permite uma velocidade muito maior para gravações, o que melhora o desempenho geral do banco de dados. Os bancos de dados compartilhados geralmente são configurados de forma que cada fragmento seja um conjunto de réplicas.

- Compartilhamento de sessão: Sementes compartilhadas de sessão e seus bancos de dados:

```
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
```

- Compartilhamento de chave secundária: A chave secundária compartilha sementes e seus bancos de dados:

```
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
```

Problema

Problema 1. Crescimento constante no consumo de memória do membro de dados devido a uma falha de um único membro.

Quando um membro que carrega dados fica inativo em 3 membros (2 arbitradores +1 que suportam dados), o único membro que suporta dados restante assume a função de Conjunto principal e de réplica continua a funcionar, mas com carga pesada e conjunto de réplicas sem nenhuma redundância de BD. Com uma arquitetura PSA de três membros, a pressão de cache aumenta se algum nó que suporta dados estiver inativo. Isso resulta em um aumento constante no consumo de memória para o nó que suporta dados restante (Primário), levando potencialmente à falha do nó devido à redução da memória disponível se deixado sem supervisão e, por fim, causa a falha do conjunto de réplicas.

```
----------------------------------------------------------------------------------------------
|[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
----------------------------------------------------------------------------------------------
```

Problema 2. Impacto no tratamento da sessão devido à falha do membro duplo.

Quando ambos os membros portadores de dados (Sessionmgr01 e sessionmgr02) ficam inativos em tais conjuntos de réplicas (Falha dupla), todo o conjunto de réplicas fica inativo e sua função básica de banco de dados é comprometida.

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

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

Essa falha do conjunto de réplicas resulta em falha de chamada no caso de conjuntos de réplicas de sessão, pois os processos de tratamento de chamadas do CPS (Processos do Quantum Network Suite (qns)) não podem acessar as sessões que já estão armazenadas nesses conjuntos de réplicas com falha.

Solução

Abordagem 1. Para Falha De Membro Único.

Você deve ser capaz de trazer de volta o membro do conjunto de réplicas com falha em uma arquitetura PSA (Primary-Secondary-Arbiter, árbitro primário secundário) com pouco tempo. No caso de a restauração de um membro portador de dados com falha em uma arquitetura PSA levar tempo, você deve remover o membro com falha.

Etapa 1.1. Identifique o membro portador de dados com falha no conjunto de réplicas específico com a arquitetura PSA de três membros. Execute este comando a partir do Gerenciador de Cluster.

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

Etapa 1.2. Remova o membro que carrega os dados com falha do conjunto de réplicas específico.

```
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
```

Etapa 1.3. Verifique se o membro com falha foi removido do conjunto de réplicas.

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

Abordagem 2. Para Falha De Membro Duplo.

Essa não é uma solução permanente quando os dois membros que suportam dados ficam inativos em um conjunto de réplicas 3 PSA. Em vez disso, é uma solução temporária para evitar ou reduzir as falhas de chamada e garantir o tratamento de tráfego perfeito, removendo os membros com falha do respectivo conjunto de réplicas, compartilhamento de sessão e compartilhamento de tarefas de acordo. Você deve trabalhar na restauração de membros falhados o mais rápido possível, a fim de evitar quaisquer efeitos indesejáveis adicionais.

Etapa 2.1. Como os conjuntos de réplicas de sessão estão inativos enquanto o Sessionmgr09 e o Sessionmgr10 estão inativos, você deve remover as entradas desses conjuntos de réplicas do fragmento e do skshard de sessão do console OSGI:

```
#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
```

Etapa 2.2. Remover estes fragmentos de sessão:

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

Etapa 2.3. Remova estes skshards:

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

Etapa 2.4.Antes de executar o rebalanceamento, verifique o banco de dados do administrador (verifique se a versão da instância corresponde a todas as VMs qns):

<#root>

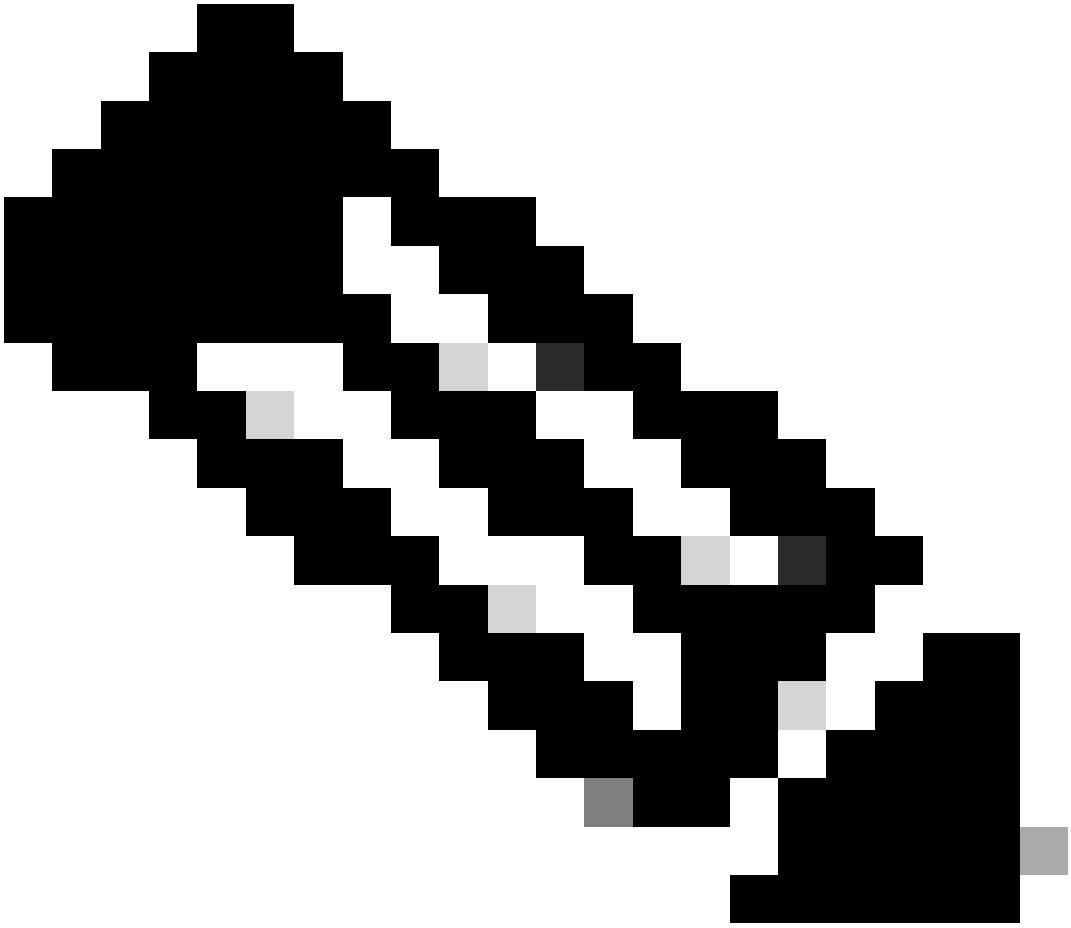
```
 #mongo sessionmgrxx:xxxx/sharding.     [Note: use the primary sessionmgr hostname and respective port

 #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 }
```

Execute este comando no BD de compartilhamento admin (usando o nome de host apropriado):



Note: Se você estiver em um membro secundário, use rs.slaveOk() para poder executar comandos.

```
[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
```

Etapa 2.5. Agora execute o rebalanceamento compartilhado da sessão.

```
Login to osgi console.

#telnet qns0x 9091

osgi>listshards

osgi>rebalance

osgi>rebalancestatus

Verify shards:

osgi>listshards
```

Etapa 2.6. Executar rebalanceamento de compartilhamento de risco:

```
Login to osgi console.

#telnet qns0x 9091

osgi>listskshard

osgi>rebalancesk

osgi>rebalanceskstatus

Verify shards:

osgi>listshards
```

Etapa 2.7. Remova o conjunto de réplicas set01a e set01g (executado no clone):

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

Etapa 2.8. Reiniciar o serviço qns (executar no clone):

```
#restartall.sh
```

Etapa 2.9. Remova as linhas set01a e set01g do arquivo mongoConfig.cfg. Executar no Gerenciador de Cluster:

```
#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]
```

Etapa 2.10. Depois de remover as linhas, salve e saia.

Execute build_etc no Gerenciador de cluster.

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

Etapa 2.11. Verifique se o conjunto de réplicas set01d foi removido através do diagnóstico.

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

Sobre esta tradução
A Cisco traduziu este documento com a ajuda de tecnologias de tradução automática e humana para oferecer conteúdo de suporte aos seus usuários no seu próprio idioma, independentemente da localização.

Observe que mesmo a melhor tradução automática não será tão precisa quanto as realizadas por um tradutor profissional.

A Cisco Systems, Inc. não se responsabiliza pela precisão destas traduções e recomenda que o documento original em inglês (link fornecido) seja sempre consultado.