Procedimiento para controlar la función y la prioridad de los administradores de sesiones en un conjunto de réplicas de CPS

Contenido

Introducción

Este documento describe el procedimiento para mover sessionmgr de la función principal y cambiar la prioridad sessionmgr en un conjunto de réplicas de Cisco Policy Suite (CPS).

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- Linux
- CPS
- MongoDB

Cisco recomienda que tenga acceso de privilegio a la raíz de CPS CLI.

Componentes Utilizados

La información que contiene este documento se basa en las siguientes versiones de software y hardware.

- CPS 20.2
- MongoDB v3.6.17
- 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. Si tiene una red

en vivo, asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

CPS utiliza MongoDB, donde se ejecutan los procesos mondios en la Máquina Virtual Sessionmgr (VM) para constituir su estructura básica de base de datos. Posee varios conjuntos de réplicas para diversos fines, que son ADMIN, Subscriber Profile Repository (SPR), BALANCE, SESSION, REPORTING y AUDIT.

Una réplica establecida en MongoDB es un grupo de procesos mondios que mantienen el mismo conjunto de datos. Los conjuntos de réplicas proporcionan redundancia y alta disponibilidad.Con varias copias de datos en diferentes servidores de base de datos, permite operaciones de lectura de loadshare.

Un conjunto de réplicas contiene varios nodos que contienen datos y opcionalmente un nodo de árbitro. De los nodos que contienen datos , uno y un solo miembro se considera el nodo primario, mientras que los otros nodos se consideran nodos secundarios (un conjunto de réplicas puede tener varios secundarios). El nodo primario controla todas las operaciones de escritura.

Los secundarios replican los registros de operaciones (oplog) primarios y aplican las operaciones a sus conjuntos de datos de modo que los conjuntos de datos secundarios reflejen el conjunto de datos primario. Si el primario no está disponible, un secundario elegible tiene una elección para elegir el nuevo primario. Un árbitro participa en las elecciones pero no contiene datos.

Para obtener el estado de conjuntos de réplicas, ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient.

Se proporciona un ejemplo de conjunto de réplicas, por ejemplo. **set07**.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC -PRIORITY
|-------------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 0 sec - 2 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|-------------------------------------------------------------------------------------------------
----------------------------------------|
```

Para obtener la información de configuración del conjunto de réplicas, utilice estos pasos.

Paso 1. Inicie sesión en el miembro MongoDB principal de ese conjunto de réplicas. Ejecute este comando desde ClusterManager.

```
Command template:
#mongo --host <sessionmgrXX> --port <Replica Set port>

Sample command:
#mongo --host sessionmgr02 --port 27727
```

Paso 2. Ejecute el comando para obtener la información de configuración del conjunto de réplicas.

```
set07:PRIMARY> rs.conf()
{
"_id" : "set07",
"version" : 2,
"members" : [
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 2,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 3,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
```

```
set07:PRIMARY>
```

> **Nota:** Sessionmgr con la prioridad más alta en un conjunto de réplicas funciona como miembro principal.

Problema

Suponga que un session mgr realiza la función de un miembro primario en uno o más conjuntos de réplicas y, en estos casos, debe mover la función principal del conjunto de réplicas a algún otro miembro de sesión,

1. Siempre que realice cualquier actividad que implique el cierre de la VM de sessionmgr, para una transición fluida.

2. Si el estado de sessionmgr se degrada por alguna razón, para mantener la función adecuada del Conjunto de réplicas con algún otro mgr de sesión saludable.

Procedimiento para mover el mensaje de sesión de principal y para cambiar la prioridad de sesión mgr en un conjunto de réplicas

Enfoque 1

Aquí la prioridad de sessionmgr en un conjunto de réplicas cambió directamente en el nivel MongoDB. Estos son los pasos para mover la sesión mg02 de un rol primario en **set07**.

Opción 1. Cambie la prioridad de sessionmgr02.

Paso 1. Inicie sesión en el miembro MongoDB principal de ese conjunto de réplicas.

```
Command template:
#mongo --host <sessionmgrXX> --port <Replica Set port>

Sample command:
#mongo --host sessionmgr02 --port 27727
```

Paso 2. Ejecute el comando para obtener la información de configuración del conjunto de réplicas.

```
set07:PRIMARY> rs.conf()
{
"_id" : "set07",
"version" : 2,
"members" : [
{
"_id" : 0, -----> Position 0
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 2,
```

```
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1, -----> Position 1
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2, -----> Position 2
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 3,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
set07:PRIMARY>
```

Nota: Tome nota de la posición de su respectivo sessionmgr en la salida rs.conf().

Paso 3. Ejecute este comando para mover el terminal al modo de configuración.

```
set07:PRIMARY> cfg = rs.conf()
{
"_id" : "set07",
"version" : 2,
"members" : [
```

```
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 2,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 3,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
set07:PRIMARY>
```

Paso 4. Ejecute este comando para cambiar la prioridad de sessionmgr.

```
Command template:
cfg.members[X].priority = X --> put the position here in [].
```

```
sample command:
cfg.members[2].priority = 1
```

En este momento, session mgr02 es el miembro principal y su posición es **2** y la prioridad es **3**.

Para sacar esta sesión mgr02 de la función principal, proporcione el número de prioridad más bajo mayor que 0 pero menor que la prioridad de un miembro secundario que tenga la prioridad más alta, por ejemplo. **1** en este comando.

```
set07:PRIMARY> cfg.members[2].priority = 1
1
set07:PRIMARY>
```

Paso 5. Ejecute este comando para confirmar el cambio.

```
set07:PRIMARY> rs.reconfig(cfg)
{
"ok" : 1,
"operationTime" : Timestamp(1641528658, 1),
"$clusterTime" : {
"clusterTime" : Timestamp(1641528658, 1),
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
"keyId" : NumberLong(0)
}
}
}
2022-01-07T04:10:57.280+0000 I NETWORK [thread1] trying reconnect to sessionmgr02:27727
(192.168.10.140) failed
2022-01-07T04:10:57.281+0000 I NETWORK [thread1] reconnect sessionmgr02:27727 (192.168.10.140)
ok
set07:SECONDARY>
```

Paso 6. Ejecute de nuevo el comando para verificar los cambios en la prioridad sessionmgr.

```
set07:SECONDARY> rs.conf()
{
"_id" : "set07",
"version" : 3,
"members" : [
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 2,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
```

```
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 1, --> Here priority has been changed from 3 to 1.
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
set07:SECONDARY>
```

Paso 7. Ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient para verificar los cambios en el estado Conjunto de réplicas.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC - PRIORITY |
-----------------------------------------------------------------------------------------------------------
-----------------------------------------
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - PRIMARY - sessionmgr01 - ON-LINE - -------- - 2 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - SECONDARY - sessionmgr02 - ON-LINE - 0 sec - 1 |
|-----------------------------------------------------------------------------------------------------------
----------------------------------------|
```

Ahora, verá que sessionmgr02 se ha movido a secundario. Para hacer que sessionmgr02 vuelva a ser miembro primario, ejecute los pasos 1 mencionados anteriormente. a 5. con este comando en el Paso 4.

cfg.member[2].priority = Cualquier número mayor que 2 pero menor que 1001 —> pone la

prioridad más alta que la del miembro primario actual que es **2** en la muestra.

```
set07:PRIMARY> cfg.members[2].priority = 5
5
set07:PRIMARY> rs.reconfig(cfg)
{
"ok" : 1,
"operationTime" : Timestamp(1641531450, 1),
"$clusterTime" : {
"clusterTime" : Timestamp(1641531450, 1),
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
"keyId" : NumberLong(0)
}
}
}
2022-01-07T04:57:31.247+0000 I NETWORK [thread1] trying reconnect to sessionmgr01:27727
(192.168.10.139) failed
2022-01-07T04:57:31.247+0000 I NETWORK [thread1] reconnect sessionmgr01:27727 (192.168.10.139)
ok
set07:SECONDARY>
```

Ejecute el comando para verificar los cambios en la prioridad sessionmgr.

```
set07:SECONDARY> rs.conf()
{
"_id" : "set07",
"version" : 4,
"members" : [
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 2,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
```

```
"priority" : 5, --> Here priority has been changed from 1 to 5.
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
set07:SECONDARY>
```

Ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient para verificar los cambios en el estado Conjunto de réplicas.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC -PRIORITY
|-------------------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 14 sec - 2 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 5 |
|-------------------------------------------------------------------------------------------------------
----------------------------------------|
```

Ahora, puede ver que sessionmgr02 se ha convertido en primario de nuevo.

Opción 2. Cambie la prioridad de otro mgr de sesión secundario para convertirlo en miembro primario. Aquí está la sesión mgr01.

Para hacer sesión mgr01 como miembro principal, ejecute los pasos 1 mencionados anteriormente. a 5. en la opción 1. con este comando en el Paso 4.

cfg.member[0].priority = Cualquier número mayor que 3 pero menor que 1001 —> Ponga la prioridad más alta que la del miembro primario actual que es "3" en la muestra.

```
set07:PRIMARY> cfg.members[0].priority = 4
4
set07:PRIMARY> rs.reconfig(cfg)
{
"ok" : 1,
"operationTime" : Timestamp(1641540587, 1),
"$clusterTime" : {
"clusterTime" : Timestamp(1641540587, 1),
```

```
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
"keyId" : NumberLong(0)
}
}
}
2022-01-07T07:29:46.141+0000 I NETWORK [thread1] trying reconnect to sessionmgr02:27727
(192.168.10.140) failed
2022-01-07T07:29:46.142+0000 I NETWORK [thread1] reconnect sessionmgr02:27727 (192.168.10.140)
ok
set07:SECONDARY>
```

Ejecute el comando para confirmar los cambios.

```
set07:SECONDARY> rs.conf()
{
"_id" : "set07",
"version" : 4,
"members" : [
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 4, --> Here priority has been changed from 2 to 4.
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 3,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
```

```
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
}
set07:SECONDARY>
```

Ejecute el comando **diagnostics.sh —get_r** desde el Administrador del clúster o el cliente para verificar los cambios en el estado Conjunto de réplicas.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC -PRIORITY
|-----------------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - PRIMARY - sessionmgr01 - ON-LINE - -------- - 4 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - SECONDARY - sessionmgr02 - ON-LINE - 0 sec - 3 |
|-----------------------------------------------------------------------------------------------------
----------------------------------------|
```

Ahora, puede ver que sessionmgr01 se ha convertido en primario, mientras que sessionmgr02 se ha convertido en secundario.

Para hacer sesión mgr02 como miembro principal de nuevo, ejecute los pasos 1 mencionados anteriormente. a 5. en la **Opción 1** con este comando en el Paso 4.

cfg.member[0].priority = Cualquier número menor que 3 pero mayor que 0 —> Ponga la prioridad por debajo de la de sessionmgr02 que es "3" en la muestra.

```
set07:PRIMARY> cfg.members[0].priority = 1
1
set07:PRIMARY> rs.reconfig(cfg)
{
"ok" : 1,
"operationTime" : Timestamp(1641531450, 1),
"$clusterTime" : {
"clusterTime" : Timestamp(1641531450, 1),
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
"keyId" : NumberLong(0)
}
}
}
2022-01-07T08:34:31.165+0000 I NETWORK [thread1] trying reconnect to sessionmgr01:27727
(192.168.10.139) failed
2022-01-07T08:34:31.165+0000 I NETWORK [thread1] reconnect sessionmgr01:27727 (192.168.10.139)
ok
set07:SECONDARY>
```

Ejecute este comando para verificar los cambios en la prioridad sessionmgr.

```
set07:SECONDARY> rs.conf()
{
"_id" : "set07",
"version" : 4,
"members" : [
{
"_id" : 0,
"host" : "sessionmgr01:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 1, --> Here priority has been changed from 4 to 1.
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 1,
"host" : "arbitervip:27727",
"arbiterOnly" : true,
"buildIndexes" : true,
"hidden" : false,
"priority" : 0,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
},
{
"_id" : 2,
"host" : "sessionmgr02:27727",
"arbiterOnly" : false,
"buildIndexes" : true,
"hidden" : false,
"priority" : 3,
"tags" : {

},
"slaveDelay" : NumberLong(0),
"votes" : 1
}
],
"settings" : {
"chainingAllowed" : true,
"heartbeatIntervalMillis" : 2000,
"heartbeatTimeoutSecs" : 1,
"electionTimeoutMillis" : 10000,
"catchUpTimeoutMillis" : -1,
"catchUpTakeoverDelayMillis" : 30000,
"getLastErrorModes" : {

},
"getLastErrorDefaults" : {
"w" : 1,
"wtimeout" : 0
},
"replicaSetId" : ObjectId("61cdb17a80b097a2e7604c97")
}
```

```
}
set07:SECONDARY>
```

Ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient para verificar los cambios en el estado Conjunto de réplicas.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC -PRIORITY
|-------------------------------------------------------------------------------------------------------
---------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 14 sec - 1 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|-------------------------------------------------------------------------------------------------------
---------------------------------------|
```

Ahora, puede ver que sessionmgr02 se ha convertido en primario, mientras que sessionmgr01 es secundario.

Enfoque 2

Puede utilizar el script CPS **set_priority.sh** de ClusterManager para cambiar la prioridad sessionmgr en un conjunto de réplicas. De forma predeterminada, la prioridad de los miembros se establece en orden (con mayor prioridad), como se define en **/etc/broadhop/mongoConfig.cfg** en ClusterManager.

Tomemos el ejemplo set07.

```
[root@installer broadhop]# cat mongoConfig.cfg
[SESSION-SET2]
SETNAME=set07
OPLOG_SIZE=5120
ARBITER=arbitervip:27727
ARBITER_DATA_PATH=/var/data/sessions.7
MEMBER1=sessionmgr02:27727
MEMBER2=sessionmgr01:27727
DATA_PATH=/var/data/sessions.1/2
[SESSION-SET2-END]
```

Para obtener el estado del conjunto de réplicas, ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC - PRIORITY |
|-------------------------------------------------------------------------------------------------------
---------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 0 sec - 2 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|-------------------------------------------------------------------------------------------------------
---------------------------------------|
```

Cuando compara los resultados mencionados anteriormente, puede ver que sessionmgr02 es el primer miembro[MEMBER1] de set07 en **/etc/broadhop/mongoConfig.cfg** , por lo que sessionmgr02 es el miembro principal en set07 de forma predeterminada.

Aquí se proporcionan las opciones de alta disponibilidad de CPS que utilizan el script **set_priority.sh** para mover el sessionmgr02 fuera de la función de miembro principal en set07.

Paso 1. Establezca la prioridad en orden ascendente.

```
Command template:
sh set_priority.sh --db arg --replSet arg --asc

where ,
--db arg --> arg is database name
[all|session|spr|admin|balance|report|portal|audit|bindings|session_configs|bindings_configs|spr
_configs]

--replSet arg -->arg is <setname>

Sample command:
sh set_priority.sh --db session --replSet set07 --asc

[root@installer ~]# sh set_priority.sh --db session --replSet set07 --asc

Set priorities is in progress. check log /var/log/broadhop/scripts/set_priority.log to know the
status
Setting priority for Replica-Set: SESSION-SET2
INFO Parsing Mongo Config file
INFO Priority set operation is completed for SESSION-SET2
INFO Priority set to the Database members is finished
INFO Validating if Priority is set correctly for Replica-Set: SESSION-SET2
WARNING Mongo Server trying to reconnect while getting config. Attempt #1
INFO Validated Priority is set correctly for Replica-Set: SESSION-SET2
Primary member sessionmgr01:27727 found for Replica SESSION-SET2

Set priorities process successfully completed.

[root@installer ~]#
```

Paso 2. Ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient para verificar el cambio.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC - PRIORITY |
|-------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - PRIMARY - sessionmgr01 - ON-LINE - -------- - 3 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - SECONDARY - sessionmgr02 - ON-LINE - 0 sec - 2 |
|-------------------------------------------------------------------------------------------
----------------------------------------|
```

Ahora, sessionmgr01 se ha convertido en el miembro principal, ya que se ha establecido una prioridad en el orden ascendente definido en **/etc/broadhop/mongoConfig.cfg**.

Para hacer que sessionmgr02 vuelva a ser un miembro primario, ejecute este comando.

```
[root@installer ~]# sh set_priority.sh --db session --replSet set07

Set priorities is in progress. check log /var/log/broadhop/scripts/set_priority.log to know the
status
Setting priority for Replica-Set: SESSION-SET2
```

```
INFO Parsing Mongo Config file
INFO Priority set operation is completed for SESSION-SET2
INFO Priority set to the Database members is finished
INFO Validating if Priority is set correctly for Replica-Set: SESSION-SET2
WARNING Mongo Server trying to reconnect while getting config. Attempt #1
INFO Validated Priority is set correctly for Replica-Set: SESSION-SET2
Primary member sessionmgr02:27727 found for Replica SESSION-SET2

Set priorities process successfully completed.

[root@installer ~]#
```

Nota: De forma predeterminada, la prioridad se ha establecido en orden descendente.

Ejecute el comando **diagnostics.sh —get_r** de ClusterManager o pcrfclient para verificar el cambio.

```
| SET NAME - PORT : IP ADDRESS - REPLICA STATE - HOST NAME - HEALTH - LAST SYNC - PRIORITY |
|-------------------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via arbitervip:27727 sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 0 sec - 2 |
| Member-2 - 27727 : 192.168.10.146 - ARBITER - arbitervip - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|-------------------------------------------------------------------------------------------------------
----------------------------------------|
```

Ahora, puede ver que sessionmgr02 se ha convertido en primario, mientras que sessionmgr01 es secundario.