Procédure de gestion du noeud arbitre dans le jeu de réplicas CPS

Table des matières

Introduction

Ce document décrit la procédure à suivre pour gérer le noeud Arbitre dans le jeu de réplicas Cisco Policy Suite (CPS).

Conditions préalables

Exigences

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Linux
- CPS
- base de données mongole

Remarque : Cisco recommande que vous ayez un accès racine privilégié à l'interface de ligne de commande CPS.

Composants utilisés

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- CPS 20.2
- Système d'informatique unifiée (UCS)-B
- MongoDB v3.6.17 et v3.4.16

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 votre réseau est en ligne, assurez-vous de bien comprendre l'incidence possible des commandes.

Informations générales

CPS utilise MongoDB pour constituer sa structure de base de données (DB). Il possède plusieurs jeux de réplicas à des fins diverses : ADMIN, SPR (Subscriber Profile Repository), BALANCE, SESSION, REPORTING et AUDIT.

Un jeu de réplicas dans MongoDB est un groupe de processus mongods qui gèrent le même jeu de données. Les jeux de réplicas assurent la redondance et la haute disponibilité (HA). Avec plusieurs copies de données sur différents serveurs de base de données, il permet des opérations de lecture en partage de charge.

Dans certaines circonstances (par exemple, si vous avez une instance principale et une instance secondaire, mais que les contraintes de coût interdisent l'ajout d'une autre instance secondaire), vous pouvez choisir d'ajouter une instance mongod à un jeu de réplicas en tant qu'arbitre pour voter lors des élections. Un arbitre a exactement 1 vote. Par défaut, un arbitre a la priorité 0.

Les arbitres sont des instances mongod qui font partie d'un jeu de réplicas mais qui ne contiennent pas de données (ce qui signifie qu'elles ne fournissent pas de redondance des données). Ils peuvent toutefois participer aux élections. Un arbitre participe aux sélections pour le primaire, mais il n'a pas de copie de l'ensemble de données et ne peut pas devenir un primaire.

Les arbitres ont des besoins en ressources minimes et ne nécessitent pas de matériel dédié. Vous pouvez déployer un arbitre sur un serveur d'applications ou sur un hôte qui surveille simplement le réseau.

Un arbitre ne stocke pas de données, mais tant que le processus mongod d'arbitre n'est pas ajouté au jeu de réplicas, l'arbitre agit comme n'importe quel autre processus mongod et démarre avec un ensemble de fichiers de données et un journal en taille réelle.

Voici un exemple de jeu de réplicas : **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 |
|-------------------------------------------------------------------------------------------
---------------------------------------|
```

Problème

Supposons qu'il y ait un problème avec un arbitre ou qu'il soit nécessaire de modifier l'arbitre dans un jeu de réplicas, vous devez supprimer l'arbitre actuel et ajouter un nouvel arbitre au jeu de réplicas.

Procédure de gestion de l'arbitre dans un jeu de réplicas

Étape 1. Vérifiez la version du shell mongo dans CPS et le nouvel arbitre. Exécutez cette commande à partir du sessionmgr principal dans le jeu de réplicas et le nouveau noeud arbitre.

Exemple de sortie de sessionmgr :

```
[root@sessionmgr02 ~]# mongo --version
MongoDB shell version v3.6.17
```

Si la version du shell mongo est la même dans sessionmgr principal et dans new arbiter ou si la nouvelle version du shell mongo arbiter est plus élevée, passez à l'étape 6.

Sinon, si la nouvelle version de l'arbitre mongo shell est plus basse, alors vous devez définir **featureCompatibilityVersion** comme valeur inférieure dans la base de données admin du jeu de réplicas avec les étapes suivantes.

Exemple de cas où la nouvelle version de l'interpréteur de commandes arbitre mongo est inférieure à celle de CPS sessionmgr :

```
[root@pcrfclient02 ~]# mongo --version
MongoDB shell version v3.4.16
```

Étape 2. Connectez-vous à l'instance de monologue principale du jeu de réplicas.

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

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

Étape 3. Exécutez cette commande pour afficher le **featureCompatibilityVersion** dans la base de données d'administration du jeu de réplicas.

```
set07:PRIMARY> db.adminCommand( { getParameter: 1, featureCompatibilityVersion: 1 } )
{
"featureCompatibilityVersion" : {
"version" : "3.6"
},
"ok" : 1,
"operationTime" : Timestamp(1663914140, 1),
"$clusterTime" : {
"clusterTime" : Timestamp(1663914140, 1),
"signature" : {
"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
"keyId" : NumberLong(0)
}
}
}
set07:PRIMARY>
```

Étape 4. Exécutez cette commande pour setfeatureCompatibilityVersion comme 3.4 dans la base de données admin du jeu de réplicas.

```
set07:PRIMARY> db.adminCommand( { setFeatureCompatibilityVersion: "3.4" } )
{ "ok" : 1 }
set07:PRIMARY>
```

Étape 5. Exécutez cette commande pour vérifier que **featureCompatibilityVersion** est passé à 3.4 dans la base de données d'administration du jeu de réplicas.

```
set07:PRIMARY> db.adminCommand( { getParameter: 1, featureCompatibilityVersion: 1 } )
{ "featureCompatibilityVersion" : { "version" : "3.4" }, "ok" : 1 }
set07:PRIMARY>
```

Étape 6. Connectez-vous au Gestionnaire de cluster et modifiez le **/var/qps/config/deploy/csv/AdditionalHosts.csv** avec les nouveaux détails de l'arbitre.

```
#vi /var/qps/config/deploy/csv/AdditionalHosts.csv

Provide new arbiter details in this format:

Host Alias IP Address
new-arbiter new-arbiter xx.xx.xx.xx
```

Étape 7. Importez la configuration CSV.

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

Étape 8. Vérifier **/etc/hosts** qui ont été mis à jour avec les informations des nouveaux arbitres.

```
#cat /etc/hosts | grep arbiter
```

Étape 9. Exécuter cette commande pour synchroniser **/etc/hosts**.

```
#/var/qps/bin/update/synchosts.sh

Syncing to following QNS Servers:
lb01 lb02 sessionmgr01 sessionmgr02 qns01 qns02 pcrfclient01 pcrfclient02
Do you want to Proceed? (y/n):y
lb01
lb02
sessionmgr01
sessionmgr02
qns01
qns02
pcrfclient01
pcrfclient02
```

Étape 10. Vérifiez que les scripts mon_db sont arrêtés sur les machines virtuelles pcrfclient.

```
#monsum | grep mon_db_for
```

S'il est arrêté, voici le résultat :

```
mon_db_for_lb_failover Not monitored Program
mon_db_for_callmodel Not monitored Program
```

S'il n'est pas arrêté, voici le résultat :

```
mon_db_for_lb_failover OK Program
mon_db_for_callmodel OK Program
```

> **Remarque** : si les scripts mon_db ne sont pas arrêtés, exécutez ces commandes sur les machines virtuelles pcffclient respectives pour les arrêter manuellement.

```
#monit stop mon_db_for_lb_failover
#monit stop mon_db_for_callmodel
```

Étape 11. Exécutez cette commande à partir de pcrfclient01 pour supprimer l'arbitre actuel du jeu de réplicas (set07 est un exemple dans cette étape).

```
#build_set.sh --session --remove-members --setname set07

Please enter the member details which you going to remove from the replica-set
Member:Port --------> arbitervip:27727
arbitervip:27727
Do you really want to remove [yes(y)/no(n)]: y
```

Étape 12. Exécutez cette commande à partir de Cluster Manager pour vérifier si l'arbitre a été supprimé de **set07**, la sortie de **set07** ne peut pas contenir l'arbitre actuel.

```
#diagnostics.sh --get_replica_status

Expected output:
----------|
|-----------------------------------------------------------------------------------------------------
----------------------------------------|
| SESSION:set07 |
| Status via sessionmgr01:27727 sessionmgr02:27727 |
| Member-1 - 27727 : - SECONDARY - sessionmgr01 - ON-LINE - 0 sec -|
| Member-2 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- -|
|-----------------------------------------------------------------------------------------------------
----------------------------------------|
```

Étape 13. Mettre à jour **mongoConfig.cfg** pour avoir l'arbitre approprié dans le jeu de réplicas modifié. Remplacez l'arbitre actuel (ARBITER=arbiter) par le nouvel arbitre (ARBITER=new-arbiter). Exécutez cette commande à partir du Gestionnaire de cluster.

```
#vi /etc/broadhop/mongoConfig.cfg
```

Configuration actuelle :

```
[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]
```

Configuration requise :

```
[SESSION-SET2]
SETNAME=set07
OPLOG_SIZE=5120
ARBITER=new-arbiter:27727
ARBITER_DATA_PATH=/var/data/sessions.7
MEMBER1=sessionmgr02:27727
MEMBER2=sessionmgr01:27727
DATA_PATH=/var/data/sessions.1/2
[SESSION-SET2-END]
```

Étape 14. Copier la mise à jour **mongoConfig.cfg** à toutes les machines virtuelles. Exécutez cette commande à partir du gestionnaire de cluster.

```
#copytoall.sh /etc/broadhop/mongoConfig.cfg /etc/broadhop/mongoConfig.cfg
```

Étape 15. Ajoutez un nouveau membre arbitre à set07. À partir de Cluster Manager, exécutez

/var/qps/install/current/scripts/build/build_etc.sh afin de créer la commande **/etc/directory**.

Étape 16. Vérifiez que le nouveau membre arbitre est ajouté au jeu de réplicas après avoir exécuté le **build_etc.sh** , vous devez maintenant attendre que le serveur AIDO crée/mette à jour les jeux de réplicas avec le nouvel arbitre.

```
#diagnostics.sh --get_replica_status

Expected Output:

| 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 : xx.xx.xx.xx - ARBITER - new-arbiter - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|----------------------------------------|
```

> **Remarque** : si le nouveau membre arbitre n'est pas ajouté, passez aux étapes suivantes. Sinon, passez à l'étape 18.

Étape 17. Exécutez cette commande à partir du gestionnaire de cluster afin d'ajouter un nouveau membre arbitre de force.

```
#build_set.sh --DB_NAME --add-members --setname Setxxx --force
```

Étape 18. Si le port d'arbitre n'est pas encore actif, exécutez cette commande à partir du nouveau noeud d'arbitre afin de démarrer le même.

```
Command syntax:
#/etc/init.d/sessionmgr-XXXXX start

Sample command:
#/etc/init.d/sessionmgr-27727 start
```

Étape 19. Vérifiez que le nouvel arbitre a bien été ajouté.

```
#diagnostics.sh --get_replica_status
```

Étape 20. Exécutez cette commande à partir du Gestionnaire de cluster pour mettre à jour la priorité de base de données en conséquence.

```
# cd /var/qps/bin/support/mongo/
# ./set_priority.sh --db session
# ./set_priority.sh --db spr
# ./set_priority.sh --db admin
# ./set_priority.sh --db balance
# ./set_priority.sh --db audit
# ./set_priority.sh --db report
```

Étape 21. Exécutez cette commande à partir du Gestionnaire de cluster pour vérifier les modifications apportées au jeu de réplicas.

```
#diagnostics.sh --get_replica_status

Expected Output:

| 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 : xx.xx.xx.xx - ARBITER - new-arbiter - ON-LINE - -------- - 0 |
| Member-3 - 27727 : - PRIMARY - sessionmgr02 - ON-LINE - -------- - 3 |
|----------------------------------------|
```

Étape 22. Vérifiez que les scripts mon_db sont restaurés sur les machines virtuelles pcrfclient. Si ce n'est pas le cas, vous devez les démarrer manuellement.

```
#monsum | grep mon_db_for
```

Afin d'activer le script mon_db, connectez-vous à toutes les machines virtuelles pcrfclient et exécutez ces commandes :

```
# monit start mon_db_for_lb_failover
# monit start mon_db_for_callmodel
```

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.