Ajustez les paramètres pour la représentation optimale CPS à un TPS plus élevé

Contenu

Introduction

Ce document aide à diagnostiquer des problèmes de performances au trafic élevé et à ajuster les paramètres de la suite de stratégie de Cisco (CPS) pour des performances optimales à l'des transactions plus élevées par seconde (TPS).

Diagnostics de problème

1. Analysez les logs de consolider-**engine** pour des codes de résultat de diamètre autres que 2001-DIAMETER_SUCCESS. Exemple :`[root@pcrfclient01 broadhop]#zcat consolidated-engine_07Apr15_16_06_37.1.log.gz| grep "Result-Code" | grep -v 2001|cut -c16-19|sort -u`
3002
5002
5012 Remarque: Cette sortie affiche 3002-DIAMETER_UNABLE_TO_DELIVER, 5002-DIAMETER_UNKNOWN_SESSION_ID et 5012-DIAMETER_UNABLE_TO_COMPLY.Vous pouvez vérifier les détails du code de résultat de diamètre dans [RFC 3588.](#)Pour le CPS qui n'est pas configuré pour des performances optimales, vous trouvez en grande partie la grande quantité pour 5012- DIAMETER_UNABLE_TO_COMPLY.
2. **La consolider-engine d'**examen se connecte pour le compte d'occurrence pour le code 5012 de résultat de diamètre.Exemple :`[root@pcrfclient01 broadhop]#zcat consolidated-engine_07Apr15_23_16_35.1.log.gz| grep "Result-Code" | grep 5012|wc -l`
6643
`[root@pcrfclient01 broadhop]#zcat consolidated-engine_07Apr15_16_06_37.1.log.gz| grep "Result-Code" | grep 5012|wc -l`
627
`[root@pcrfclient01 broadhop]#zcat consolidated-engine_07Apr15_16_26_37.1.log.gz| grep "Result-Code" | grep 5012|wc -l`
2218
`[root@pcrfclient01 broadhop]#zcat consolidated-engine_07Apr15_16_46_35.1.log.gz| grep "Result-Code" | grep 5012|wc -l`
`0`Si on observe le code de résultat de 5012 diamètres à un haut débit à un TPS plus élevé, procédez aux vérifications supplémentaires de log dans cette procédure.
3. Vérifiez dans le log de consolider-**engine** que le « délai d'attente de connection wait après que 0 ms » erreur soit observés avant la stratégie et la fonction de remplissage de règles

(PCRF) envoie le DiameterResponseMessage avec le Résultat-code : 5012. Exemple :<snip>

```
INFO : (balance) Error found, rolling back transaction
ERROR  : (core) Error processing policy request: com.mongodb.DBPortPool$Connection
WaitTimeOut: Connection wait timeout after 0 ms
com.mongodb.DBPortPool.get(DBPortPool.java:222)
com.mongodb.DBTCPConnector$MyPort.get(DBTCPConnector.java:413)
com.mongodb.DBTCPConnector.innerCall(DBTCPConnector.java:238)
com.mongodb.DBTCPConnector.call(DBTCPConnector.java:216)
com.mongodb.DBApiLayer$MyCollection.__find(DBApiLayer.java:288)
com.mongodb.DBApiLayer$MyCollection.__find(DBApiLayer.java:273)
com.mongodb.DBCollection.findOne(DBCollection.java:728)
com.mongodb.DBCollection.findOne(DBCollection.java:708)
com.broadhop.balance.impl.dao.impl.MongoBalanceRepository$6.findOne(MongoBalance
Repository.java:375)
<snip>
```

Remarque: Vous pouvez vérifier TPS sur le système CPS au milieu d'un temps problématique avec la commande de **top_qps.sh** qui est disponible dans la version 5.5 et ultérieures CPS.

Solution

1. Changez la configuration de filetage dans le builder de stratégie du par défaut 20 à **50**. Afin de faire ceci, ouvrez une session au builder de stratégie et choisissez les **données de référence >** les **systèmes >** le **system-1 >** des **configurations embrochables >**Threading des **configurations**. Par défaut (quand le champ de filetage de configuration est blanc) le nombre de thread pour la connexion de mongo est 20 dans la configuration de builder de stratégie ainsi il peut traiter cette quantité de demandes quand il fonctionne sur le bas TPS. À mesure que le TPS augmente ces thread est occupé et par conséquent plus de thread sont exigés afin d'accomplir les demandes.Le compte de thread de 50 est suffisant afin de manipuler environ 5000 TPS car plus de thread sont disponibles que puisse traiter un nombre supérieur de demandes.Ce sont des thread d'engine de stratégie et sont définis avec le nom « ordonne " et devrait être configurés avec ce nom seulement.
2. Ajoutez **Dmongo.client.thread.maxWaitTime=5000 à** /etc/broadhop/pcrf/qns.conf. Exemple :

```
cat /etc/broadhop/pcrf/qns.conf
QNS_OPTS="
-DbrokerUrl=failover:(tcp://lb01:61616,tcp://lb02:61616)?randomize=false
-DjmsFlowControlHost=lb02
-DjmsFlowControlPort=9045
-Dcc.collectd.ip.primary=pcrfclient01
-Dcc.collectd.port.primary=27017
-Dcc.collectd.ip.secondary=pcrfclient01
-Dcc.collectd.port.secondary=27017
-DudpPrefix=lb
-DudpStartPort=5001
-DudpEndPort=5003
-DqueueHeartbeatIntervalMs=25
-Dcom.broadhop.memcached.ip.local=lbvip02
-Dmongo.client.thread.maxWaitTime=5000
```

?Dmongo.client.thread.maxWaitTime est un temps en quelques millisecondes des attentes d'un thread pour qu'une connexion devienne disponible. Si ce paramètre n'est pas spécifié il considère la valeur par défaut qui est 0 ms. Par conséquent, on observe l'erreur tandis que les tests sont à un TPS plus élevé. L'ajout de ce paramètre dans /etc/broadhop/pcrf/qns.conf augmente le temps où les nouveaux thread attendent la connexion de mongo quand les

tests sont sur une haute TPS.2000 est la valeur recommandée QA et a été testé pour la haute TPS. Pour TPS plus grand que 5000 vous pouvez le configurer à 5000 ms afin d'optimiser la représentation.

3. Ajoutez **-Dspr.mongo.socket.timeout=5000 à** /etc/broadhop/pcrf/qns.conf.Par défaut la valeur est de 60000 secondes milliseconds.(60). Cela prend donc du plus long temps de devenir disponible pour d'autres thread.La configuration recommandée est de 5000 millisecondes (5 secondes) afin de faciliter un délai d'attente plus rapide et permettre à d'autres thread pour traiter rapide.
4. Changez les connexions par valeur d'hôte du par défaut 5 à **20** dans le builder de stratégie. Dans l'ot de commande faites ceci, ouvrez une session au builder de stratégie et choisissez les **données de référence >** les **systèmes >** le **system-1 >** des **configurations > configuration >** des **connexions embrochables d'USuM par hôte**. C'est par nombre de la suite de réseau de Quantum (QNS) de connexions avec le DB de mongo. Ceci signifie que pour 4 QNS, 4*20=80 est le nombre total de connexions.Ceci est exigé pour les mises à jour fréquentes dans le mongodb. Par conséquent, il est recommandé pour être mis à jour en tant que 20 par recommandation QA pour des performances optimales.Configurez également la **préférence lue par DB** comme **SecondaryPreferred** qui signifie que tout le QNS reçoit des données de la base de données secondaire et reçoit seulement des données de primaire quand le DB secondaire est occupé. Ceci aide à optimiser la représentation puisque le DB primaire moins est chargé.
5. Configurez la racine appropriée se connectant de niveau pour le système. Les logs excessifs peuvent bloquer le traitement au niveau QNS et livre. Par conséquent il est recommandé que vous configurez la racine se connectant de niveau à **avertissez** ou des niveaux supérieurs à **/etc/brodhop/logback.xml** et aux fichiers de **/etc/broadhop/controlcenter/logback.xml**.Exemple :

```
[root@pcrfclient01 ~]#cat /etc/broadhop/logback.xml

<snip>
<!-- Configure default Loggers -->
<root level="warn">
<appender-ref ref="FILE" />
<appender-ref ref="SOCKET" />
</root>

</configuration>
```

Changez également ces niveaux se connectants :

```
[root@pcrfclient01 ~]#cat /etc/broadhop/logback.xml

<snip>
<!-- Configure default Loggers -->
<root level="warn">
<appender-ref ref="FILE" />
<appender-ref ref="SOCKET" />
</root>

</configuration>
```

Exemple :

```
[root@pcrfclient01 ~]# cat /etc/broadhop/controlcenter/logback.xml

<snip>

<!-- Configure default Loggers -->
<root level="warn">
<appender-ref ref="FILE" />
</root>

</configuration>
```

Ces modifications doivent être répliquées à travers tous les virtual machine.

Exécutez **syncconfig.sh** et puis exécutez **restartall.sh** (ou **stopall.sh** et puis **startall.sh**) afin d'appliquer toute la ces derniers change.

Avertissement : Exécutez ces changements d'une fenêtre de maintenance seulement.