

# Configurar a porcentagem do limite de fragmentação de banco de dados personalizado no CPS

## Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Componentes Utilizados](#)

[Informações de Apoio](#)

[Configurar](#)

[Configurações](#)

[Abordagem do CPS hospedado no OpenStack](#)

[Abordagem para CPS hospedado no VMWare](#)

[Verificar](#)

[Troubleshoot](#)

## Introduction

Este documento descreve como configurar percentuais de limite de fragmentação de banco de dados personalizados no Cisco Policy Suite (CPS).

## Prerequisites

## Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Linux
- CPS
- MongoDB

**Observação:** a Cisco recomenda que você tenha acesso de usuário 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

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. 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 para constituir sua estrutura básica de banco de dados (DB).

A fragmentação é uma característica do MongoDB. Por meio desse alerta, você monitora proativamente a fragmentação do MongoDB e, assim, evita a possível utilização mais alta de recursos (disco e memória) devido ao MongoDB.

O CPS gera um alarme de Simple Network Management Protocol (SNMP) quando a porcentagem de fragmentação do MongoDB excede um valor especificado.

O `/etc/collectd.d/dbMonitorList.cfg` o arquivo presente nas Máquinas Virtuais (VMs) do sessionmgr contém a lista de bancos de dados e seus respectivos valores percentuais de limite de fragmentação.

## Configurar

### Configurações

### Abordagem do CPS hospedado no OpenStack

Etapa 1. Na VM do Cluster Manager, execute este comando para criar um backup do arquivo de configuração atual.

```
#cp /etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg.bkp
```

Etapa 2. Execute este comando no Gerenciador de Cluster para obter a configuração atual das VMs do sessionmgr (para comparar e validar a pós-alteração).

```
#for host in $(hosts-all.sh | grep 'sessionmgr'); do echo checking in $host; ssh $host "cat
/etc/collectd.d/dbMonitorList.cfg"; done
```

Saída de exemplo:

```
checking in sessionmgr01
session_cache|session|40
sk_cache|secondary_key|40
diameter|endpoints|40
spr|subscriber|40
balance_mgmt|account|40
checking in sessionmgr02
session_cache|session|40
sk_cache|secondary_key|40
diameter|endpoints|40
spr|subscriber|40
balance_mgmt|account|40
```

Etapa 3. Modifique o limite padrão (40) para o valor recomendado (por exemplo; 60). Execute

este comando a partir do Gerenciador de Cluster.

**Observação:** esse comando altera o limite para todos os BDs. Se o requisito for atualizar o limite para um BD específico, atualize o arquivo manualmente.

```
#sed -i 's/40/60/g'  
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg
```

**Etapa 4. Execute este comando para comparar os arquivos no Gerenciador de cluster para validar a alteração.**

```
#diff /etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg  
/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/dbMonitorList.cfg.bkp
```

**Saída de exemplo:**

```
4c4  
<session_cache|session|60  
---  
>session_cache|session|40  
9c9  
<sk_cache|secondary_key|60  
---  
>sk_cache|secondary_key|40  
14c14  
<diameter|endpoints|60  
---  
>diameter|endpoints|40  
19c19  
<spr|subscriber|60  
---  
>spr|subscriber|40  
24c24  
<balance_mgmt|account|60  
---  
>balance_mgmt|account|40
```

**Etapa 5. Execute este comando para criar a alteração no Cluster Manager.**

```
[root@installer ~]# /var/qps/bin/build/build_puppet.sh  
Copying /etc/puppet to /var/qps/images/puppet.tar.gz...  
Creating MD5 Checksum...  
[root@installer ~]#
```

**Etapa 6. Execute este comando no Gerenciador de Cluster para aplicar a alteração nas VMs do sessionmgr.**

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo starting vm-init in  
$host; ssh $host "/etc/init.d/vm-init > /dev/null 2>&1 &"; done  
starting vm-init in sessionmgr01  
starting vm-init in sessionmgr02  
[root@installer ~]#
```

**Passo 7. Aguarde até que o boneco esteja completo. Execute este comando a partir do Gerenciador de Cluster para ver o progresso na configuração do Puppet.**

```
#for host in $(hosts-all.sh | grep 'sessionmgr' | tail -1); do echo checking in $host; ssh $host  
"tail -f /var/log/puppet.log"; done
```

```
2022-11-08 06:32:23 +0000 Service[whisper](provider=cps) (info): whisper will be managed using  
monit.
```

```
2022-11-08 06:32:23 +0000 Service[whisper](provider=cps) (info): whisper will be managed using  
monit.
```

```
2022-11-08 06:32:23 +0000 /Stage[main]/Whisper/Service[whisper] (notice): Triggered 'refresh'  
from 1 event
```

```
2022-11-08 06:32:27 +0000 Stage[main] (info): Unsheduling all events on Stage[main]
```

```
2022-11-08 06:32:28 +0000 Puppet (notice): Applied catalog in 83.52 seconds
```

```
[Tue Nov 08 06:32:30 +0000 2022] * Completed puppet configuration for dcl-sessionmgr02...
```

```
[Tue Nov 08 06:32:30 +0000 2022] - NTP sync started, check the logs in vm-init.log
```

## Abordagem para CPS hospedado no VMWare

Etapa 1. Atualize o `/var/qps/config/deploy/csv/Configuration.csv` arquivo no Gerenciador de Cluster com o nome do banco de dados necessário e seu respectivo valor percentual de limite. O formato para fornecer o valor de porcentagem de limite personalizado é como este (onde XX é o valor numérico da porcentagem...por exemplo; 60).

```
session_cache,XX,  
sk_cache,XX,  
diameter,XX,  
spr,XX,  
balance_mgmt,XX,
```

Configuração de exemplo:

```
session_cache,60,  
sk_cache,60,  
diameter,60,  
spr,60,  
balance_mgmt,60,
```

Etapa 2. Execute estes comandos para atualizar o `/etc/collectd.d/dbMonitorList.cfg` para que ele tenha os novos valores de limite do arquivo `Configuration.csv`:

```
[root@installer ~]# /var/qps/install/current/scripts/import/import_deploy.sh  
Filenames that will be processed
```

```
AdditionalHosts.csv Configuration.csv DBConfigServer.csv Definitions.csv Hosts.csv  
ReplicationSets.csv SessionCache.csv VLANs.csv VMSpecification.csv SecureConfig.csv  
VipProxyConfiguration.csv DSCPConfig.csv CriticalFiles.csv
```

```
The CSV files in /var/qps/config/deploy/csv are converted to json files in  
/var/qps/config/deploy/json..
```

```
build the hosts file to /var/www/html/hosts...
```

```
build the /etc/hosts file from the json configuration... /etc/hosts is backed to /etc/hosts.back  
Skipping backup of '/etc/hosts' -- no changes detected.
```

```
Redis by default disabled -DenableQueueSystem=false in /etc/broadhop/qns.conf
```

```
Removing feature configs moved to core
```

```
Removing ws feature from pb and pcrf feature file
```

```
Building /etc/broadhop...
```

```
Copying to /var/qps/images/etc.tar.gz...
```

```
Creating MD5 Checksum...
```

```
Generating /etc/broadhop/servers.all
```

```
Rebuilding facts for: 'installer' (aka 'installer')
```

```
Creating md5sum for hosts file to validate later
```

```
Rebuilding facts for: 'dcl-lb01' (aka 'lb01')
```

```
Rebuilding facts for: 'dcl-sessionmgr01' (aka 'sessionmgr01')
Rebuilding facts for: 'dcl-lb02' (aka 'lb02')
Rebuilding facts for: 'dcl-qns01' (aka 'qns01')
Rebuilding facts for: 'dcl-qns02' (aka 'qns02')
Rebuilding facts for: 'dcl-pcrfclient01' (aka 'pcrfclient01')
Rebuilding facts for: 'dcl-sessionmgr02' (aka 'sessionmgr02')
Rebuilding facts for: 'dcl-pcrfclient02' (aka 'pcrfclient02')
No file for VipProxyConfiguration found
Copying /etc/puppet to /var/qps/images/puppet.tar.gz...
Creating MD5 Checksum...
[root@installer ~]#
```

**Etapa 3. Execute este comando no Gerenciador de Cluster para aplicar a alteração nas VMs do sessionmgr.**

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo starting vm-init in
$host; ssh $host "/etc/init.d/vm-init > /dev/null 2>&1 &"; done
starting vm-init in sessionmgr01
starting vm-init in sessionmgr02
[root@installer ~]#
```

**Etapa 4. Aguarde até que o boneco esteja completo. Execute este comando a partir do Gerenciador de Cluster para ver o progresso na configuração do Puppet.**

```
#for host in $(hosts-all.sh | grep 'sessionmgr' | tail -1); do echo checking in $host; ssh $host
"tail -f /var/log/puppet.log"; done
```

```
2022-11-08 06:48:34 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:48:34 +0000 Service[whisper](provider=cps) (info): whisper will be managed using
monit.
2022-11-08 06:48:34 +0000 /Stage[main]/Whisper/Service[whisper] (notice): Triggered 'refresh'
from 1 event
2022-11-08 06:48:39 +0000 Stage[main] (info): Unsheduling all events on Stage[main]
2022-11-08 06:48:40 +0000 Puppet (notice): Applied catalog in 93.27 seconds
[Tue Nov 08 06:48:42 +0000 2022] * Completed puppet configuration for dcl-sessionmgr02...
[Tue Nov 08 06:48:42 +0000 2022] - NTP sync started, check the logs in vm-init.log
```

## Verificar

Use esta seção para confirmar se a sua configuração funciona corretamente.

Valide a configuração mais recente nas VMs do sessionmgr e compare-a com a saída da Etapa 2. Execute este comando a partir do Gerenciador de Cluster.

```
[root@installer ~]# for host in $(hosts-all.sh | grep 'sessionmgr'); do echo checking in $host;
ssh $host "cat /etc/collectd.d/dbMonitorList.cfg"; done
checking in sessionmgr01
session_cache|session|60
sk_cache|secondary_key|60
diameter|endpoints|60
spr|subscriber|60
balance_mgmt|account|60
checking in sessionmgr02
session_cache|session|60
sk_cache|secondary_key|60
diameter|endpoints|60
spr|subscriber|60
balance_mgmt|account|60
```

```
[root@installer ~]#
```

## Troubleshoot

Esta seção disponibiliza informações para a solução de problemas de configuração.

Este alerta de fragmentação do MongoDB foi introduzido em 20.1 e não foi medido em versões anteriores. Por padrão, o valor do limite de fragmentação é 40%. Esse valor de limite deve ser alterado com base no tamanho da implantação, nos padrões de tráfego (modelos de chamada) e em outros fatores de padrões de tráfego. Caso contrário, o CPS lançará alertas de violação de limite de fragmentação de banco de dados não desejado.

## 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.