

Dépannage et vérification du fabric à l'aide des commandes de ligne unique

Table des matières

[Introduction](#)

[Conditions préalables](#)

[Exigences](#)

[Composants utilisés](#)

[Informations générales](#)

[Outils utilisés pour récupérer les informations](#)

[Liste de toutes les commandes à une seule ligne](#)

[Obtenez uniquement les ID de noeud des LEAF au sein du fabric :](#)

[Vérifiez s'il existe des réinitialisations d'interface :](#)

[Recherchez l'interface la mieux notée :](#)

[Rechercher toutes les modifications de topologie STP :](#)

[Vérifiez s'il y a des abandons RPF de multidiffusion :](#)

[Vérifiez si Glean reçoit trop de paquets :](#)

[Statistiques d'abandon QoS :](#)

[abandons d'interface:](#)

[Erreurs FCS \(erreurs CRC non estompées\)](#)

[FCS + erreurs CRC bloquées](#)

[Abandons de tampon de sortie](#)

[output errors](#)

[Effacer tous les compteurs d'interface](#)

[Problèmes de session BGP :](#)

[Problèmes de session OSPF :](#)

[Paquets principaux acheminés vers le processeur](#)

[Vérifiez à l'échelle du fabric à partir du point où toute une relation contractuelle spécifique est déployée](#)

[Vérifiez où un boîtier est déjà déployé et obtenez l'epg correspondant](#)

[Utilisation de la mémoire de tous les noeuds du fabric :](#)

[Vérifier les abandons sur la pile \(les paquets d'exception sont pointés\)](#)

[Validation du contrat](#)

Introduction

Ce document décrit différentes façons de détecter un problème global sur le fabric.

Conditions préalables

Exigences

- Cisco recommande une connaissance de l'ACI
- Connaissances de base de bash

Composants utilisés

Ce document n'est pas limité à des versions de matériel et de logiciel spécifiques.

Périphériques utilisés :

- Cisco ACI version 4.2(3)

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

Outils utilisés pour récupérer les informations

- Envoyer à la place : sed « s/<oldword>/<new word>/g » g définit qu'il est fait plusieurs fois.
- Sed pour saisir les lignes du début à la fin : « /<début/,/fin/p> ». Combinez l'option -n (noprnt) avec l'indicateur d'impression /p pour dupliquer la fonctionnalité de grep.
- Sed peut être utilisé plusieurs fois en utilisant « ; » par exemple : serait : sed « s/<ancien mot>/<nouveau mot>/g ; s/<oldword2>/<new word2>/g »
- awk -F '<field_separator>' '{print \$2}' Dans cet exemple spécifique, vous fractionnez la ligne par le FIELD_SEPARATOR défini et imprimez le 2ème bloc délimité. Les deux syntaxes font exactement la même chose.
- awk '{ print \$1, \$2 }' imprime les deux premiers champs de chaque enregistrement d'entrée, avec un espace entre eux.
- tri | uniq Signale les lignes répétées. Utilisation de lignes de préfixe -c par le nombre d'occurrences.
- sort -nrk <column> trie les lignes les plus élevées. -n pour un tri numérique, -k pour une clé afin de pouvoir modifier la colonne et si vous voulez définir le plus bas, vous pouvez supprimer -r
- python -m json.tool Affiche JSON dans un joli format.

Liste de toutes les commandes à une seule ligne

Obtenez uniquement les ID de noeud des LEAF au sein du fabric :

1. Dans une liste :

```
APIC#acidiag fmvread | grep leaf | awk '{print $1}'
```

2. Dans une ligne avec des virgules comme séparateurs :

```
APIC#acidiag fnvread | grep leaf | awk '{print $1}' | sed -z 's/\n/,/g;s/,$/\n/'
```

Vérifiez s'il existe des réinitialisations d'interface :

Les informations sont triées sur les interfaces présentant le plus grand nombre de réinitialisations.

```
APIC#moquery -c ethpm.PhysIf | egrep "dn|lastLinkStChg|resetCtr" | tr -d "\n" | sed "s/dn/\ndn/g;s/last
```

option plus lente

```
APIC#for leaf in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo " -> leaf ID: $leaf "
```

Recherchez l'interface la plus cotée :

Afin de trouver où le plus de trafic est reçu :

Interrogez le fabric pour toutes les interfaces avec un débit de sortie supérieur à une valeur spécifique (b). La valeur de m définit si b est gb, mb ou kb.

Pour filtrer sur gb définissez m à 125000000. Pour filtrer sur mb définissez m à 125000. Pour filtrer sur kb définissez m à 125.

```
APIC#bash
```

```
APIC#b=1; m=125000;b=$((b*m)); printf "%-65s %25s\n", "Node/Interface" "Bits/Second"; icurl 'http://loc
```

```
APIC#for leaf in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo " -> leaf ID: $leaf "
```

Rechercher toutes les modifications de topologie STP :

1. La commande va dans chaque leaf et vérifie s'il y a des modifications récentes de la topologie et dans quelles interfaces :

```
APIC#for node in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo "node ID: $node "; fab
```

2. La commande va dans chaque leaf et vérifie le nombre le plus élevé de modifications PVRSTP et quelles interfaces sont vues :

```
APIC#for node in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo "node ID: $node "; fab
```

Vérifiez s'il y a des abandons RPF de multidiffusion :

Cela doit être pour le leaf individuel à la fois et il vérifie tous les VRF pim activés pour toutes les chutes RPF :

```
APIC#for vrf in `show ip mroute summary vrf all | grep 'IP Multicast Routing Table for VR' | awk '{prin
```

Vérifiez si Glean reçoit trop de paquets :

1. Le fabric ARP glane :

```
APIC#for leaf in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo " -> leaf ID: $leaf "
```

2. Glean ARP Paquets reçus :

```
APIC#for leaf in `acidiag fnvread | grep leaf | awk '{print $1}'`; do echo; echo " -> leaf ID: $leaf "
```

Statistiques d'abandon QoS :

Vérifiez les pertes QoS reçues pour l'ensemble du fabric :

```
APIC#moquery -c qosmIfClass -f 'qosm.IfClass.RxDropPacketsCount!="0"' | egrep "RxDropPacketsCount|dn"
```

Vérifiez les pertes de transmission QoS pour l'ensemble du fabric :

```
APIC#moquery -c qosmIfClass -f 'qosm.IfClass.TxDropPacketsCount!="0"' | egrep "TxDropPacketsCount|dn" |
```

abandons d'interface:

Erreurs FCS (erreurs CRC non estompées)

```
APIC#moquery -c rmonDot3Stats -f 'rmon.Dot3Stats.fCSErrors>="1"' | egrep "dn|fCSErrors"
```

FCS + erreurs CRC bloquées

```
APIC#moquery -c rmonEtherStats -f 'rmon.EtherStats.cRAlignErrors>="1"' | egrep "dn|cRAlignErrors"
```

Abandons de tampon de sortie

```
APIC#moquery -c rmonEgrCounters -f 'rmon.EgrCounters.bufferdropkts>="1"' | egrep "dn|bufferdropkts"
```

output errors

```
APIC#moquery -c rmonIfOut -f 'rmon.IfOut.errors>="1"' | egrep "dn|errors"
```

Effacer tous les compteurs d'interface

Commande permettant d'obtenir la liste des noeuds de fabric

```
APIC# acidiag fmvread | egrep " active" | egrep "leaf|spine" | awk '{print $1}' | sed -e 'H;${x;s/\n/,/;101,102,103,204,205,206,301,1001,1002,2001,2002
```

Commande permettant d'effacer les compteurs de la liste précédente

```
APIC# fabric 101,102,103,204,205,206,301,1001,1002,2001,2002 clear counters interface all
```

Problèmes de session BGP :

Afin de vérifier s'il y a des sessions BGP ayant des problèmes sur le fabric sous-jacent

```
APIC#moquery -c bgpPeerEntry -f 'bgp.PeerEntry.operSt!="established" and bgp.PeerEntry.dn*"overlay-1"
```

Afin de vérifier n'importe quelle session BGP

```
APIC#moquery -c bgpPeerEntry -f 'bgp.PeerEntry.operSt!="established" | egrep "dn|operSt" | tr -d "\n" |
```

Problèmes de session OSPF :

Identifiez les sessions qui ne sont pas à l'état complet.

```
APIC#moquery -c ospfAdjEp -f 'ospf.AdjEp.operSt!="full"' | egrep "dn|peerIp" | tr -d '\n' | sed "s/dn
```

Identifiez les sessions qui oscillent constamment et triez les informations sur le nombre le plus élevé :

```
APIC#moquery -c ospfAdjStats -f 'ospf.AdjStats.stChgCnt!="0"' | egrep "dn|stChgCnt" | tr -d "\n" | tr -
```

Paquets principaux acheminés vers le processeur



Tenez compte des éléments suivants : cette commande débogue 500 paquets. Pour un montant inférieur, modifiez le nombre après -c.

```
1EAF#tcpdump -i kpm_inb -c 500 > /tmp/cpu-dp.txt
```

```
LEAF#cat /tmp/cpu-dp.txt | grep IP | awk '{print $3 , $4 , $5}' | grep -v $HOSTNAME | awk -F ':' '{prin
```

Direct sans créer de fichier flambant neuf :

```
LEAF#tcpdump -i kpm_inb -c 500 | grep IP | awk '{print $3 , $4 , $5}' | grep -v $HOSTNAME | awk -F ':' '{prin
```

Vérifiez à l'échelle du fabric à partir du point où toute une relation contractuelle spécifique est déployée

```
APIC#moquery -c actrlRule -f 'actrl.Rule.sPcTag=="32783" and actrl.Rule.dPcTag=="46" and actrl.Rule.sco
```

Vérifiez où un boîtier est déjà déployé et obtenez l'epg correspondant

```
APIC#moquery -c l2CktEp -f 'l2.CktEp.encap=="vlan-3018"'
```

Utilisation de la mémoire de tous les noeuds du fabric :

```
APIC#bash
```

```
APIC# clear ; echo -e "Node ID\t\tFree Memory\tUsed Memory" ; moquery -c procSysMemHist15min -f 'proc.S
```

Vérifier les abandons sur la pile (les paquets d'exception sont pointés)

Vérifiez les pertes TX. Utilisez la commande avec sort -nk 6 -r :

```
APIC# cat istack_debug | egrep "Protocol:|x_pkts_dropped" | tr -d "\n" | sed "s/Protocol/\nProtocol/g"
```

Validation du contrat

Examiner toutes les relations d'un contrat spécifique. Utilisez le script qui remplace le nom du contrat et du locataire :

APIC# CONTRACT='brc-<contract-name>'

APIC# TN='<tenant>'

#CHECK CONSUMERS

#To get the count of epgs consuming a contract (excluding vzany consumer):

APIC#icurl -g 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=subtree&tar

#To list all epg objects consuming a contract (excluding vzany consumers):

APIC#icurl -g 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=subtree&tar

#To get the count of vzanys consuming a contract:

APIC#icurl 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=children&targe

#To list all vzany objects consuming a contract:

APIC#icurl 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=children&targe

#CHECK PROVIDERS

#To get the count of epgs providing a contract (excluding vzany consumer):

APIC#icurl -g 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=subtree&tar

#To list all epg objects providing a contract (excluding vzany consumers):

APIC#icurl -g 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=subtree&tar

#To get the count of vzanys providing a contract:

APIC#icurl 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=children&targe

#To list all vzany objects providing a contract:

APIC#icurl 'http://localhost:7777/api/node/mo/uni/tn-'\$TN'/'\$CONTRACT'.json?query-target=children&targe

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.