Solucionar problemas e verificar a estrutura com comandos de linha única

Contents

Introdução

Este documento descreve diferentes maneiras de detectar um problema geral na estrutura.

Pré-requisitos

Requisitos

- A Cisco recomenda o conhecimento da ACI
- Conhecimento básico de bash

Componentes Utilizados

Este documento não se restringe a versões de software e hardware específicas.

Dispositivos usados:

- Cisco ACI executando a versão 4.2(3)

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

Ferramentas usadas para capturar as informações

- Enviar para substituto: sed "s/<oldword >/<new wordk>/g" g define que isso é feito mais de uma vez.
- Enviar para capturar linhas do início ao fim: sed "/<begin/,/end/p>". Combine a opção -n (noprint) com o sinalizador /p print para duplicações da funcionalidade do grep.
- O Sed pode ser usado mais de uma vez usando ‘;’, por exemplo: poderia: sed "s/<oldword >/<new wordk>/g; s/<oldword2 >/<new word2>/g"
- awk -F '<separador_de_campo>' '{print $2}' Neste exemplo específico, divida a linha pelo FIELD_SEPARATOR definido e imprima o segundo bloco delimitado. Ambas as sintaxes fazem exatamente a mesma coisa.
- awk '{ print $1, $2 }' imprime os dois primeiros campos de cada registro de entrada, com um espaço entre eles.
- classificar | uniq Informa as linhas repetidas. Usando linhas de prefixo -c pelo número de ocorrências.
- sort -nrk <coluna> classifica as linhas com a maior altura. -n para uma classificação numérica, -k para uma chave para que possamos modificar a coluna e, se você quiser definir o mais baixo, poderá remover -r
- python -m json.tool Mostra o JSON em um formato bonito.

Lista de todos os comandos de linha única

Obtenha apenas as IDs de nó dos LEAFs na estrutura:

1. Em uma lista:

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

2. Em uma linha com vírgulas como separadores:

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

Verifique se há redefinições de interface:

As informações são classificadas nas interfaces com o maior número de redefinições.

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

opção mais lenta

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

Verifique a interface com a classificação mais alta:

Para descobrir onde a maior parte do tráfego é recebida:

Consulte a estrutura para todas as interfaces com throughput de saída sobre o valor específico (b). O valor de m define se b é gb, mb ou kb.

Para filtrar em kb defina m como 125000000. Para filtrar em mb defina m como 125000. Para filtrar em kb defina m como 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  "
```

Localizar todas as alterações de topologia do STP:

1. O comando entra em cada folha e verifica se há alguma alteração recente na topologia e em

que interfaces:

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

2. O comando entra em cada folha e verifica a contagem mais alta em alterações de PVRSTP e quais interfaces são vistas:


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


Verifique se há descartes de RPF multicast:

Isso precisa ser para a folha individual no momento e verifica todos os PIM VRF habilitados para qualquer queda de RPF:

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

Verifique se há muitos pacotes recebendo Glean:

1. Gleans ARP de malha:

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

2. ARP obtém pacotes recebidos:


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


Estatísticas de queda de QoS:

Verifique os descartes recebidos de QoS para toda a estrutura:

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

Verifique os descartes de transmissão de QoS para toda a estrutura:

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

desconexão de interfaces:

Erros de FCS (erros de CRC não estopados)

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

Erros de FCS + CRC interrompidos

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

Quedas de buffer de saída

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

erros de saída

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

Limpar todos os contadores de interface

Comando para obter uma lista de nós de estrutura

```
APIC# acidiag fnvread | 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
```

Comando para limpar os contadores da lista anterior

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

Problemas de sessão BGP:

Para verificar se há sessões BGP com problemas na estrutura subjacente

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

Para verificar qualquer sessão BGP

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

Problemas de sessão do OSPF:

Identifique as sessões que não estão em estado full.

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

Identifique as sessões que oscilam constantemente e classifique as informações na contagem mais alta:

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

Pacotes primários que são apontados para a CPU

⚠ Ter em conta: esse comando depura 500 pacotes. Para uma quantidade menor, modifique o número após -c.

```
lEAF#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
```

Direcionar sem criar um arquivo totalmente novo:

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

Verificar toda a malha a partir do apic onde todo um relacionamento de contrato específico é implantado

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

Verifique onde um encapsulamento já está implantado e obtenha o epg correspondente

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

Utilização de memória de todos os nós na malha:

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

Verificar quedas na pilha (os pacotes de exceção são lançados)

Revise as quedas de TX. Use o comando com sort -nk 6 -r :

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

Validação do contrato

Revisar todos os relacionamentos de um contrato específico. Use o script que substitui o nome do contrato e do locatário:

```
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
```

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.