Probleemoplossing en verificatie van de fabric met opdrachten met één lijn

Inhoud

Inleiding

In dit document worden verschillende manieren beschreven waarop we een algeheel probleem in de stof kunnen detecteren.

Voorwaarden

Vereisten

- Cisco raadt kennis van ACI aan
- Basiskennis van bash

Gebruikte componenten

Dit document is niet beperkt tot specifieke software- en hardware-versies.

Gebruikte apparatuur:

- Cisco ACI actieve versie 4.2(3)

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

Achtergrondinformatie

Gereedschap dat gebruikt wordt om de informatie in te pakken

- Plaatsing in plaats van: Met behulp van "s/<oldword >/<new work>/g" definieert u dat dit meerdere malen gebeurt.
- Stel in dat u regels van begin tot eind invoegen: gebruikt "/<start/,/end/p>". Combineer de optie -n (noprint) met de afdrukvlag /p voor duplicaten van de functionaliteit van grep.
- Sed kan meer dan eenmaal worden gebruikt met behulp van ";" bijvoorbeeld: zou: gebruikt "s/<oldword >/<new wordk>/g; s/<oldword2 >/<nieuw woord2>/g"
- awk -F '<field_separator>' '{print $2}' In dit specifieke voorbeeld splitst u de lijn door de gedefinieerde FIELD_SEPARATOR en drukt u de tweede afgebakende chunk af. Beide syntaxen doen precies hetzelfde.
- awk '{ print $1, $2}' drukt de eerste twee velden van elke input record af, met een spatie ertussen.
- soort | uniq Rapporteert de herhaalde lijnen. Gebruik -c prefixregels door het aantal voorvallen te bepalen.
- sorteren -nrk <column> sorteert de hoogste lijnen. -n voor een numerieke soort, -k voor een sleutel zodat we de kolom kunnen wijzigen en als u het laagst wilt definiëren kunt u verwijderen -r
- python -m json.tool toont JSON in een mooie indeling.

Lijst van alle single line commando's

Alleen de knooppunt-ID’s van de LEAF’s in de stof:

1. In een lijst:

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

2. In de regel met komma's als scheidingstekens:

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

Controleer of er interfaceresets zijn:

De informatie wordt gesorteerd op de interfaces met het hoogste aantal reset.

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

langzamere optie

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

Controleer of de best beoordeelde interface:

Om te weten te komen waar het meeste verkeer wordt ontvangen:

Query the fabric voor alle interfaces met uitvoerdoorvoersnelheid over specifieke waarde (b). De waarde van m bepaalt of b gb, mb of kb is.

Om te filteren op gb stel m in op 125000000. Om te filteren op mb stel m in op 125000. Om te filteren op kb stel m in op 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  "
```

Vind alle wijzigingen in STP-topologie:

1. Het bevel gaat in elk blad en verifieert als er om het even welke recente

topologieveranderingen zijn en waarin interfaces:

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

2. De opdracht gaat naar elk blad en verifieert de hoogste telling op veranderingen PVRSTP en welke interfaces worden gezien:


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


Zorg ervoor dat er multicast RPF-druppels zijn:

Dit moet voor het individuele blad op het ogenblik zijn en het verifieert al toegelaten pim VRF voor om het even welke RPF dalingen:

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

Controleer of er te veel pakketten Glean krijgen:

1. Fabric ARP glanst:

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

2. ARP-glean ontvangen pakketten:

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

Statistieken QoS-drop:

Controleer of QoS druppels voor de gehele stof heeft ontvangen:

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

Controleer of QoS-verzenddruppels voor de gehele stof zijn:

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

Interface Drops:

FCS-fouten (niet-gestompte CRC-fouten)

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

FCS + standaard CRC-fouten

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

Uitgaande bufferdruppels

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

output errors

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

Alle interfacetellers wissen

Opdracht voor een lijst met fabric nodes

```
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
```

Opdracht om tellers voor vorige lijst te wissen

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

BGP-sessieproblemen:

Om te controleren of er BGP-sessies zijn die problemen hebben met de onderliggende stof

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

Om een BGP-sessie te controleren

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

OSPF-sessieproblemen:

Identificeer sessies die niet volledig zijn.

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

Identificeer sessies die constant flappen en sorteer de informatie op de hoogste telling:

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

Primaire pakketten die op de CPU zijn gericht

⚠ Rekening houden met: deze opdracht debuteert 500 pakketten. Voor een lager bedrag kunt u het nummer na -c wijzigen.

```
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
```

Direct zonder een gloednieuw bestand te maken:

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

Controleer de fabric-wide vanaf een apic waar alle een specifieke contractuele relatie is geïmplementeerd

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

Controleer waar een encap reeds wordt opgesteld en krijg overeenkomstige epg

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

Geheugengebruik van alle knooppunten in Fabric:

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

Controleer de druppels op de stapel (uitzonderingspakketten worden bestraft)

Beoordeel TX druppels. Gebruik de opdracht met soort -nk 6 -r :

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

Contractvalidatie

Bekijk alle relaties voor een specifiek contract. Gebruik het script ter vervanging van de naam van het contract en de huurder:

```
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
```

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.