Troubleshoot and Verify the Fabric with Single Line Commands

Contents

Introduction

This document describes different ways we can detect an overall issue on the fabric.

Prerequisites

Requirements

- Cisco recommends knowledge of ACI
- Basic knowledge of bash

Components Used

This document is not restricted to specific software and hardware versions.

Devices used:

- Cisco ACI running 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. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Tools used to grab the information

- Sed to substitute: sed **“s/<oldword >/<new wordk>/g”** g defines that it is done more than once.
- Sed to grab lines from start to end: **sed “/<beginning/,/end/p>”.** Combine the -n (noprint) option with the /p print flag for duplicates of the functionality of grep.
- Sed can used more than once by using ‘;’ for example: would : sed **“s/<oldword >/<new wordk>/g; s/<oldword2 >/<new wordk2>/g”**
- **awk -F '<field_separator>' '{print $2}'** In this specific example, you split the line by the defined FIELD_SEPARATOR and print the 2nd delimited chunk. Both syntaxes do exactly the same thing.
- awk '{ print $1, $2 }' prints the first two fields of each input record, with a space between them.
- **sort | uniq** Reports the repeated lines. Using **-c** prefix lines by the number of occurrences.
- **sort -nrk** <column> sorts the lines highest. -n for a numeric sort, -k for a key so we can modify the column and if you want to define lowest you can remove -r
- **python -m json.tool** Shows JSON in a pretty format.

List of all single line command

Get only the node IDs of the LEAFs within the fabric:

1. In a list:

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

2. In a line with commas as separators:

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

Verify if there are interface resets:

The information is sorted on the interfaces with highest number of resets.

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

slower option

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

Check for the Highest rated interface:

In order to find where the most traffic is received:

Query the fabric for all interfaces with output throughput over specific value (b). The value of m defines whether b is gb, mb, or kb.

To filter on gb set m to 125000000. To filter on mb set m to 125000. To filter on kb set m to 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  "
```

Find all STP topology changes:

1. The command goes into each leaf and verifies if there are any recent topology changes and in which interfaces:

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

2. The command goes into each leaf and verifies the highest count on PVRSTP changes and which interfaces are seen:

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

Ensure if there are multicast RPF drops:

This needs to be for the individual leaf at the time and it verifies all the enabled pim VRF for any RPF drops:

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

Verify if there are too many packets getting Glean:

1. Fabric ARP gleans :

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

2. ARP glean Received packets:

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

QoS drop stats:

Verify QoS received drops for the entire fabric :

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

Verify QoS transmit drops for the entire fabric :

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

Interface Drops:

FCS Errors (non-stomped CRC errors)

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

FCS + Stomped CRC Errors

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

Output Buffer Drops

```
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"
```

Clear all interface counters

Command to get a list of 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
```

Command to clear counters for previous list

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

BGP session issues:

In order to check if there are any BGP sessions having issues on the fabric underlay

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

In order to check any BGP session

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

OSPF session issues:

Identify sessions that are not on full state.

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

Identify sessions that flap constantly and sort the information on the highest count:

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

Primary packets that are punted to the CPU

⚠ Take into account: this command debugs 500 packets. For lower amount, modify the number after -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
```

Direct without creating a brand new file:

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

Check fabric-wide from apic where all a specific contract relationship is deployed

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

Check where an encap is already deployed and get corresponding epg

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

Memory Utilization of All nodes in Fabric:

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

Verify drops on istack (exception packets get punted)

Review TX drops. Use the command with sort -nk 6 -r :

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

Contract Validation

Review all the relationships for a specific contract. Use the script replacing the name of the contract and tenant:

```
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
```