使用单线路命令排除故障并检验交换矩阵

目录

简介

本文档介绍我们检测交换矩阵整体问题的不同方法。

先决条件

要求

- 思科建议了解ACI

- Bash基础知识

使用的组件

本文档不限于特定的软件和硬件版本。

使用的设备：

- 运行版本4.2(3)的思科ACI

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

背景信息

用于获取信息的工具

- 用于替换：sed “s/<oldword >/<new wordk>/g” g定义它不止执行一次。
- 用于从头到尾抓取行：sed“/<beginning/,/end/p>”。将 — n(noprint)选项与/p打印标志相结合，以复制grep的功能。
- Sed可以通过使用“；”多次使用，例如：将：sed /<oldword >/<new wordk>/g;s/<oldword2 >/<new wordk2>/g"
- awk -F '<field_separator>' '{print $2}'在本特定示例中，您按定义的FIELD_SEPARATOR拆分行并打印第2个分隔块。两个语法都执行完全相同的事情。
- awk '{print $1, $2 }'打印每个输入记录的前两个字段，这两个字段之间有一个空格。
- sort | uniq 报告重复的行。使用-c前缀行，按出现的次数。
- sort -nrk <column>将行排序为最高。-n表示数字排序，-k表示键，因此我们可以修改列，如果要定义最低值，可以删除 — r
- python -m json.tool以漂亮的格式显示JSON。

所有单行命令的列表

仅获取交换矩阵内枝叶的节点ID:

1. 在列表中：

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

2. 在逗号作为分隔符的行中：

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

验证是否存在接口重置：

信息在重置次数最高的接口上排序。

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

较慢选项

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

检查最高评分的接口：

要查找接收最多流量的位置：

查询交换矩阵，查找输出吞吐量超过特定值(b)的所有接口。 m的值定义b是gb、mb还是kb。

要按gb过滤，请将m设置为125000000。要按mb过滤，请将m设置为125000。要按kb过滤，请将m设置为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  "
```

查找所有STP拓扑更改：

1. 命令进入每个枝叶并验证最近是否有任何拓扑更改以及哪些接口：

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

2. 该命令进入每个枝叶并验证PVRSTP更改的最大计数以及所看到的接口：

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

确保存在组播RPF丢弃：

这需要在当时用于单个枝叶，并且它验证所有RPF丢弃的所有已启用的pim VRF:

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

验证是否有太多数据包获得Glean:

1. 交换矩阵ARP扫描：

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

2. ARP收集已接收数据包：

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

QoS丢弃统计信息：

验证整个交换矩阵的QoS已接收丢弃：

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

验证整个交换矩阵的QoS传输丢弃：

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

接口丢弃:

FCS错误（非堆叠CRC错误）

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

FCS +堆叠CRC错误

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

输出缓冲区丢弃

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

输出错误

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

清除所有接口计数器

用于获取交换矩阵节点列表的命令

```
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
```

清除上一个列表的计数器的命令

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

BGP会话问题：

为了检查交换矩阵底层上是否有任何BGP会话存在问题

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

要检查任何BGP会话

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

OSPF会话问题：

确定未处于完全状态的会话。

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

识别不断摆动的会话，并按最高计数对信息进行排序：

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

传送到CPU的主数据包

⚠ 考虑以下因素：此命令调试500个数据包。对于较小的金额，请修改 — c后的数字。

```
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
```

直接访问，无需创建全新的文件：

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

从部署所有特定合同关系的apic检查整个交换矩阵

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

检查封装已部署的位置并获取相应的epg

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

交换矩阵中所有节点的内存利用率：

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

检验istack上的丢包（丢弃异常数据包）

检查TX丢弃。使用sort -nk 6 -r命令：

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

合同验证

复查特定合同的所有关系。使用脚本替换合同和租户的名称：

```
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
```

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。