思科ACI交换矩阵中的SNMP故障排除

简介

本文档介绍如何在Cisco ACI for ACI版本5.x及更高版本中配置、验证和排除SNMP故障。它涵盖SNMP策略模型、所需的管理合同、陷阱配置、使用CLI和托管对象(MO)查询的操作验证，以及枝叶/主干交换机和APIC控制器之间最常见故障场景的结构化故障排除工作流程。

背景信息

本文档中的材料摘自ACI中的思科ACI解决方案交付团队内部技术说明SNMP:Tomas de Leon撰写的概述、配置、故障排除和警告/问题，以Cisco APIC系统管理配置指南(版本5.x)和Cisco ACI MIB快速参考指南为补充。

概述

ACI中的SNMP架构

SNMP（简单网络管理协议）是一种基于UDP的协议，用于管理网络管理和监控。在ACI中，SNMP独立运行于每个受管实体上。每个枝叶交换机、主干交换机和APIC控制器都是自己的SNMP代理 — 必须单独轮询或监控每个代理。

ACI支持以下SNMP功能：

- 读取操作(Get、GetNext、BulkGet、Walk)-在枝叶/主干交换机和APIC控制器上受支持。
- 通知(陷阱) — 枝叶/主干交换机和APIC控制器支持的SNMPv1、v2c和v3陷阱。
- SNMPv3 -枝叶/主干交换机和APIC控制器上支持。
- 写入操作（设置）—在任何ACI设备上不受支持。
- IPv6 — 仅支持IPv4上的SNMP。

注意：在APIC集群中，每个APIC提供自身本地的MIB对象。您必须独立轮询每个APIC;没有集群范围的SNMP汇聚。同样，必须单独查询每个枝叶和主干交换机。

APIC上的SNMPD架构

APIC运行snmpd进程，该进程包含两个内部组件：

- Agent — 处理SNMP协议处理和会话管理的开源net-snmp代理（版本5.7.6或更高版本）。
- DME（数据模型引擎） — 与APIC管理信息树(MIT)接口，读取托管对象(MO)并将MO属性转换为SNMP对象格式。SNMP陷阱根据MO上发生的事件和故障生成。

SNMP策略模型和部署链

ACI对SNMP使用策略驱动的模型。SNMP配置抽象为snmpPol托管对象，并且必须在将其部署到任何节点之前与交换矩阵的Pod策略组相关联。完整的部署链是：

1. SNMP策略(`snmpPol`) — 定义管理状态、社区字符串、客户端组策略(ACL)和SNMPv3用户。
2. Pod策略组 — 引用SNMP策略以及其他Pod级别策略（BGP、ISIS、NTP等）。
3. Pod配置文件选择器 — 将Pod策略组应用于交换矩阵Pod。

此外，SNMP陷阱配置要求：

1. SNMP监控目标组(`snmpGroup`) — 定义陷阱目标主机、端口、SNMP版本和社区。
2. 监控源(`snmpSrc`) — 将目标组链接到三个不同的监控策略范围：Fabric Default、Fabric Common Policy和Access Policy Default。

允许UDP端口161（SNMP请求）和UDP端口162（SNMP陷阱）的管理合同对于APIC节点是必需的。枝叶和主干节点还需要正确的iptables规则，配置客户端组策略时会自动编程。

支持的 MIB

APIC支持的MIB包括：

- 实体MIB — PhysicalTable
- Cisco Entity Ext MIB - PhysicalProcessorTable、LEDTable
- Cisco Entity FRU Control MIB — PowerSupplyGroupTable、PowerStatusTable、FanTrayStatusTable、PhysicalTable
- 思科实体传感器MIB - SensorValueTable、SensorThresholdTable
- Cisco Process MIB — CPUTotalTable、ProcessTable、ProcessExtRevTable、ThreadTable

枝叶和主干交换机公开标准NX-OS MIB，包括IF-MIB、IP-MIB、CISCO-CDP-MIB、CISCO-ENTITY-QFP-MIB和完整的CISCO-ENTITY-FRU-CONTROL-MIB套件。

APIC上生成的SNMP陷阱包括：`cefcFRUInserted`、`cefcFRURemoved`、`cefcFanTrayStatusChange`、`cefcModuleStatusChange`、`entSensorThresholdNotification`、`cefcPowerStatusChange`、

cpmCPURisingThreshold、 cpmCPUFallingThreshold。

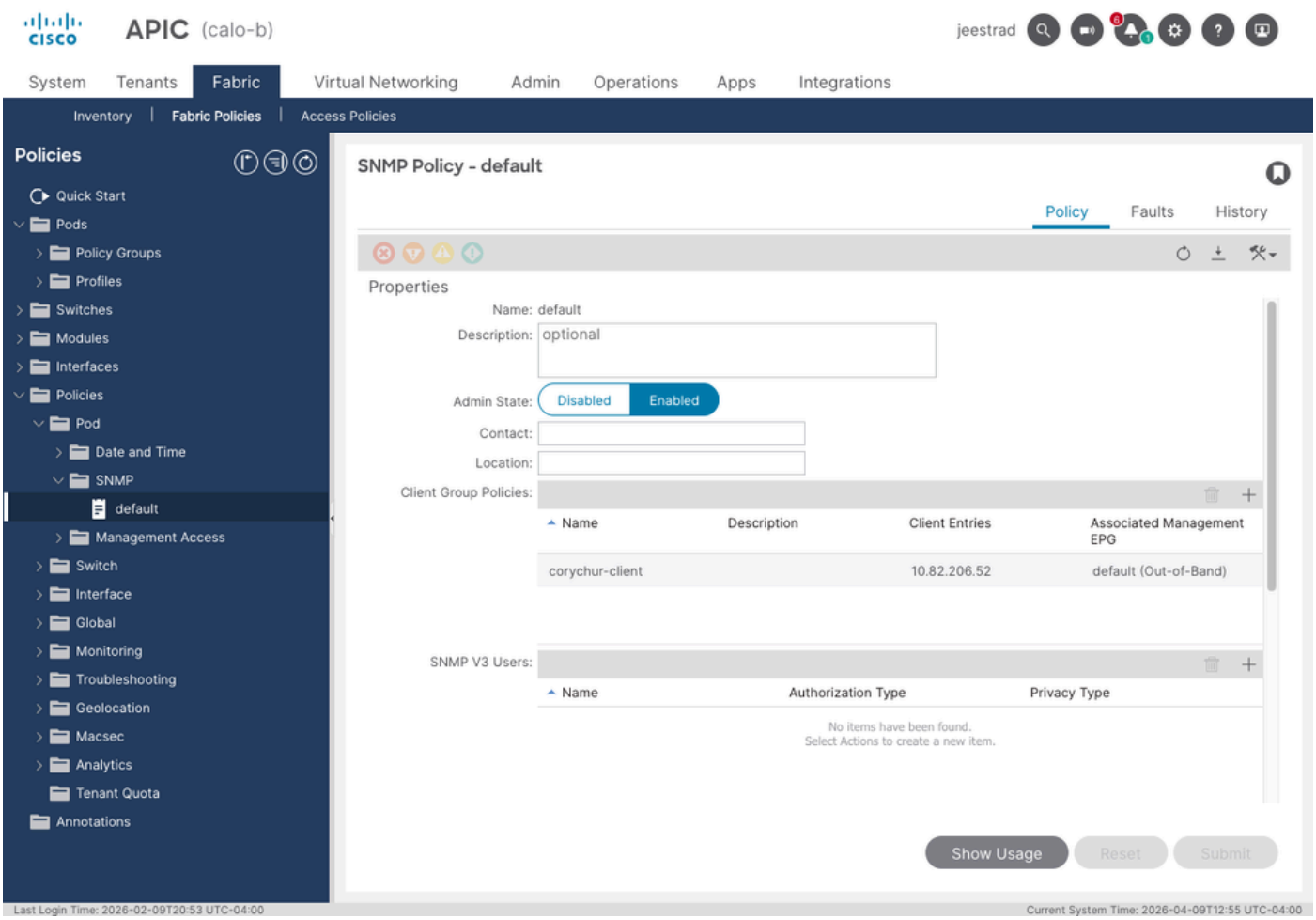
在ACI中配置SNMP

> 注意：本部分提供配置工作流的摘要，作为后续验证和故障排除部分的上下文。请参阅《思科APIC系统管理配置指南》，了解全面的配置步骤。

步骤 1：配置SNMP策略

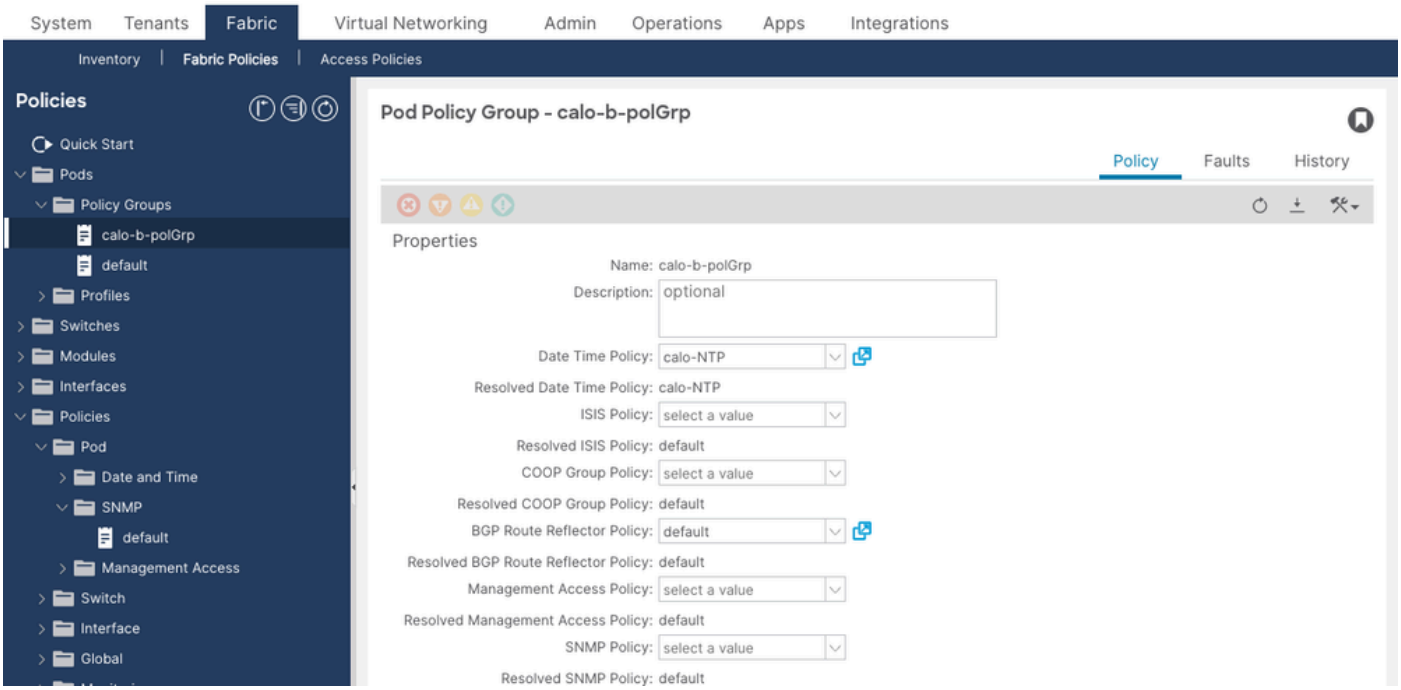
导航到交换矩阵>交换矩阵策略>策略> Pod > SNMP。选择（或创建）SNMP策略，通常命名为default。配置:

- 管理状态 — 设置为启用。
- Community Policies — 添加NMS使用的社区字符串。
- 客户端组策略(Client Group Policies) — 定义一个或多个客户端组配置文件，每个配置文件指定允许的SNMP客户端IP和关联的管理EPG（带外或带内）。
- SNMPv3 Users — 如果使用SNMPv3，请在此处添加具有身份验证和隐私参数的用户。



步骤 2：将SNMP策略与Pod策略组关联

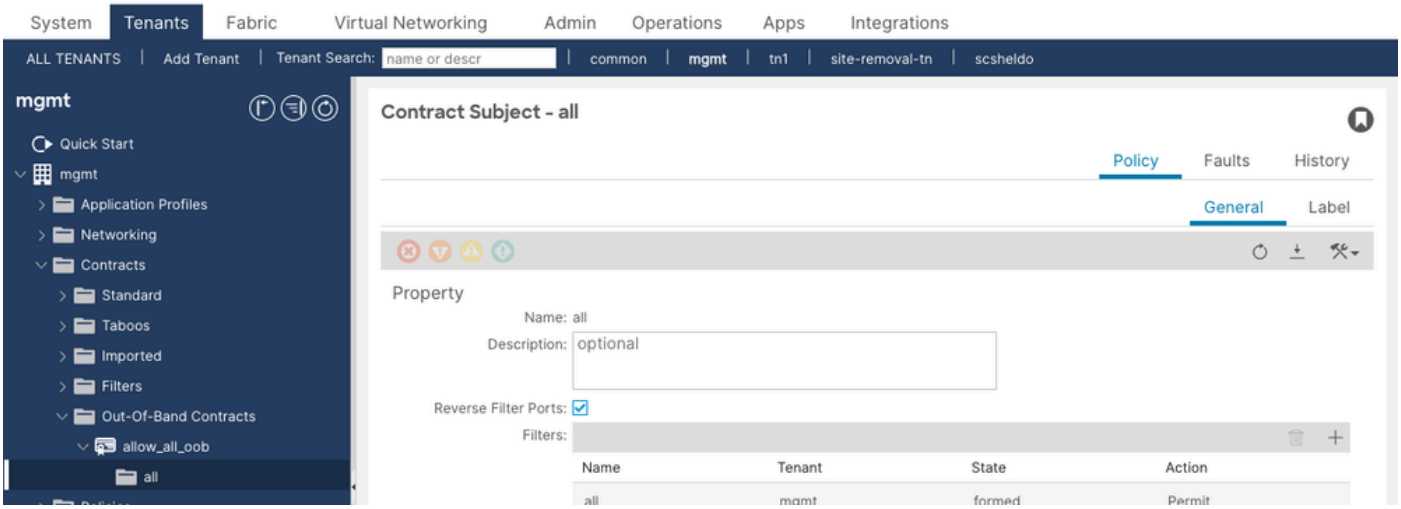
导航到交换矩阵>交换矩阵策略> Pod >策略组。选择活动的Pod策略组(通常命名为default)。 将SNMP Policy字段设置为指向在步骤1中创建的SNMP策略。验证Resolved SNMP Policy字段是否显示正确的策略名称。



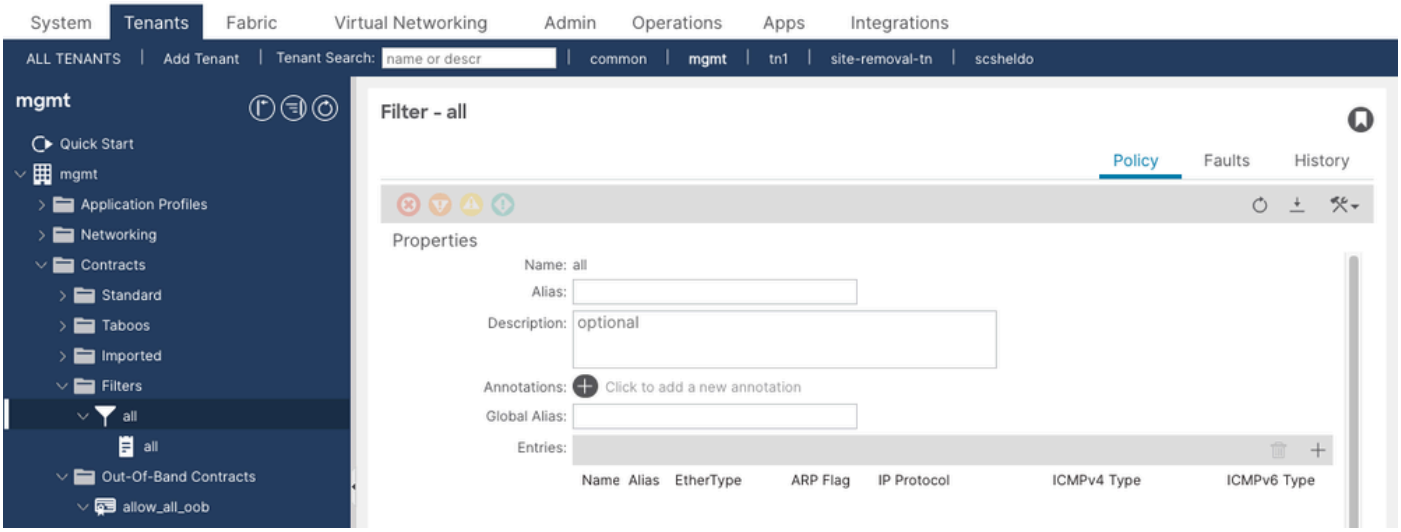
然后导航到交换矩阵>交换矩阵策略> Pod >配置文件，展开默认Pod配置文件，并确认活动选择器引用正确的Pod策略组。

步骤 3：配置UDP端口161的管理合同

导航到租户>管理>合同>带外合同。验证活动OOB合同的主题引用允许UDP端口161(SNMP请求)的过滤器条目。 如果没有APIC上的此合同，所有SNMP GET/WALK数据包将以静默方式丢弃。



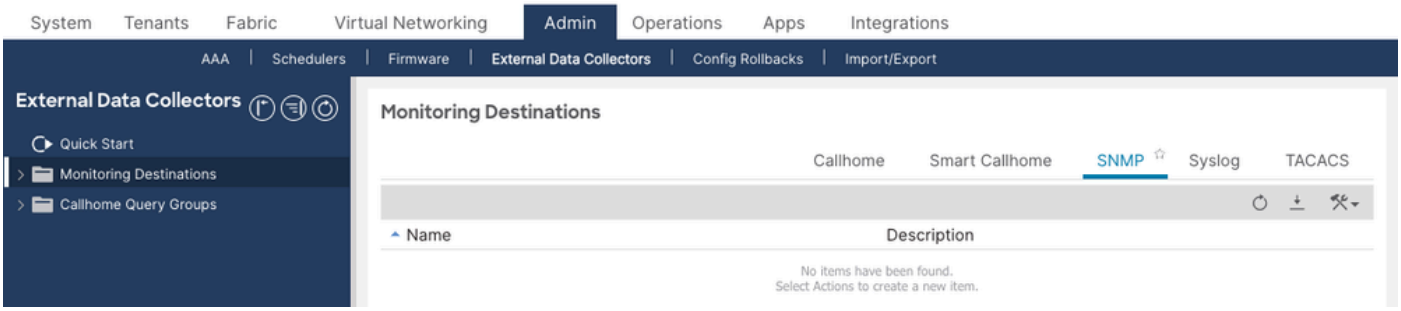
附加到合同主题的过滤器条目必须包含具有EtherType IP、Protocol UDP和Destination Port 161的条目。上面的示例显示了一个允许所有（未指定的协议）过滤器 — 这允许SNMP，但比建议用于生产的过滤器范围更广。首选具有特定UDP/161和UDP/162条目的专用SNMP过滤器条目。



注意：在早期的ACI固件版本中，某些端口在枝叶和主干节点上始终处于打开状态，SNMP不需要管理合同。在ACI 5.x中，APIC节点需要合同。枝叶和主干节点使用从客户端组策略而不是管理合同派生的iptables规则。

步骤 4：配置SNMP陷阱目标

导航到Admin > External Data Collectors > Monitoring Destinations > SNMP。右键单击并选择创建SNMP监控目标组。SNMP选项卡显示所有已配置的目标组。空表表示尚未配置陷阱目标。



定义：

- 组名称
- 陷阱目标:主机名/IP、UDP端口（默认162）、SNMP版本、社区字符串和管理EPG

步骤 5：配置监控源

监控源将SNMP目标组链接到监控策略，这些策略控制哪些事件和故障生成陷阱。必须在以下所有三个位置中配置监控源，否则将不会发送某些节点类型的陷阱：

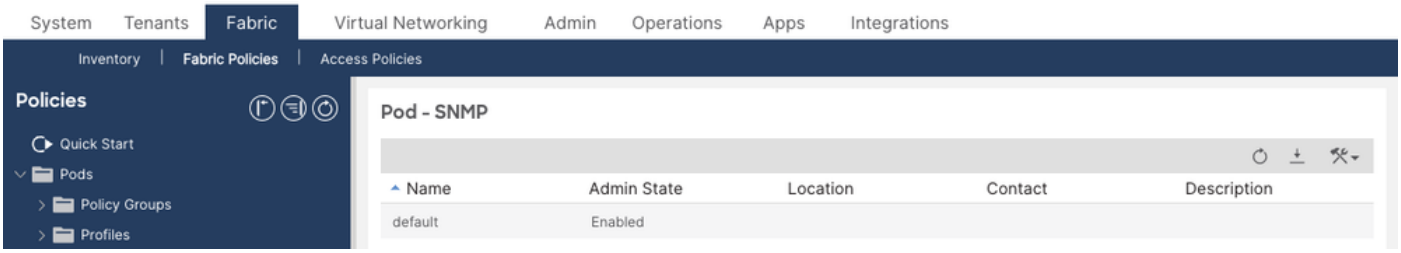
- Fabric > Fabric Policies > Policies > Monitoring > default > Callhome/Smart Callhome/SNMP/Syslog/TACACS(涵盖交换矩阵基础设施事件)
- Fabric > Fabric Policies > Policies > Monitoring > Common Policy > Callhome/Smart Callhome/SNMP/Syslog/TACACS（涵盖交换矩阵范围的常见事件）
- Fabric > Access Policies > Policies > Monitoring > default > Callhome/Smart Callhome/SNMP/Syslog（涵盖接入/基础设施事件）

在每个位置，选择SNMP作为源类型，并创建一个引用步骤4中创建的目标组的新SNMP源。

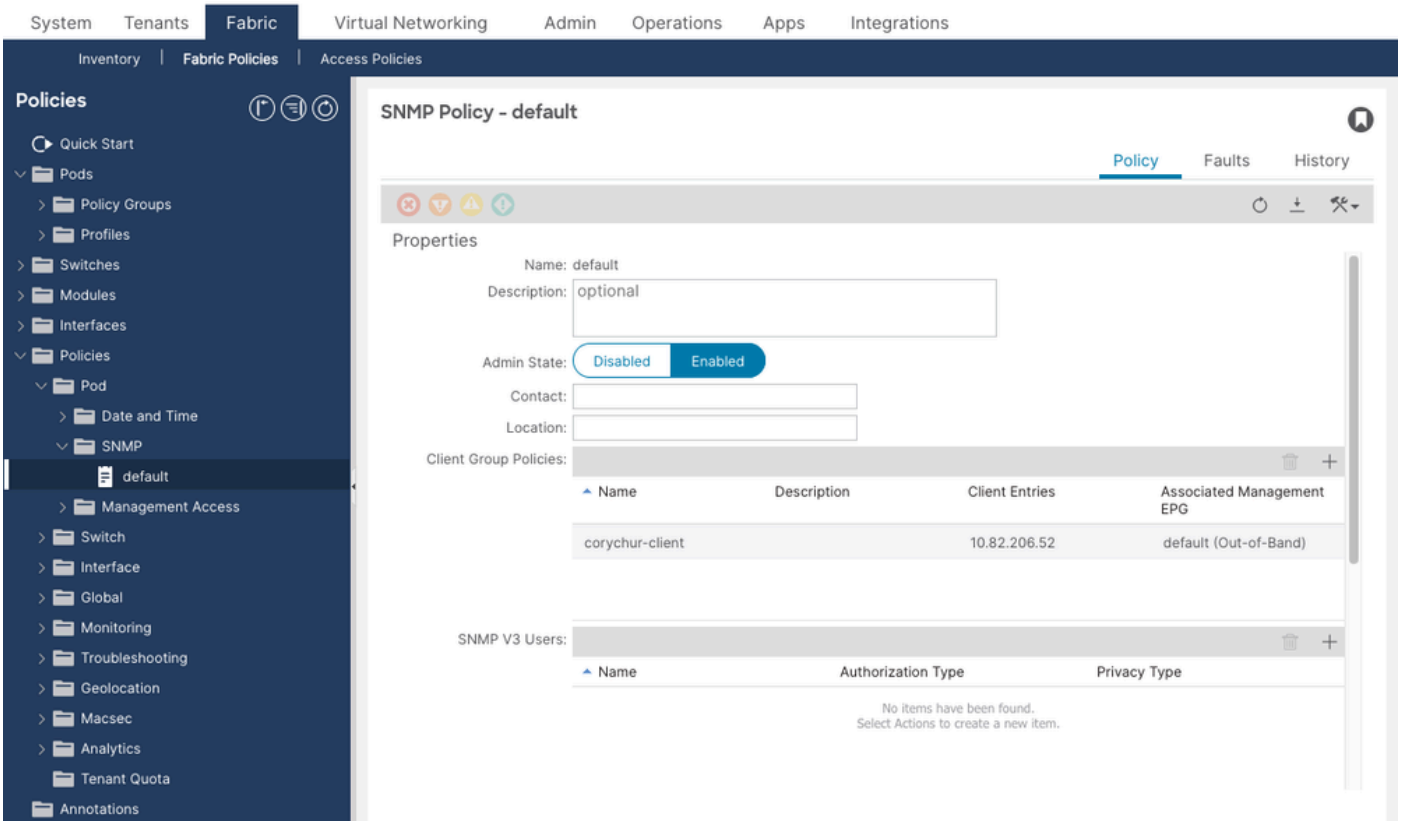
验证配置

验证SNMP策略部署

导航到Fabric > Fabric Policies > Policies > Pod > SNMP，确认默认SNMP策略存在且其Admin State设置为Enabled。Policy Groups列表显示所有已配置的SNMP策略及其管理状态概览。



有关详细验证，请点击策略名称将其打开。确认Admin State（管理状态）切换设置为Enabled，并且Client Group Policies（客户端组策略）列出所有允许的NMS主机及其关联的管理EPG。



在任何APIC上运行以下MO查询，以确认交换矩阵中存在并启用了SNMP策略：

<#root>

```
apic1#

moquery -c snmpPol

Total Objects shown: 1

# snmp.Pol
name          : default
adminSt       : enabled                    <--- must be "enabled"
contact       : NOC Team
descr         : ACI Fabric SNMP Policy
dn            : uni/fabric/snmppol-default
loc           : DC1 ACI Fabric
```

```
monPolDn     : uni/fabric/monfab-default
```

如果adminSt被禁用，则SNMP不会在任何节点上运行。在APIC GUI中的Fabric > Fabric Policies > Policies > Pod > SNMP > default下启用它。

验证社区字符串配置

<#root>

```
apic1#

moquery -c snmpCommunityP

Total Objects shown: 1

# snmp.CommunityP
name         : public                      <--- confirm this matches your NMS community string
dn           : uni/fabric/snmppol-default/community-public
descr        : SNMP Community String
```

如果未返回社区，或者名称与NMS使用的名称不匹配，请在SNMP策略下添加或更正社区字符串。

验证客户端组策略（SNMP访问控制）

客户端组策略用作SNMP GET/WALK访问的ACL。每个策略指定允许哪些客户端IP地址轮询管理VRF的枝叶/主干节点。在枝叶/主干节点上，这些策略被转换为iptables规则。

<#root>

```
apic1#

moquery -c snmpClientGrpP -x query-target=children

Total Objects shown: 3

# snmp.ClientP
addr         : 10.1.1.50                   <--- NMS server IP
dn           : uni/fabric/snmppol-default/clgrp-NMS-Clients/client-[10.1.1.50]
name         : nms-server1

# snmp.ClientP
addr         : 10.1.1.51
dn           : uni/fabric/snmppol-default/clgrp-NMS-Clients/client-[10.1.1.51]
name         : nms-server2

# snmp.ClientGrpP
name         : NMS-Clients
```

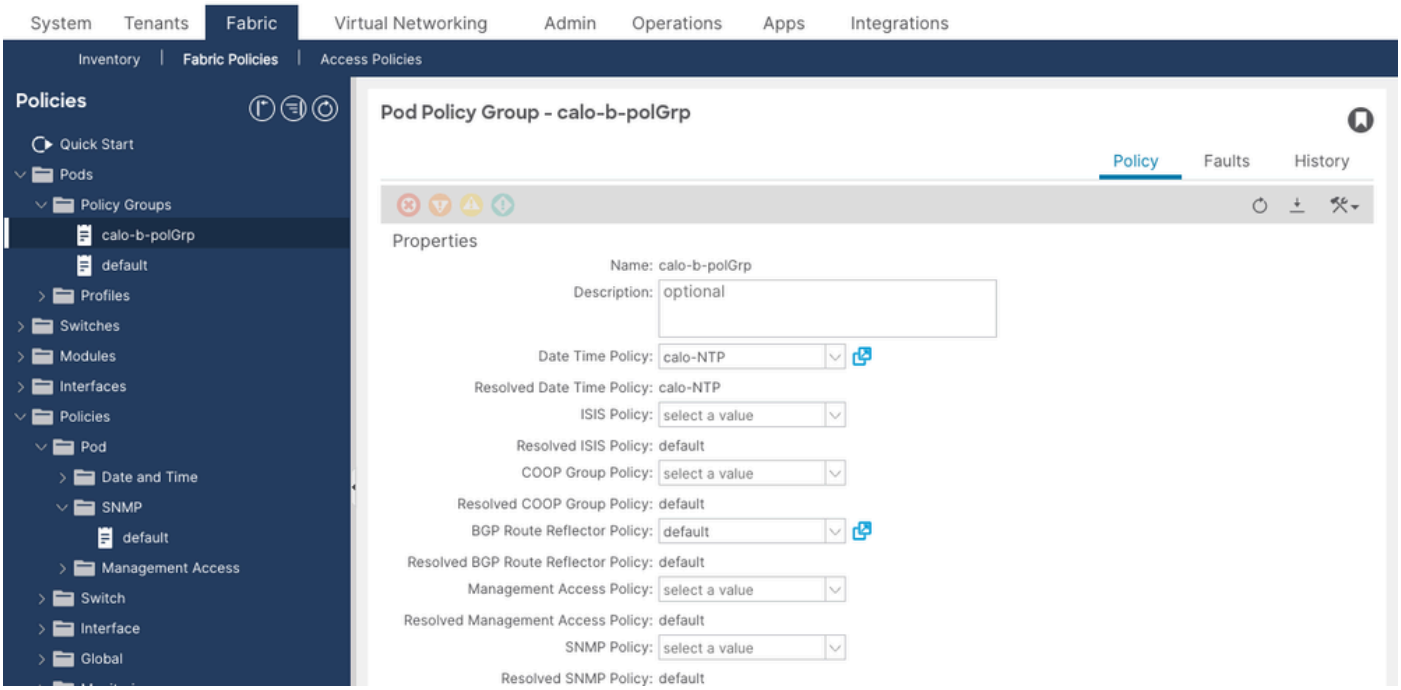
```
dn           : uni/fabric/snmppol-default/clgrp-NMS-Clients
```

确认客户端条目中存在NMS服务器IP。如果客户端IP丢失，则来自该主机的SNMP GET/WALK请求将被枝叶/主干节点上的iptables丢弃。

注意：SNMPv3警告 — 使用SNMPv3时，APIC上不实施客户端组策略。无论客户端组配置如何，都允许任何SNMPv3 GET/WALK到APIC。在APIC上实施SNMPv3的客户端组是已知限制。在枝叶和主干交换机上，SNMPv2c和SNMPv3的客户端组实施行为相同。

验证Pod策略组引用SNMP策略

导航到交换矩阵>交换矩阵策略> Pod >策略组，然后打开活动的Pod策略组。确认SNMP Policy下拉字段设置为所需的SNMP策略，并且Resolved SNMP Policy字段显示相同的名称。缺少或未解析的策略意味着SNMP配置永远不会被推送到交换机。



在上面的屏幕截图中，SNMP Policy字段显示“select a value”（空），而Resolved SNMP Policy显示“default” — 这意味着策略从交换矩阵默认值继承，但未显式设置。建议明确设置SNMP策略字段以避免歧义。

通过REST API验证：

<#root>

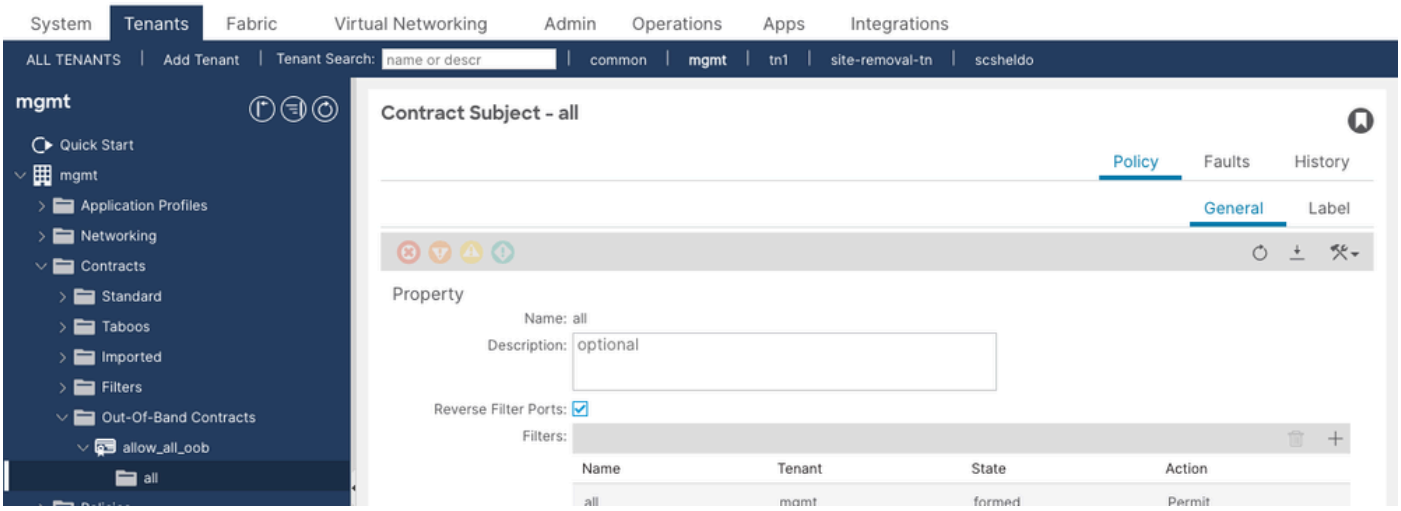
```
apic1#
```

```
moquery -c fabricPodPGrp -x rsp-subtree=full

# fabric.PodPGrp
name          : default
dn            : uni/fabric/funcprof/podpgrp-default

  # fabric.RsSnmpPol
  tnSnmpPolName : default                  <--- must reference the SNMP policy
  state         : formed                   <--- must be "formed"
```

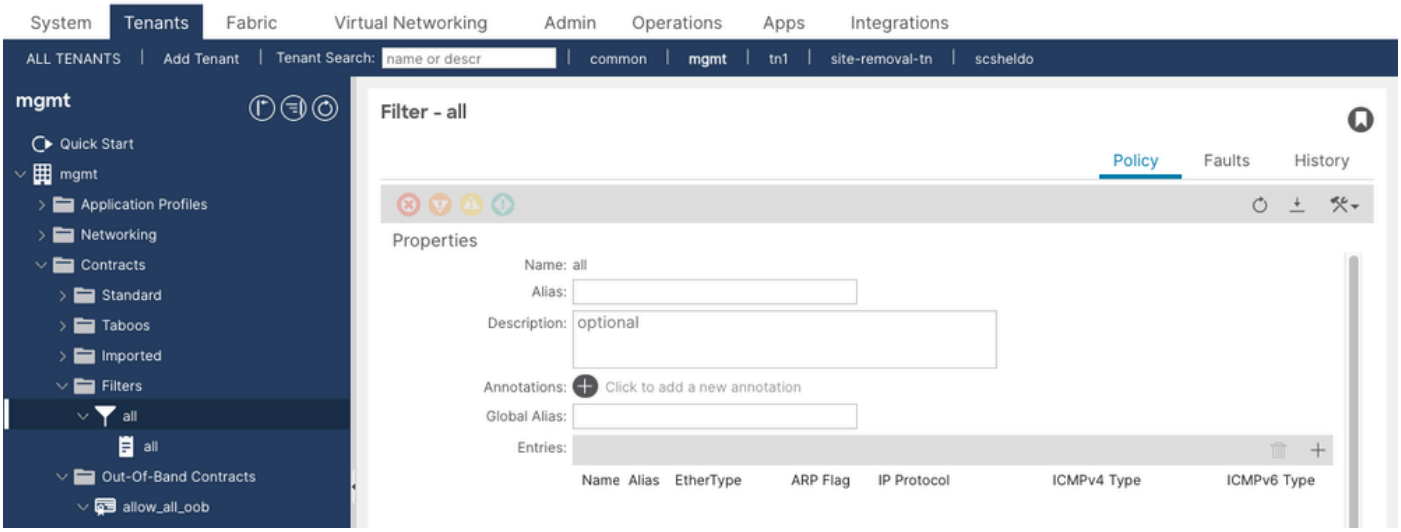
如果state未形成，则SNMP策略关系断开。在Pod策略组中重新选择SNMP策略并提交。

验证UDP 161管理合同（APIC节点）

导航到租户>管理>合同>带外合同（如果使用了INB管理，则导航到带内合同）。 打开活动的OOB合同，然后单击Policy选项卡。验证主题引用了允许UDP端口161的过滤器。



展开主题引用的过滤器，并确认其条目中包含EtherType IP、协议UDP、目标端口161的条目。过滤器条目确定允许哪些流量通过OOB管理合同到达APIC。



过滤器应显示：

- EtherType:IP
- IP 协议:UDP
- 目标端口自：161
- 目标端口到：161

如果希望APIC通过OOB接口出站发送SNMP陷阱，还要验证是否允许UDP端口162。

通过MO查询检查：

<#root>

apic1#

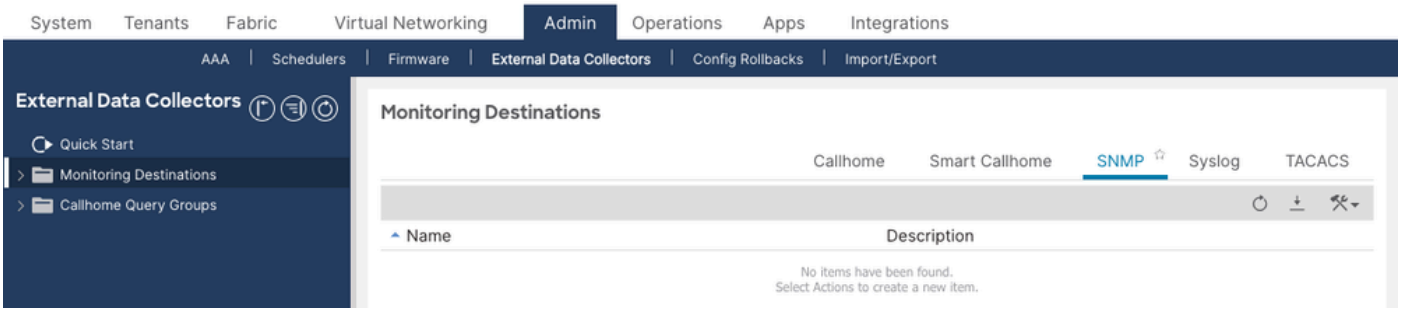
moquery -c vzEntry -x query-target-filter='and(eq(vzEntry.dFromPort,"161"),eq(vzEntry.prot,"17"))'

```
Total Objects shown: 2

# vz.Entry
name         : snmp-get
dn           : uni/tn-mgmt/flt-snmp-filter/e-snmp-get
dFromPort    : 161                        <--- destination port 161
dToPort      : 161
prot         : 17                         <--- UDP
stateful     : no
```

如果未返回任何结果，则不存在UDP 161的过滤器。向管理合同添加一个。

验证SNMP陷阱目标配置

导航到Admin > External Data Collectors > Monitoring Destinations > SNMP以查看所有已配置的SNMP目标组。空列表表示未配置陷阱目标，且不会从任何节点发送陷阱。



<#root>

apic1#

moquery -c snmpTrapDest

Total Objects shown: 1

```
# snmp.TrapDest
host         : 10.1.1.50                  <--- NMS trap receiver IP
port         : 162                         <--- trap UDP port
ver          : v2c                         <--- SNMP version
secName      : public                      <--- community string (v2c) or username (v3)
v3SecLvl     : noauth
notifT       : traps
vrfName      : mgmt:inb                    <--- VRF used to reach the trap receiver
epgDn        : uni/tn-mgmt/mgmtp-default/inb-default
dn           : uni/fabric/snmpgroup-NMS-DestGrp/trapdest-10.1.1.50-port-162
```

确认陷阱目标IP、端口、版本、社区字符串和管理VRF(`mgmt:inb`或`management` for OOB)与您的环境匹配。VRF必须与分配给目标的管理EPG匹配。

验证是否在所有三个范围内配置了监控源

SNMP源必须存在于所有三个监控策略范围内。任何范围中缺少源意味着不会转发相关事件的陷阱。

<#root>

apic1#

moquery -c snmpSrc | egrep "snmp.Src|name|dn|incl|minSev|monPolDn"

```
# snmp.Src
name         : NMS-snmpSrc
```

```
dn           : uni/fabric/monfab-default/snmpsrc-NMS-snmpSrc        <--- Fabric Default
incl         : audits,events,faults
minSev       : info
monPolDn     : uni/fabric/monfab-default

# snmp.Src
name         : NMS-snmpSrc
dn           : uni/fabric/moncommon/snmpsrc-NMS-snmpSrc              <--- Fabric Common
incl         : audits,events,faults
minSev       : info
monPolDn     : uni/fabric/moncommon

# snmp.Src
name         : NMS-snmpSrc
dn           : uni/infra/moninfra-default/snmpsrc-NMS-snmpSrc        <--- Access Default
incl         : audits,events,faults
minSev       : info
monPolDn     : uni/infra/moninfra-default
```

如果缺少这三个源中的任何一个，则使用GUI在相应的监控策略中创建缺少的SNMP源。

操作验证

使用show snmp summary(APIC)检验SNMP状态

直接在每个APIC上运行此命令，以确认SNMP代理正在运行且已应用配置：

<#root>

```
apic1#

show snmp summary


Active Policy:
default, Admin State: enabled             <--- admin state must be "enabled"

Local SNMP engineID: [Hex] 0x8000000980e2b692088976c75600000000

----------------------------------------
Community           Description
----------------------------------------
public              SNMP Community String  <--- community must be present

------------------------------------------------------------
User                Authentication   Privacy
------------------------------------------------------------
                                          <--- empty if using v2c only

------------------------------------------------------------
Client-Group        Mgmt-Epg              Clients
------------------------------------------------------------
NMS-Clients         default (In-Band)     10.1.1.50,10.1.1.51  <--- verify client IPs
```

```
-------------------------------------------------------------
Host                Port    Version   Level       SecName
-------------------------------------------------------------
10.1.1.50           162     v2c       noauth      public        <--- trap destination
```

输出中要验证的内容：

- 必须启用管理状态。
- 社区必须与NMS的配置匹配。
- 客户端组必须列出所有允许的NMS IP和正确的管理EPG。
- 主机（陷阱目标）必须列出具有正确端口和版本的NMS陷阱接收器。

使用show snmp summary（枝叶/主干）检验SNMP状态

<#root>

```
leaf101#

show snmp summary

Admin State : enabled, running (pid:8192)  <--- must show "enabled, running" with a PID

Local SNMP engineID: [Hex] 80000009037C69F6105BF9

-------------------------------------------------------------------------
Community       Context         Status
-------------------------------------------------------------------------
public                          ok                                      <--- community status must be "o

-------------------------------------------------------------------------
Client          VRF             Status
-------------------------------------------------------------------------
10.1.1.50       mgmt:inb        ok                                      <--- client entry must be "ok"
10.1.1.51       mgmt:inb        ok

-------------------------------------------------------------------------------
Host            Port    Ver     Level   SecName         VRF
-------------------------------------------------------------------------------
10.1.1.50       162     v2c     noauth  public          mgmt:inb        <--- trap destination
```

输出中要验证的内容：

- 必须启用管理状态，并使用pid运行。如果显示disabled，则不应用SNMP策略或Pod策略链已断开。
- 社区状态必须正常。错误状态表示存在策略部署问题。
- 每个NMS主机的客户端VRF必须与管理EPG的VRF匹配(mgmt:inb用于带内，管理用于OOB)。
- 陷阱主机必须列出具有正确VRF上下文的目标。

检验snmpd进程是否正在运行

在枝叶或主干上：

```
<#root>
leaf101#
ps aux | grep snmp
root      5881  2.5 1907404  411444 ?    Ssl  Apr05   /isan/bin/snmpd -f -s -d udp:161 udp6:161 tcp:161
leaf101#
pidof snmpd
5881
```

在APIC上：

```
<#root>
apic1#
ps aux | grep snmp
ifc   32182  1.4   0.1  641196  239716  ?   Ssl  Apr10   /mgmt//bin/snmpd.bin \
  -f -p /tmp/snmpd2.pid -a -A -LE 0-2 -c /data//snmp/snmpd.conf
```

如果在枝叶或主干上找不到snmpd进程，则表明该节点上未运行SNMP。检查SNMP策略管理状态是否已启用以及Pod策略链是否已正确配置。

Spoiler（突出显示以便阅读）

验证SNMP端口是否正在侦听

```
<#root>
leaf101#
netstat -lutn | grep 161
Active Internet connections (only servers)
Proto  Recv-Q  Send-Q  Local Address    Foreign Address  State
tcp         0       0  0.0.0.0:161      0.0.0.0:*        LISTEN   <--- SNMP agent is accepting requests
```

```
udp         0       0  0.0.0.0:161      0.0.0.0:*
udp6        0       0  :::161           :::*
```

如果端口161未列在LISTEN状态，则snmpd进程未运行或无法绑定到端口。

验证枝叶/主干上的iptables规则

客户端组策略会转换为每个枝叶和主干上的iptables规则。使用以下命令检查规则：

<#root>

```
leaf101#

iptables -S | grep -i snmp

-N snmp_rules
-N vrf_2_snmp_rules
-N vrf_9_snmp_rules
-A INPUT -p udp -m udp --dport 161 -j snmp_rules    <--- SNMP port 161 redirects to snmp_rules chain
-A snmp_rules -m vrf --vrf 2 -j vrf_2_snmp_rules    <--- VRF 2 = OOB management
-A snmp_rules -m vrf --vrf 9 -j vrf_9_snmp_rules    <--- VRF 9 = In-Band management
-A snmp_rules -j DROP                                <--- default drop; only permitted clients pass
-A vrf_2_snmp_rules -s 10.1.1.50/32 -j ACCEPT       <--- permitted NMS client (OOB VRF)
-A vrf_9_snmp_rules -s 10.1.1.50/32 -j ACCEPT       <--- permitted NMS client (INB VRF)
```

要确定交换矩阵的正确VRF ID，请运行：

<#root>

```
leaf101#

show vrf

VRF-Name                VRF-ID  State   Reason
management              2       Up      --
mgmt:inb                9       Up      --
```

iptables规则中的VRF ID必须与`show vrf`报告匹配。如果iptables规则中没有客户端IP，则该主机的SNMP请求将被静默丢弃，即使snmpd进程正在运行。

使用计数器检查是否匹配或丢弃了任何SNMP数据包：

<#root>

```
leaf101#

iptables -nvL | grep -A 20 "Chain snmp_rules"

Chain snmp_rules (1 references)
 pkts bytes target         prot opt in     out    source       destination
```

```
1    73 vrf_9_snmp_rules all -- *      *       0.0.0.0/0    0.0.0.0/0    vrf 9
0     0 DROP            all -- *      *       0.0.0.0/0    0.0.0.0/0    <--- if pkts>0 here, client
```

注意：如果SNMP正在运行，但iptables未显示snmp_rules链，或者链为空，则可以重新启动snmpd进程以强制iptables规则重新编程。将SIGKILL发送到snmpd PID是安全的 — ACI进程管理器(监管)将自动重新启动它。运行`pidf snmpd`以获取PID，然后`kill -9 [snmpd_pid]`。在10-15秒后，使用`pidof snmpd`确认新的PID。

验证SNMP端口是否正在侦听枝叶101# netstat -lutn | grep 161活动Internet连接（仅服务器）Proto Recv-Q Send-Q本地地址外部地址状态tcp 0 0.0.0.0:161 0.0.0:* LISTEN <— SNMP代理正在接受请求udp 0 0.0.0.0:161 0.0.0:* udp6 0 0 :::161 ::*如果端口161未列在LISTEN状态下，snmpd进程未运行或未能绑定到端口。　　验证枝叶/主干客户端组策略上的iptables规则已转换为每个枝叶和主干上的iptables规则。使用以下命令检查规则：leaf101# iptables -S | grep -i snmp -N snmp_rules -N vrf_2_snmp_rules -N vrf_9_snmp_rules -A INPUT -p udp -m udp —dport 161 -j snmp_rules <— SNMP端口161重定向到snmp_rules链 — A snmp_rules -m vrf 2 -j vrf_2_snmp_rules <— VRF 2 = OOB管理 — A snmp_rules -m vrf —vrf 9 -j vrf_9_snmp_rules <— VRF 9 =带内管理 — A snmp_j DROP < — 默认丢弃；仅允许客户端通过 — A vrf_2_snmp_rules -s 10.1.1.50/32 -j ACCEPT < — 允许的NMS客户端(OOB VRF)-A vrf_9_snmp_rules -s 10.1.1.50/32 -j ACCEPT < — 允许的NMS客户端(INB VRF)要标识交换矩阵的正确VRF ID，请运行：leaf101# show vrf VRF-Name VRF-ID State Reason management 2 Up — mgmt:inb 9 Up — iptables规则中的VRF ID必须与show vrf报告匹配。如果iptables规则中没有客户端IP，则该主机的SNMP请求将被静默丢弃，即使snmpd进程正在运行。使用计数器检查是否匹配或丢弃了任何SNMP数据包：leaf101# iptables -nvL | grep -A 20 "Chain snmp_rules" Chain snmp_rules（1个参考）pkts bytes target prot opt in out source destination 1 73 vrf_9_snmp_rules all — * 0.0.0.0/0 0.0.0.0/0 vrf 9 0 DROP all — * 0.0.0.0/0 0.0.0.0/0 < — 如果此处是pkts>0，则客户端IPs缺失注：如果SNMP正在运行，但iptables未显示snmp_rules链，或者链为空，则可以重新启动snmpd进程以强制iptables规则重新编程。将SIGKILL发送到snmpd PID是安全的 — ACI进程管理器（管制的）将自动重新启动它。运行pidf snmpd以获取PID，然后停用–9 [snmpd_pid]。在10-15秒后使用pidof snmpd确认新的PID。

检验与SNMP端口的网络连接

<#root>

```
leaf101#

netstat -ai | grep eth0

Iface  MTU  Met  RX-OK  RX-ERR  RX-DRP  RX-OVR  TX-OK  TX-ERR  TX-DRP  TX-OVR  Flg
eth0   1500   0  501277      0       0       0  633546      0       0       0  BMRU

leaf101#

netstat -ai | grep kpm_inb

Iface     MTU  Met    RX-OK  RX-ERR  RX-DRP  RX-OVR    TX-OK  TX-ERR  TX-DRP  TX-OVR  Flg
kpm_inb  9300   0  10361421      0       0       0   8958506      0     126       0  BMRU
```

确认管理接口处于活动状态（无RX-ERR增量）且正在传递流量。eth0是OOB管理接口；kpm_inb是交换机上的带内管理接口。

使用tcpdump验证SNMP陷阱发送

要确认陷阱正在从枝叶或主干节点生成和发送，请捕获相应接口上的流量。以管理员身份访问该节点并使用：

<#root>

```
leaf101#

tcpdump -i kpm_inb -f port 162 -vv

tcpdump: listening on kpm_inb, link-type EN10MB (Ethernet), capture size 65535 bytes
17:21:49.810052 IP (tos 0x0, ttl 64, id 63116, proto UDP, length 218)
    172.18.242.14.35582 > 10.1.1.50.snmp-trap: { SNMPv2c C=public
    { V2Trap(171) R=253 system.sysUpTime.0=5888267
      S:1.1.4.1.0=E:cisco.9.276.0.1
      interfaces.ifTable.ifEntry.ifIndex.436224000=436224000
      interfaces.ifTable.ifEntry.ifOperStatus.436224000=2 }}        <--- verify trap is being sent to N
```

对于OOB:

<#root>

```
leaf101#

tcpdump -i eth0 -f port 162 -vv
```

Spoiler（突出显示以便阅读）

对于APIC陷阱(INB):

<#root>

```
apic1#

tcpdump -i bond0.1100 -f port 162

20:01:08.453473 IP apic1-inb.cisco.com.59417 > 10.1.1.50.snmptrap: C=public V2Trap(85) S:
1.1.4.1.0=E:cisco.9.117.2.0.2 E:cisco.9.117.1.1.2.1.1.10548=1 E:cisco.9.117.1.1.2.1.2.10548=2
```

注意：在APIC上，bond0.1100是带内管理接口VLAN子接口。将1100替换为为带内管理EPG配

置的VLAN封装。将oobmgmt用作APIC上OOB捕获的接口名称。

对于APIC陷阱(INB):apic1# tcpdump -i bond0.1100 -f端口162 20:01:08.453473 IP apic1-inb.cisco.com.59417 > 10.1.1.50.snmptrap:C=公共V2Trap(85)S:1.1.4.1.0=E:cisco.9.117.2.0.2 E:cisco.9.117.1.1.2.1.1.10548=1 E:cisco.9.117.1.1.2.1.2.10548=2注：在APIC上，bond0.1100是带内管理接口VLAN子接口。使用为带内管理EPG配置的VLAN封装替换1100。使用oobmgmt作为APIC上OOB捕获的接口名称。

使用tcpdump验证SNMP GET/WALK请求

<#root>

leaf101#

tcpdump -i kpm_inb -f port 161 -vv

```
17:26:08.548149 IP 10.1.1.50.64245 > leaf101.cisco.com.snmp: { SNMPv2c C=public
  { GetRequest(28) R=949769396 system.sysDescr.0 }}              <--- GET request received
17:26:08.552290 IP leaf101.cisco.com.snmp > 10.1.1.50.64245: { SNMPv2c C=public
  { GetResponse(191) R=949769396
    system.sysDescr.0="Cisco NX-OS(tm) aci, Software (aci-n9000-system), \
Version 15.0(1k), RELEASE SOFTWARE" }}                          <--- response returned; SNMP working
```

如果您看到的是GetRequest而没有GetResponse ，则表示请求已收到，但未得到应答。检查snmpd进程和团体字符串。如果既没有看到请求也没有看到响应，则在到达节点之前请求会被阻止（请检查路由和iptables）。

故障排除工作流程

分类决策树

当工程师报告SNMP不工作时，请使用此诊断树。从观察到的症状开始，然后按照分支进行隔离。

症状：没有对SNMP GET/WALK请求的响应

1. 检查APIC上的SNMP管理状态。运行moquery -c snmpPol。如果adminSt已禁用，请启用它并继续执行第7步。
2. 检查snmpd进程。在受影响的节点上，运行ps aux | grep snmp或pidof snmpd。如果没有进程正在运行，则不会部署SNMP策略。验证Pod策略链(SNMP策略→Pod策略组和→配置文件)。
3. 检查端口161是否正在侦听。运行netstat -lutn | grep 161。如果端口161未处于LISTEN状态，snmpd进程已失败；从/var/log/dme/log/svc_ifc_dbgrelem.log*收集日志并重新启动进程。

4. 检查路由。运行`show ip route vrf management`和`show ip route vrf mgmt:inb`。确认指向正确VRF中的NMS主机的路由存在。
5. 检查APIC上的管理合同。如果目标是APIC（不是枝叶/主干），请验证OOB或INB管理合同中是否允许UDP 161。
6. 在节点上执行tcpdump。运行`tcpdump -i kpm_inb -f port 161 -vv`(或`eth0`用于OOB)。 如果显示GetRequest但随后没有GetResponse，则请求到达节点，但snmpd没有响应 — 请检查社区字符串。如果没有显示任何请求，则问题出在上游（路由或合同）。
7. 从允许的客户端进行测试。从客户端组中列出的NMS主机运行`snmpget -v2c -c [community] [node-ip] SNMPv2-MIB::sysDescr.0`。成功响应确认SNMP完全可操作。

症状：NMS未收到SNMP陷阱

1. 检查陷阱目标配置。运行`moquery -c snmpTrapDest`。确认NMS IP、端口、版本和社区与NMS预期值匹配。
2. 检查所有三个作用域中是否存在监控源。运行`moquery -c snmpSrc | egrep "snmp.Src|name|dn"`。 使用`uni/fabric/monfab-default`、`uni/fabric/moncommon`和`uni/infra/moninfra-default`的`monPolDn`值确认条目存在。如果缺少任何SNMP，请在相应的监控策略中添加SNMP源。
3. 检查snmpd进程。验证snmpd是否正在应发送陷阱的节点上运行。
4. 生成测试事件并使用tcpdump捕获。摆动接口或更改状态以生成事件。在节点上，运行`tcpdump -i kpm_inb -f port 162 -vv`。如果线路上未显示陷阱流量，则事件不会生成陷阱 — 重新检查监控源(包括属性)(必须包括`故障`或`事件`)。
5. 检查与陷阱接收器的连接。确认陷阱接收器可从管理VRF到达：`show ip route vrf mgmt:inb`应显示通往NMS主机的路径。
6. 如果陷阱显示在tcpdump上，但NMS上，则问题在于网络侧：防火墙、路由或NMS配置。检查NMS是否正在从ACI节点的管理源IP侦听UDP 162。

常见情况

情形 1：已启用SNMP策略，但未从枝叶/主干返回数据

问题：APIC上的SNMP策略显示已启用Admin State。NMS可以到达枝叶的管理IP。`snmpget`超时且无响应。

配置检查：验证Pod策略组引用SNMP策略，并且Resolved SNMP Policy显示正确的名称。如果Pod策略组的SNMP策略字段为空或未形成关系，则snmpd进程可能无法在交换机上启动。

运行检查：通过SSH连接到受影响的枝叶并运行`show snmp summary`。如果输出显示`Admin State:已禁用`，即使APIC显示已启用，但尚未部署策略。检查Pod策略链中是否存在缺失或引用不当的Pod策略组。

根本原因：SNMP策略未链接到Pod策略组，或者Pod配置文件选择器未将正确的Pod策略组应用于此Pod。

解决方案：

1. 导航到交换矩阵>交换矩阵策略> Pod >策略组>默认。
2. 确认SNMP Policy字段指向已启用的SNMP策略。
3. 导航到交换矩阵>交换矩阵策略> Pod >配置文件，并确认活动选择器引用此Pod策略组。
4. 保存后，在2分钟内重新选中`show snmp summary`在枝叶上。

方案 2：SNMP GET/WALK适用于某些NMS主机，但不适用于其他主机

问题：一台NMS服务器可以成功轮询ACI节点。位于不同子网的第二台NMS服务器未收到响应。

配置检查：在APIC上运行`moquery -c snmpClientGrpP -x query-target=children`。确认第二台NMS服务器的IP已列为客户端条目。如果缺少IP，则该IP将被snmp_rules链底部的iptables DROP规则阻止。

运行检查：在受影响的枝叶上，确认OOB或INB管理合同中允许UDP 161。如果没有合同或过滤器具有SNMP端口，则请求将被丢弃。

根本原因：第二个NMS服务器IP不在客户端组策略中。

解决方案：在Fabric > Fabric Policies > Policies > Pod > SNMP > Default > Client Group Policies下的SNMP Client Group Policy中添加缺失的NMS IP作为客户端条目。所有节点上的iptables规则将在保存策略后的几分钟内更新。

情形 3：未收到SNMP陷阱 — 陷阱已生成但未交付

问题：故障在APIC故障表中可见。`moquery -c snmpTrapDest`显示正确的NMS IP。NMS未收到陷阱。

配置检查：运行`moquery -c snmpSrc | egrep "snmp.Src|name|dn"`。 验证所有三个作用域中是否存在监控源(`monfab-default`、`moncommon`、`moninfra-default`)。一个常见的疏导是仅在交换矩阵默认策略中配置源，这遗漏了访问策略事件。

运行检查：触发测试事件（例如，将接口切换为管理关闭状态）。 在相关节点上，运行`tcpdump -i kpm_inb -f port 162`。如果陷阱数据包出现在节点的接口上，则ACI端工作正常，问题出在通向NMS（防火墙、路由）的网络路径上。 如果线路上未出现陷阱，则缺少ACI监控源，或者事件类型未包含在源的`incl`属性中。

根本原因1:所需范围中缺少一个或多个监控源。

根本原因2:监控源（包括属性）排除生成的事件类型(例如，包括：没有故障的事件意味着不会发送基于故障的陷阱)。

解决方案：

1. 在GUI中为三个范围（交换矩阵默认值、交换矩阵通用和访问默认值）中的每一个范围添加缺失的监控源。 将目标组设置为已配置的SNMP目标组。
2. 验证incl属性包括审计、事件、故障，以确保全面陷阱覆盖范围。
3. 更改后，重新触发测试事件并重新检查tcpdump。

Spoiler（突出显示以便阅读）

注意：在APIC上，tcpdump/code>命令仅对根用户可用。对于APIC和交换机iptables命令仅对根用户可用。

场景 4：SNMPv3客户端组实施在APIC上不起作用

问题：不在客户端组策略中的SNMP客户端可以使用SNMPv3成功查询APIC，即使从枝叶/主干节点进行的相同查询失败也是如此。

根本原因：这是已知**的警告**。客户端组策略（基于iptables的源IP实施）不适用于SNMPv3 GET/Walk到APIC控制器。无论客户端组配置如何，任何主机都可以通过SNMPv3查询APIC。在枝叶和主干交换机上，客户端组实施对SNMPv2c和SNMPv3的工作方式相同。

缓解：在APIC上使用管理合同过滤器按源子网限制SNMP访问。客户端组对枝叶/主干节点有效。对于使用SNMPv3的APIC，依靠管理合同基于源的过滤作为访问控制机制。

场景 5：SNMP查询成功，但MIB数据不完整或过时

问题：SNMP GET/WALK返回数据，但某些MIB OID返回空值或过时的值。特别是，接口统计信息或运行状态数据并不反映当前交换矩阵状态。

运行检查：确认正在查询哪个APIC。每个APIC仅返回其本地数据的MIB对象。在要查询的APIC上运行show snmp summary，并将结果与预期结果进行比较。对于交换机级数据(IF-MIB、entityMIB)，请直接查询交换机，而不是APIC。

根本原因：查询APIC以获取枝叶级MIB数据。每个APIC仅为其自己的托管对象提供MIB对象。必须通过直接轮询每个枝叶和主干来检索交换机级数据（接口统计信息、CPU、内存、环境传感器）。

解决方案：将NMS配置为直接轮询接口和硬件MIB数据的枝叶和主干管理IP。仅对APIC本地MIB（与APIC服务器硬件相关的实体、FRU、进程和传感器）使用APIC管理IP。

场景 6：SNMP适用于枝叶/主干，但不适用于APIC

问题：从NMS到枝叶和主干节点的SNMPv2c GET成功。同一NMS无法轮询APIC。

配置检查：APIC SNMP需要允许UDP 161的显式管理合同。导航到**租户>管理**，并检查OOB/INB合同及其用于UDP 161的过滤器。

运行检查：在APIC上，运行iptables -S | grep 161。如果fp-137（或等效OOB合同）链下未显示UDP 161的ACCEPT规则，则缺少UDP 161的合同过滤器或未部署。

<#root>

apic1#

iptables -S | grep 161

```
-A fp-137 -s 10.0.0.0/8 -p udp -m udp --dport 161 -j ACCEPT     <--- permit SNMP from the management su
-A fp-137 -s 172.18.0.0/16 -p udp -m udp --dport 161 -j ACCEPT  <--- permit SNMP from INB management su
```

如果没有这些规则，请将UDP 161的过滤器条目添加到管理合同主题并重新验证。

根本原因：管理合同缺失或配置错误。在ACI 5.x中，APIC节点严格执行管理合同 — SNMP数据包将被丢弃，除非存在明确许可。

解决方案：

1. 导航到**租户>管理>安全策略>带外合同**。
2. 展开OOB合同，选择主题，然后验证/添加UDP端口**161的过滤器**。
3. 如果NMS通过INB管理到达APIC，请对In-Band合同重复此步骤。
4. 使用iptables -S验证 |保存后,APIC上的grep 161。

场景 7：SNMP iptables规则缺失或不正确

问题：show snmp summary显示已应用SNMP策略，但iptables -S | grep snmp不返回规则，或NMS客户端IP不在规则中。

运行检查：确认snmpd与pidof snmpd一起运行。如果snmpd正在运行，但iptables没有SNMP规则，则进程在部署客户端组策略之前启动。重新启动snmpd可强制在重新启动次数小于250时重新编写规则：

```
<#root>

leaf101#

pidof snmpd

5881

leaf101# show system internal sysmgr service name snmpd
Service "snmpd" ("snmpd", 127):
UUID = 0x1A, PID = 5881, SAP = 1545
State: SRV_STATE_HANDSHAKED (entered at time Mon Aug 25 19:23:50 2025).

Restart count: 3


Time of last restart: Mon Aug 25 19:23:48 2025.
Previous PID: 32080
Reason of last termination: SYSMGR_DEATH_REASON_FAILURE_SIGNAL
Tag = N/A
Plugin ID: 0
leaf101#

kill -9 5881
```

ACI进程管理器将自动重新启动snmpd。重新启动后，验证：

```
<#root>

leaf101#

iptables -S | grep -i snmp
```

此时应显示snmp_rules链和每VRF客户端ACCEPT规则。

根本原因：snmpd进程在客户端组策略完全部署到节点之前重新启动或启动，使iptables没有SNMP访问规则。

注意：在APIC上，tcpdump/code>命令仅对根用户可用。对于APIC和交换机，iptables命令仅对根用户可用。场景4：SNMPv3客户端组实施未处理APIC问题：不在客户端组策略中的SNMP客户端可以使用SNMPv3成功查询APIC，即使从枝叶/主干节点进行的相同查询失败也是如此。根本原因：这是已知警告。客户端组策略（基于iptables的源IP实施）不适用于SNMPv3 GET/Walk到APIC控制器。无论客户端组配置如何，任何主机都可以通过SNMPv3查询APIC。在枝叶和主干交换机上，客户端组实施对SNMPv2c和SNMPv3的工作方式相同。缓解：在APIC上使用管理合同过滤器按源子网限制SNMP访问。客户端组对枝叶/主干节点有效。对于使用SNMPv3的APIC，依靠管理合同基于源的过滤作为访问控制机制。 场景 5：SNMP查询成功，但MIB数据不完整或陈旧问题：SNMP GET/WALK返回数据，但某些MIB OID返回空值或过时的值。特别是，接口统计信息或运行状态数据并不反映当前交换矩阵状态。运行检查：确认正在查询哪个APIC。每个APIC仅返回其本地数据的MIB对象。在要查询的APIC上运行show snmp summary，并将结果与预期结果进行比较。对于交换机级数据(IF-MIB、entityMIB)，请直接查询交换机，而不是APIC。根本原因：查询APIC以获取枝叶级MIB数据。每个APIC仅为其自己的托管对象提供MIB对象。必须通过直接轮询每个枝叶和主干来检索交换机级数据（接口统计信息、CPU、内存、环境传感器）。解决方案：将NMS配置为直接轮询接口和硬件MIB数据的枝叶和主干管理IP。仅对APIC本地MIB（与APIC服务器硬件相关的实体、FRU、进程和传感器）使用APIC管理IP。 场景 6：SNMP适用于枝叶/主干，但不适用于APIC问题：从NMS到枝叶和主干节点的SNMPv2c GET成功。同一NMS无法轮询APIC。配置检查：APIC SNMP需要允许UDP 161的明确管理合同。导航到Tenants > mgmt，并检查UDP 161的OOB/INB合同及其过滤器。运行检查：在APIC上，运行iptables -S | grep 161。如果fp-137（或等效的OOB合同）链下未显示UDP 161的ACCEPT规则，则UDP 161的合同过滤器缺失或未部署。apic1# iptables -S | grep 161 -A fp-137 -s 10.0.0.0/8 -p udp -m udp –dport 161 -j ACCEPT < – 允许来自管理子网的SNMP -A fp-137 -s 172.18.0.0/16 -p udp -m udp –dport 161 -j ACCEPT < – 允许来自INB管理子网的SNMP如果这些规则不存在，请将UDP 161的过滤器条目添加到管理合同主题并重新验证。根本原因：管理合同缺失或配置错误。在ACI 5.x中，APIC节点严格执行管理合同 — SNMP数据包将被丢弃，除非存在明确许可。解决方案：导航到Tenants > Mgmt > Security Policies > Out-Of-Band Contracts。展开OOB合同，选择主题，然后验证/添加UDP端口161的过滤器。如果NMS通过INB管理到达APIC，请对In-Band合同重复此操作。使用iptables -S验证 保存后，APIC上的|grep 161。 场景 7：SNMP iptables规则不存在或不正确的问题：show snmp summary显示SNMP策略已应用，但iptables -S | grep snmp不返回任何规则，或者规则中没有NMS客户端IP。运行检查：确认snmpd与pidof snmpd一起运行。如果snmpd正在运行，但iptables没有SNMP规则，则进程在部署客户端组策略之前启动。重新启动snmpd可强制在重新启动次数小于250时重新编写规则：leaf101# pidf snmpd 5881leaf101# show system internal sysmgr service name snmpdService "snmpd"("snmpd", 127):UUID = 0x1A，PID = 5881,SAP = 1545状态：SRV_STATE_HANDSHAKED（在2025年8月25日星期一19:23:50时输入）。重新启动计数：3上次重新启动时间：2025年8月25日（星期一）19:23:48 2025。上一个PID:32080上次终止原因：SYSMGR_DEATH_REASON_FAILURE_SIGNALTag = N/APlugin ID:0 leaf101# kill -9 5881 ACI进程管理器将自动重新启动snmpd。重新启动后，验证：leaf101# iptables -S | grep -i snmp此时应显示snmp_rules链和每VRF客户端ACCEPT规则。根本原因：snmpd进程在客户端组策略完全部署到节点之前重新启动或启动，使iptables没有SNMP访问规则。

用于扩展故障排除的日志文件

当上述验证步骤无法解决问题时，枝叶、主干和APIC节点上的以下日志文件包含与SNMP相关的诊断信息：

<#root>

leaf101#

zgrep "snmp" /var/log/dme/log/svc_ifc_dbgrelem.log*

leaf101#

zgrep "snmpd" /var/log/dme/log/svc_ifc_dbgrelem.log*

leaf101#

zgrep "snmpd_log" /var/log/dme/log/*

这些日志包含通过show snmp summary不可见的snmpd重新启动事件、策略部署事件以及社区/客户端配置错误。

参考

- 思科APIC系统管理配置指南，版本5.x — 管理SNMP
- Cisco ACI MIB快速参考指南

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）。