使用SCAPY将Nexus 9000配置为流量生成器

目录

简介

本文档介绍Scapy，这是一个用于N9K交换机轻松创建和处理数据包的Python数据包处理工具。

先决条件

将Scapy下载到交换机bootflash。

要下载Scapy，请使用GitHub GitHub-SCAPY[中的链接](#)

要求

Cisco 建议您了解以下主题：

- Nexus 9000/3000交换机。

使用的组件

- N9K-C9396PX

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。

安装

将Scapy代码下载并提取到交换机引导闪存；FTP、SFTP或SCP可用。

启用此功能，在本例中为SCP。

```
switch(config)# feature scp-server
switch(config)# sh feature | i scp
scpServer              1          enabled
```

将文件从笔记本电脑复制到交换机。

```
scp scapy-vxlan-master.zip admin@10.88.164.13:/
```

一旦映像位于引导闪存中，就需要对其进行解压缩。它需要启用bash功能并从bash解压。

```
switch(config)# feature bash
switch(config)# run bash
bash-4.3$ sudo su -
root@switch#cd /bootflash
root@switch#unzip scapy-vxlan-master.zip
```

解压缩后，可使用dir命令在引导闪存中找到文件，即已压缩和未压缩文件。

```
switch# dir bootflash: | i i scapy
       4096    Jul 09 18:00:01 2019  scapy-vxlan-master/
    1134096    Jul 19 23:35:26 2023  scapy-vxlan-master.zip
```

现在有了Scapy。

请注意，您需要使用根权限调用该程序，并且还需要导航到Scapy目录。

```
switch(config)# run bash
Enter configuration commands, one per line. End with CNTL/Z.
```

```
bash-4.2$ sudo su -
root@switch#cd /
root@switch#cd bootflash/scapy-vxlan-master       <<< Move to the scapy folder scapy-vxlan-master
root@switch#python                                <<< Run python once located inside the folder
Python 2.7.2 (default, Mar  9 2015, 15:52:40)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from scapy.all import *                       <<< Import libraries from scapy
>>>
```

创建数据包

本示例说明如何创建基本IP数据包，以说明使用Scapy生成流量的过程。

```
Create l2 source and destination mac addresses.
>>> l2=Ether()
>>> l2.src='00:aa:12:34:12:34'
>>> l2.src='00:ff:aa:bb:cc:11'

Create l3 source and destination IP addresses.
>>> l3=IP()
>>> l3.src='10.1.1.1'
>>> l3.dst='10.2.2.2'
```

另一种功能是从之前捕获的pcap文件发送数据包。这通过命令rdpcap实现。

该命令的输出是一个包含在pcap文件中捕获的所有数据包的Python列表。在本例中，traffic.pcap包含10个数据包，这些数据包将被分配至作为数据包创建的列表。

```
>>> pkts = rdpcap('bootflash/traffic.pcap')
>>> len(pkts)
10
>>> type(pkts)
<class 'scapy.plist.PacketList'>
```

注:pcap文件需要存储在交换机的启动闪存中。

发送流量

创建数据包后，我们使用命令sendp开始通过指定接口发送数据包。

```
>>> packet = l2/l3.                   << packet now have the values for source and destination declared
>>> sendp(packet, iface='Eth1-1').    << Sending the packet through interface eth1/1
.
Sent 1 packets.
```

然后，您可以重复遍历数据包列表，以通过指定的接口发送流量。

```
>>> while True:
...  for i in range(len(pkts)):          <<< It goes through the list pkts with 10 packets and send 1 by
...   sendp(pkts[i], iface='Eth1-1')
...
.
Sent 1 packets.
.
Sent 1 packets.
```

注意：只能使用交换机端口模式访问。否则，它将显示错误。

错误示例：

```
>>> sendp(l2/l3, iface='Eth1-6')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "scapy/sendrecv.py", line 335, in sendp
socket = socket or conf.L2socket(iface=iface, *args, **kargs)
File "scapy/arch/linux.py", line 477, in __init__
set_promisc(self.ins, self.iface)
File "scapy/arch/linux.py", line 165, in set_promisc
mreq = struct.pack("IHH8s", get_if_index(iff), PACKET_MR_PROMISC, 0, b"")
File "scapy/arch/linux.py", line 380, in get_if_index
return int(struct.unpack("I", get_if(iff, SIOCGIFINDEX)[16:20])[0])
File "scapy/arch/common.py", line 59, in get_if
ifreq = ioctl(sck, cmd, struct.pack("16s16x", iff.encode("utf8")))
IOError: [Errno 19] No such device
```

确保接口可用，运行ifconfig命令，必须在该命令中列出接口。

```
bash-4.3$ ifconfig | grep Eth
Eth1-1 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:88
Eth1-2 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:89
Eth1-5 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8c
Eth1-6 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8d
```

```
Eth1-8 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:8f
Eth1-11 Link encap:Ethernet HWaddr 00:a2:ee:74:4b:c1
...
```

验证

您可以使用命令检查任何给定数据包。

```
>>> pkts[5].show()
###[ Ethernet ]###
  dst       = 01:00:0c:cc:cc:cd
  src=58:97:bd:00:a4:f2
  type      = 0x8100
###[ 802.1Q ]###
     prio      = 6
     id        = 0
     vlan      = 104
     type      = 0x32
###[ LLC ]###
        dsap      = 0xaa
        ssap      = 0xaa
        ctrl      = 3
###[ SNAP ]###
           OUI       = 0xc
           code      = 0x10b
###[ Spanning Tree Protocol ]###
              proto     = 0
              version   = 2
              bpdutype  = 2
              bpduflags = 60
              rootid    = 32872
              rootmac   = 58:97:bd:00:a4:f1
              pathcost  = 0
              bridgeid  = 32872
              bridgemac = 58:97:bd:00:a4:f1
              portid    = 32769
              age       = 0.0
              maxage    = 20.0
              hellotime = 2.0
              fwddelay  = 15.0
###[ Raw ]###
                 load      = '\x00\x00\x00\x00\x02\x00h'
```

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）。