Konfigurieren des Nexus 9000 als Traffic Generator mit SCAPY

Inhalt

Einleitung

In diesem Dokument wird Scapy beschrieben, ein Python-Paketbearbeitungs-Tool für N9K-Switches, mit dem Pakete einfach erstellt und bearbeitet werden können.

Voraussetzungen

Laden Sie Scapy auf den Switch-Bootflash herunter.

Um Scapy herunterzuladen, verwenden Sie den Link von GitHub [GitHub-SCAPY](#)

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- Nexus 9000/3000-Switch

Verwendete Komponenten

- N9K-C9396PX

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Installation

Laden Sie den Scapy-Code herunter, und extrahieren Sie ihn in Ihren Switch-Boot-Flash. FTP, SFTP oder SCP sind verfügbar.

Aktivieren Sie die Funktion, in diesem Fall SCP.

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

Kopieren Sie die Datei vom Laptop auf den Switch.

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

Sobald sich das Image im Boot-Flash befindet, muss es dekomprimiert werden. Es muss den Feature-Bash aktivieren und aus dem Bash entpacken.

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

Nach der Dekomprimierung können Sie die Dateien mit dem Befehl **dir** auf dem Boot-Flash, der komprimierten und der unkomprimierten Datei finden.

```
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
```

Scapy ist jetzt verfügbar.

Beachten Sie, dass Sie das Programm mit Root-Berechtigungen aufrufen müssen und dass Sie auch zum Scapy-Verzeichnis navigieren müssen.

```
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
>>>
```

Erstellen eines Pakets

Dies ist ein Beispiel dafür, wie ein einfaches IP-Paket erstellt wird, um das Verfahren zum Generieren von Datenverkehr mithilfe von Scapy zu veranschaulichen.

```
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'
```

Eine weitere Möglichkeit besteht darin, ein Paket aus einer zuvor erfassten pcap-Datei zu senden. Dies wird mit dem Befehl **rdpcap** erreicht.

Die Ausgabe dieses Befehls ist eine Python-Liste, die alle Pakete enthält, die in Ihrer pcap-Datei erfasst wurden. In diesem Beispiel enthält **traffic.pcap** 10 Pakete, die der als pkts erstellten Liste zugewiesen werden.

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

Hinweis: Die pcap-Datei muss im Boot-Flash des Switches gespeichert werden.

Datenverkehr senden

Nachdem das Paket erstellt wurde, verwenden wir den Befehl **sendp**, um unser Paket über die angegebene Schnittstelle zu senden.

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

Anschließend können Sie die Liste der Pakete durchlaufen, um den Datenverkehr über die von Ihnen angegebene Schnittstelle zu senden.

```
>>> 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.
```

> **Hinweis**: Es sind nur Switchports-Moduszugriffe zur Verwendung verfügbar. Andernfalls wird ein Fehler angezeigt.

Beispiel für einen Fehler:

```
>>> 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
```

Stellen Sie sicher, dass die Schnittstelle verwendbar ist, führen Sie den Befehl **ifconfig** aus, und die Schnittstelle muss dort aufgeführt sein.

```
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
...
```

Überprüfung

Sie können den Befehl verwenden, um ein beliebiges Paket zu überprüfen.

```
>>> 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'
```

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.