Configuración de Nexus 9000 como generador de tráfico con SCAPY

Contenido

Introducción

Este documento describe Scapy, una herramienta de manipulación de paquetes Python para que los switches N9K creen y manipulen paquetes con facilidad.

Prerequisites

Descargue Scapy a la memoria de inicialización del switch.

Para descargar Scapy, utiliza el enlace de GitHub GitHub-SCAPY

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- Switch Nexus 9000/3000.

Componentes Utilizados

- N9K-C9396PX

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Instalación

Descargue y extraiga el código Scapy en la memoria flash de inicio del switch; hay disponibles FTP, SFTP o SCP.

Active la función, en este caso, SCP.

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

Copie el archivo al switch desde el portátil.

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

Una vez que la imagen está en la memoria flash de inicio, es necesario descomprimirla. Necesita habilitar la función bash y descomprimirla de bash.

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

Una vez descomprimidos, los archivos se pueden ubicar con el comando **dir** en la memoria flash de inicio, comprimida y sin comprimir.

```
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
```

Ahora Scapy está disponible.

Observe que debe llamar al programa con privilegios raíz y que también debe desplazarse al directorio Scapy.

```
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
>>>
```

Crear un paquete

Este es un ejemplo de cómo crear un paquete IP básico para ilustrar el procedimiento para generar tráfico con Scapy.

```
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'
```

Otra capacidad es enviar un paquete desde un archivo pcap previamente capturado. Esto se logra con el comando **rdpcap**.

El resultado de ese comando es una lista de Python que contiene todos los paquetes capturados en su archivo pcap. En este ejemplo, **traffic.pcap** contiene 10 paquetes y esos paquetes se están asignando a la lista creada como pkts.

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

Nota: El archivo pcap debe almacenarse en la memoria flash de inicio del switch.

Enviar tráfico

Una vez que se crea el paquete, utilizamos el comando **sendp** para comenzar a enviar el paquete a través de la interfaz especificada.

```
>>> 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.
```

A continuación, puede recorrer en iteración la lista de paquetes para enviar el tráfico a través de la interfaz que especifique.

```
>>> 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.
```

> **Nota**: Solo se puede utilizar el acceso en modo de puertos de switch. De lo contrario, se mostrará un error.

Ejemplo del error:

```
>>> 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
```

Asegúrese de que la interfaz sea utilizable, ejecute el comando **ifconfig**, la interfaz debe aparecer allí.

```
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
...
```

Verificación

Puede utilizar el comando para verificar cualquier paquete dado.

```
>>> 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'
```

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).