Table of Contents

Scapy

coap, scapy, schc

Scapy es una herramienta interactiva de manipulación de paquetes.

En otras palabras, permite crear un paquete de red al gusto - especificando todos los campos de todas las cabeceras y el payload - para enviarlo por un interfaz cualquiera del sistema.

Para hacer funcionar Scapy en un sistema, primero clonar el repositorio de Scapy:

```
git clone https://github.com/secdev/scapy
```

Una vez clonado, podemos crear tantos scripts como queramos, dentro del directorio `scapy`.

Editamos el archivo `scapy/ejemplo1.py`:

snippet.python

```
from scapy.all import *
from scapy.contrib.coap import CoAP

e = Ether(src="08:00:27:65:65:50", dst="0a:00:27:00:00:00")
i = IPv6(src="fe80::0A00:27FF:FE54:2E4A",
dst="fe80::0800:27ff:fe00:0000")
u = UDP(sport=5683,dport=59355)

c = CoAP(type='ACK', code='2.01 Created', msg_id=0, paymark=b'\xff')

p = e/i/u/c # The packet p is a concatenation of all the above headers.

print("Sending example packet: ");
p.show()

print("Example packet hexdump: ");
hexdump(p)

sendp(p, iface="enp7s0")
```

En `ejemplo1.py` se crea un paquete p CoAP ACK código 2.01 con un MSG_ID de valor 0 y sin payload. Las direcciones IPv6 están especificadas en `i` y las direcciones Ethernet en `e`. Este paquete es lanzado por la interfaz de red `enp7s0`.

NOTA la dirección Ethernet `0a:00:27:00:00:00` corresponde con una dirección ethernet multi-cast a nivel de enlace.

Para ejecutarlo:

snippet.sh

```
sudo python3 ./ejemplo1.py
```

NOTA necesita ejecutarse como sudo por que crea el paquete con detalles de la capa de enlace Ethernet y eso el kernel de linux no deja hacerlo a nivel de usuario normal.

Otros ejemplos del formato para escribir las cabeceras CoAP incluyen:

snippet.python

```
c = CoAP(type='ACK', code='2.01 Created', token='ab', msg_id=0,
options=[('Location-Path', 'storage'),(92, b'\xCA\xFE'), (92,
b'\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\
xAF\xBE\xAF')], paymark=b'\xff')
c = CoAP(type='ACK', code='2.01 Created', msg_id=0,
options=[('Location-Path', 'storage')], paymark=b'\xff')
```

A parte de simplemente enviar un mensaje, se puede crear un servidor customizado que responda a peticiones.

Otro ejemplo

snippet.python

```
from scapy.all import *
from scapy.contrib.coap import CoAP
import datetime
import os # os.urandom()


try:
    while True:
        # We wait for a packet with dst port 5683.
        # {

        print("Listening for Uplink Packet udp and port 5683");
        uplink_rx = sniff(iface='eth1', count=1, filter="udp and port
5683");

        print("Uplink packet received");
        # Print packet
        uplink_rx[0].show();
        print("Uplink packet hexdump");
        hexdump(uplink_rx[0])
        # }

        #
        # Paquete 1
```

```
        # {
        print("Sending downlink packet: ");
        e = Ether(src="08:00:27:65:65:50", dst="0a:00:27:00:00:00")
        i = IPv6(src="fe80::a00:27ff:fe65:6550",
dst="fe80::800:27ff:fe00:0")
        u = UDP(sport=10001,dport=10001)

        # c = CoAP(type='ACK', code='2.01 Created', token='ab',
msg_id=0, options=[('Location-Path', 'storage'),(92, b'\xCA\xFE'), (92,
b'\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\xAF\xBE\
xAF\xBE\xAF')], paymark=b'\xff')
        c = CoAP(type='ACK', code='2.01 Created', token='ab', msg_id=0,
options=[('Location-Path', 'storage')], paymark=b'\xff')

        # d =
b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00'
        p = e/i/u/c
        p.show()
        hexdump(p)
        sendp(p, iface="eth1")
        # }
        #


#
# We catch CTRL+C to stop the script. We need to capture
# the interrupt outside of the loop and then ignore it.
# {
except KeyboardInterrupt:
    pass
# }
```

Este ejemplo espera a la recepción de cualquier paquete que coincida con el filtro `filter="udp and port 5683"` recibido en el interfaz `eth1`.

Cada vez que un paquete de esas características es recibido, lo imprime por pantalla y genera un paquete p que envía también por el interfaz `eth1`.