세션 하이재킹

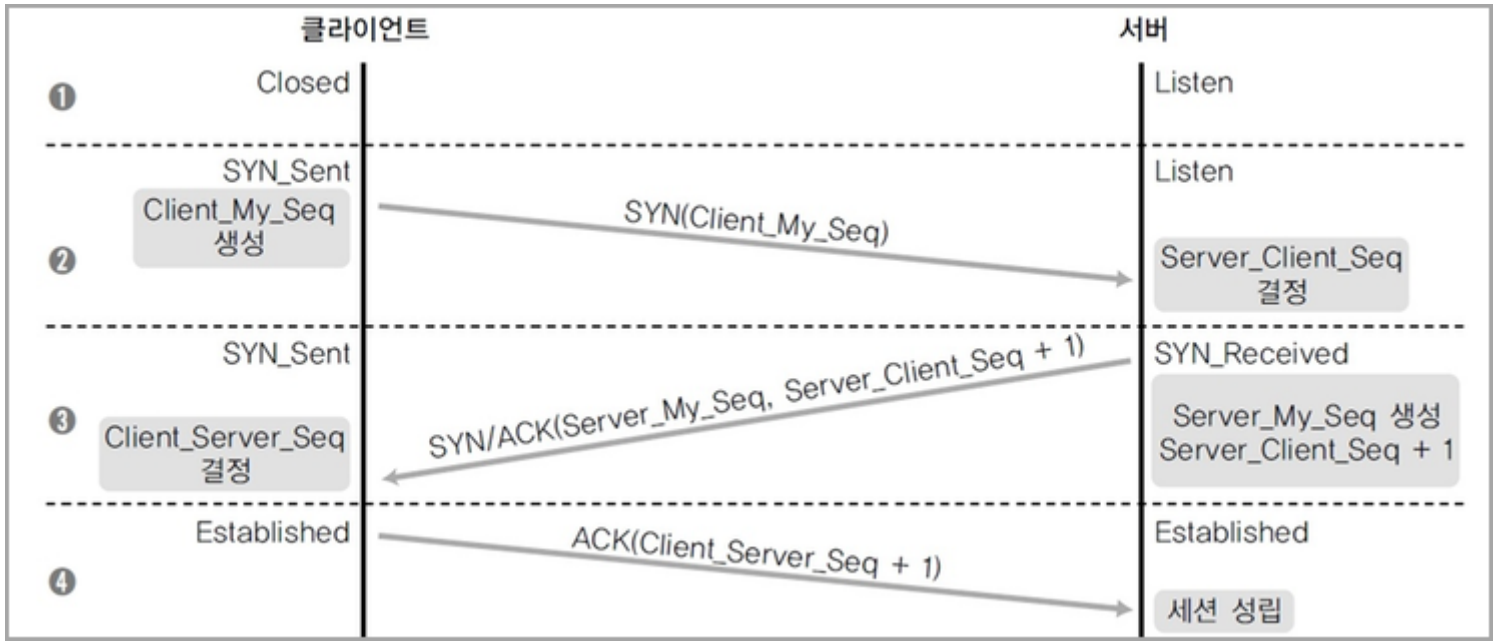
202112021 박채우

TCP

전송 제어 프로토콜은 인터넷 프로토콜에서 핵심 프로토콜 중 하나로 TCP/IP라는 명칭으로 널리 알려져 있다.

해당 프로토콜은 4계층/전송계층에 해당된다.

TCP 프로토콜은 다양한 특징이 있다. 신뢰성, 연결 지향성, 혼잡제어, 흐름제어 ..

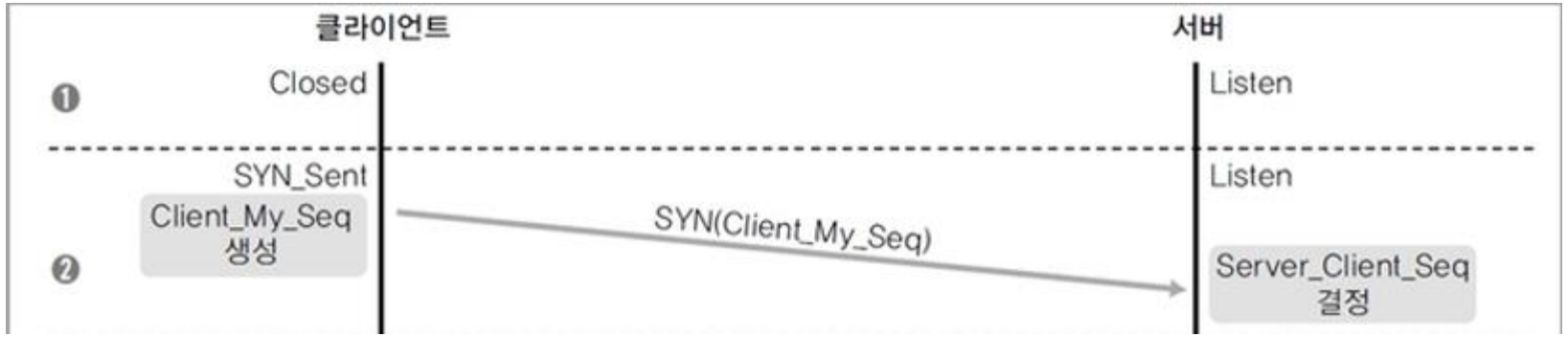

클라이언트
서버
1
Closed
Listen
2
SYN_Sent
Client_My_Seq
생성
SYN(Client_My_Seq)
Listen
Server_Client_Seq
결정
3
SYN_Sent
Client_Server_Seq
결정
SYN/ACK(Server_My_Seq, Server_Client_Seq + 1)
SYN_Received
Server_My_Seq 생성
Server_Client_Seq + 1
4
Established
ACK(Client_Server_Seq + 1)
Established
세션 성립

TCP

TCP는 3-way handshake 과정을 통해 상호 간의 동기화된 시퀀스 번호를 가지고 서버와 클라이언트를 연결한다.

1. 클라이언트는 32비트인 임의의 숫자로 자신의 시퀀스 넘버를 생성 후 서버에게 SYN을 전송한다.

서버는 수신된 SYN(클라이언트의 시퀀스 넘버)를 저장한다.



TCP

2. 서버는 서버 자신의 시퀀스 넘버를 생성 후 저장한 클라이언트의 시퀀스 넘버에 패킷 길이를 더하여 클라이언트에게 SYN/ACK 를 전송한다.

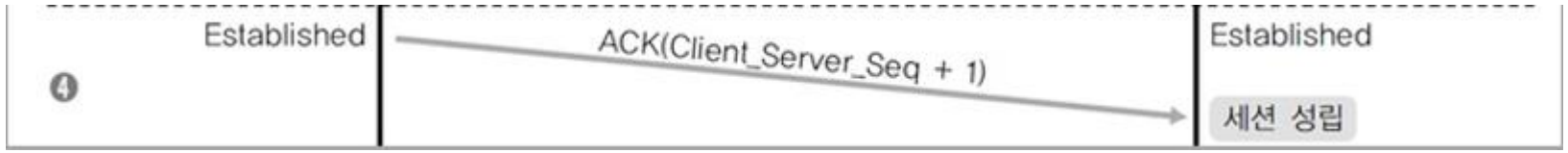
3. 클라이언트는 SYN/ACK 에서 클라이언트 자신의 시퀀스 넘버와 일치하면 서버의 시퀀스 넘버를 저장한다.

TCP

4. 클라이언트는 저장한 서버의 시퀀스에 패킷 길이를 더하여 서버에게 ACK를 보내고 자신은 Established 상태가 된다.

5. 서버는 수신된 ACK 에서 서버의 시퀀스가 자신의 시퀀스와 비교 후 일치하면 Established 상태가 된다.

즉 정상적인 동기화 상태에서는 클라이언트 자신의 시퀀스 넘버는 서버가 저장한 클라이언트의 시퀀스 넘버와 같고 서버의 시퀀스 넘버는 클라이언트가 저장한 서버의 시퀀스 넘버와 같은 상태이다.



TCP

이러한 TCP 프로토콜에서는 다양한 취약점이 존재한다.

2020년 이스라엘의 한 보안 기업이 TCP/IP 계층에서 19가지 취약점을 발견했다고 발표했다. 이 취약점을 이용한 공격에서는 전 세계 수백만 대의 IoT 장치에 대한 원격 공격이 가능하다고 경고했으며 원격코드 실행, 도스 공격, 정보 탈취 등의 악성 행위가 가능하다고 발표했다.

세션 하이재킹은 TCP 프로토콜의 취약점 중 하나인 시퀀스 넘버가 잘못되었을 때 취약한 점을 이용해서 공격한다.

TCP

TCP는 클라이언트와 서버 간 통신을 할 때 패킷의 연속성을 보장하기 위해 서버와 클라이언트 모두 각각의 시퀀스 넘버를 사용한다. 이때 시퀀스 넘버가 잘못 된다면 이를 바로 잡기 위한 작업을 하는데 세션 하이재킹은 이 작업 도중 발생하는 취약점을 이용해서 공격을 수행한다.

세션 하이재킹

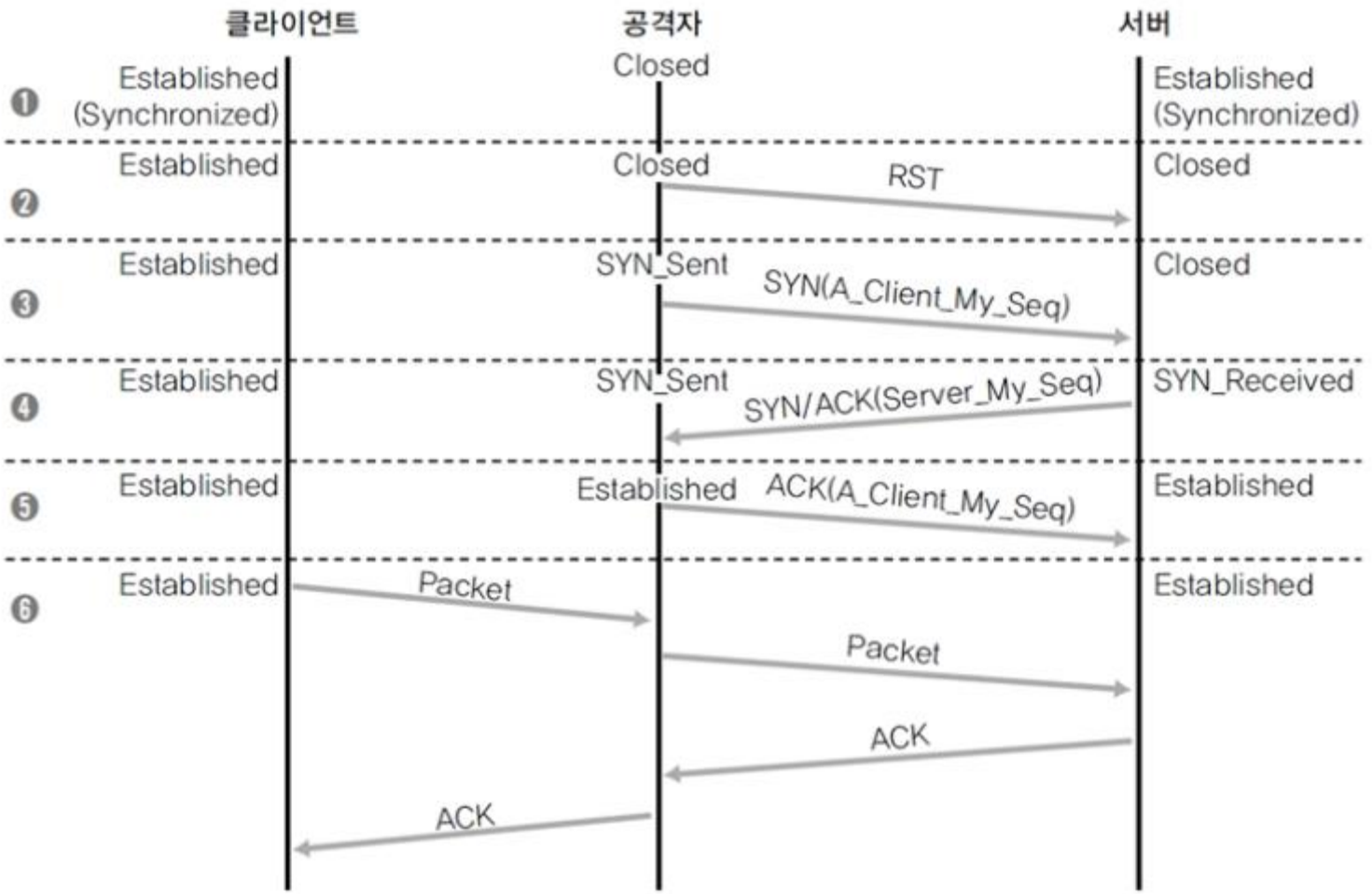
세션은 사용자와 컴퓨터, 즉 클라이언트와 서버가 서로 연결되어 활성화된 상태를 의미한다.

즉 세션 하이재킹은 서로간 연결이 활성화 되어있는 상태에서 로그인 상태를 공격자가 가로채는 것을 의미한다. 세션 하이재킹은 아이디와 비밀번호를 모르더라도 공격자가 클라이언트인 것처럼 위장해서 서버에 접속할 수 있다.

세션 하이재킹

공격 원리

1. 클라이언트와 서버 사이의 패킷을 통제하고 쌍방의 통신 패킷 모두 공격자를 지나가게 한다.
2. 서버에 클라이언트 주소로 연결을 재설정하기 위한 RST 패킷을 보내며 서버는 RST 패킷을 받아 클라이언트의 시퀀스 넘버가 재설정된 것으로 판단하고 3-way handshake 과정을 다시 수행한다.
3. 공격자는 클라이언트에서 연결되어 있던 TCP 연결을 자신이 얻게 된다.


클라이언트
공격자
서버
Established (Synchronized)
Closed
Established (Synchronized)
Established
Closed
RST
Closed
Established
SYN_Sent
SYN(A_Client_My_Seq)
Closed
Established
SYN_Sent
SYN/ACK(Server_My_Seq)
SYN_Received
Established
Established
ACK(A_Client_My_Seq)
Established
Established
Packet
Packet
ACK
ACK

ACK Storm

공격자가 서버와 3-way handshake 과정을 거칠 때 클라이언트는 서버에게 지속적으로 정상적인 패킷을 보내게 된다. 이때 서버는 이 패킷의 시퀀스 넘버를 비정상적인 시퀀스 넘버로 인식하여 서버와 클라이언트간 지속적으로 SYN, ACK, SYN/ACK 패킷을 보내게 된다.

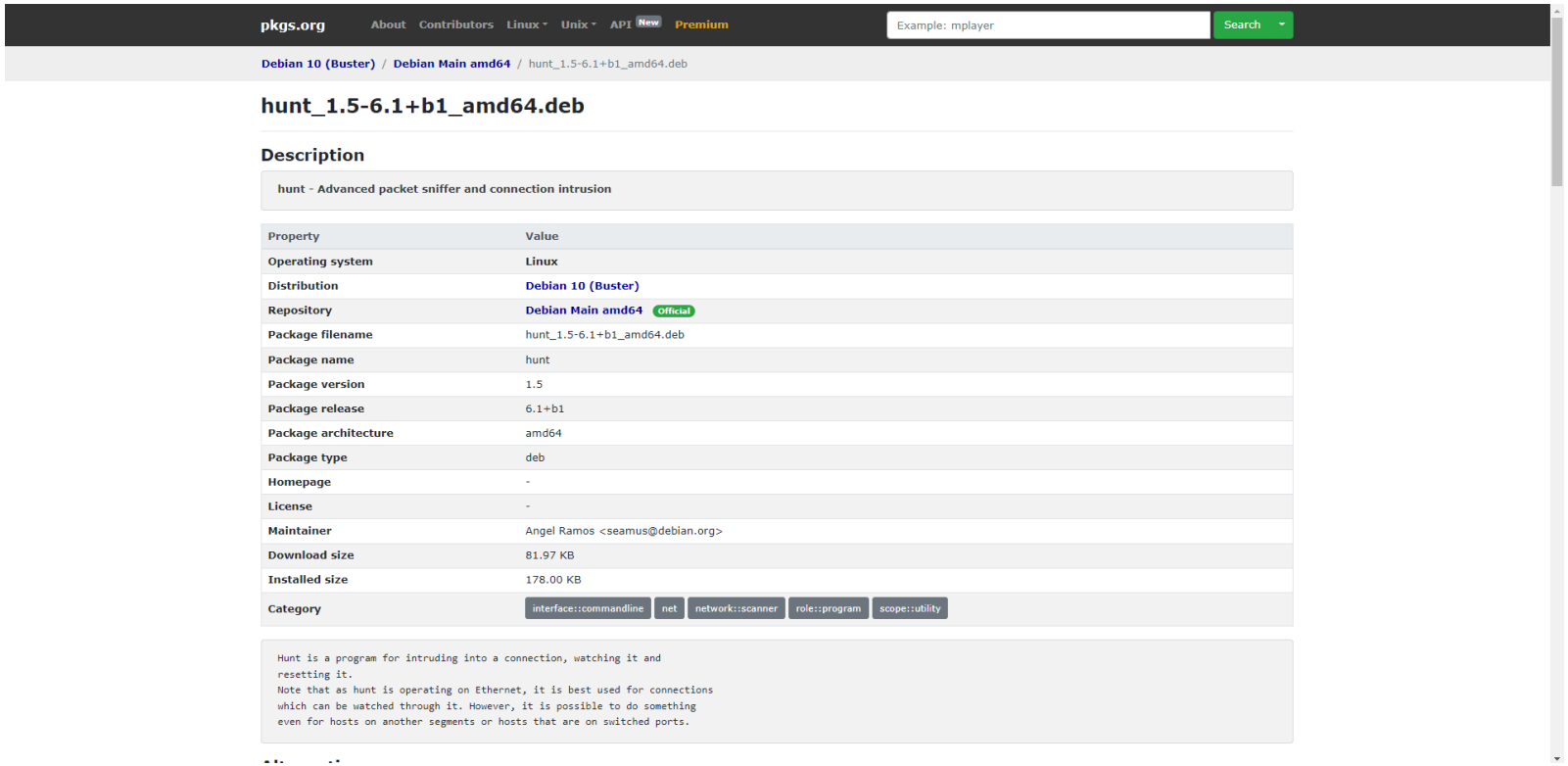
즉 이러한 송수신이 지속적으로 반복되는 경우를 Ack Storm 이라 하며 공격자는 이러한 패킷이 전달되지 않도록 서버와 클라이언트간 패킷을 자신을 거쳐 가도록 하는 작업이 필요하다.

세션 하이재킹

세션 하이재킹 공격은 공격 과정 속에서 정상적인 사용자로 위장한 후 시스템을 공격하는 모습이 IP 스푸핑과 유사하다고 할 수 있다.

IP 스푸핑은 서버와 클라이언트의 트러스트 정보를 이용하여 공격을 시도하는 것이고 세션 하이재킹 공격은 활성화 되어 있는 세션을 RST 패킷을 이용하여 리셋 시킨 후 이를 악용하여 공격을 시도한다.

실습

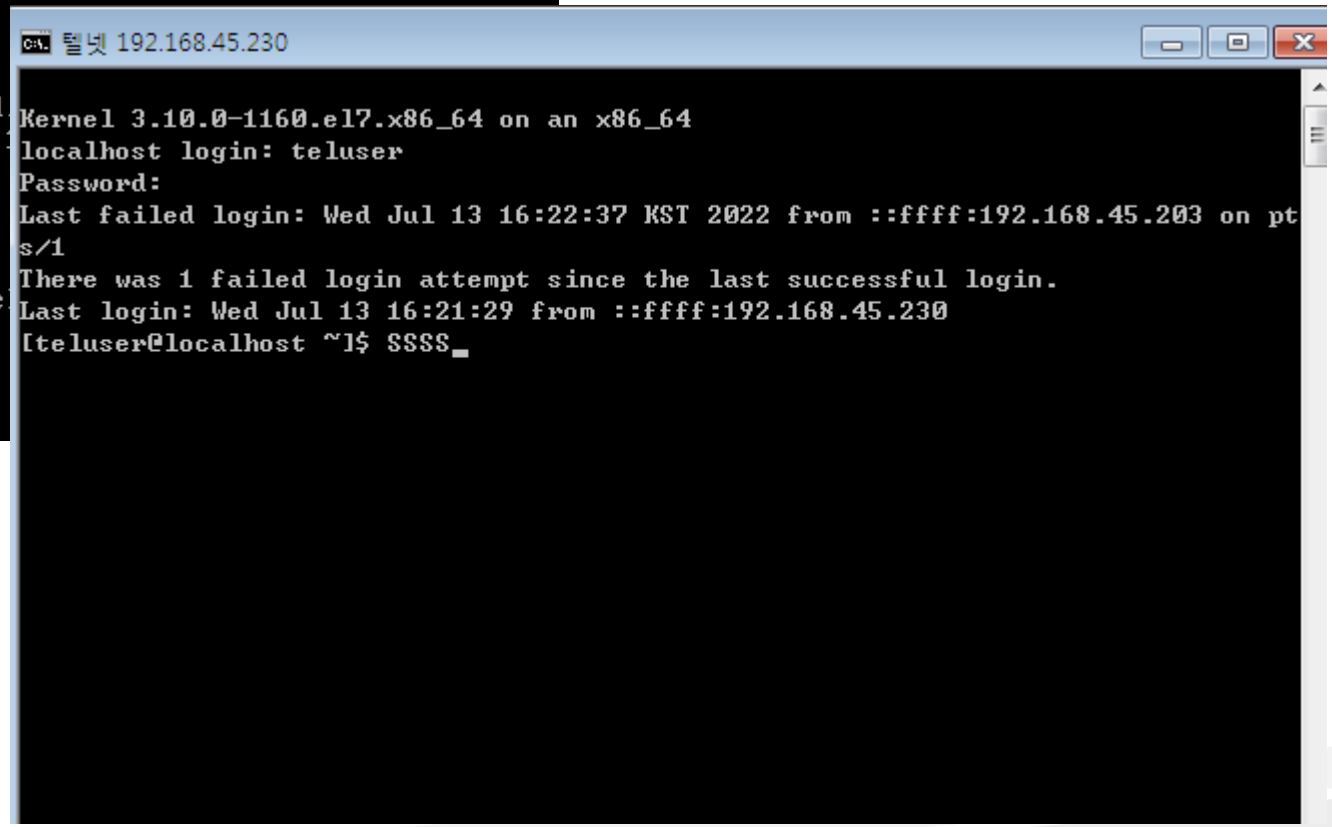


	IP주소
Attacker(칼리 리눅스)	192.168.45.55
Victim(윈도우7)	192.168.45.21
Server(CentOS)	192.168.45.230

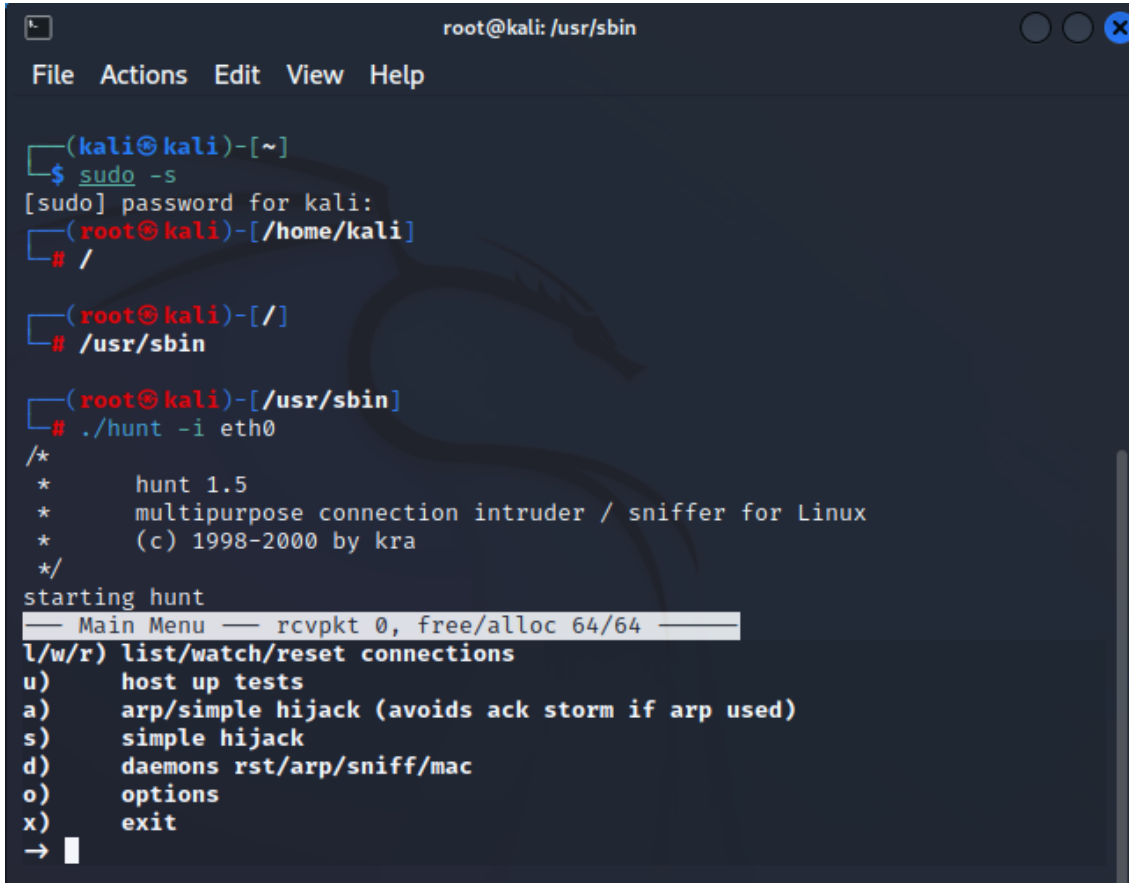
실습

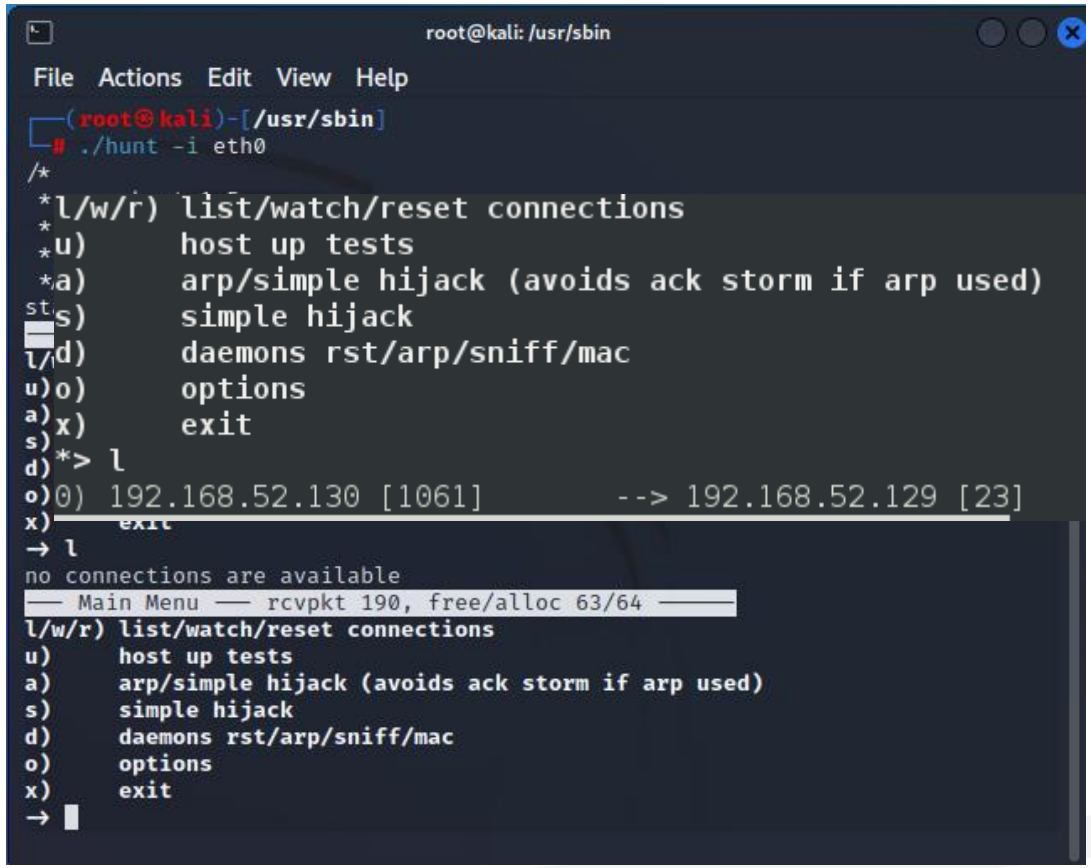
```
[root@localhost ~]# systemctl telnet.socket
Unknown operation 'telnet.socket'.
[root@localhost ~]# systemctl status telnet.socket
● telnet.socket - Telnet Server Activation Socket
   Loaded: loaded (/usr/lib/systemd/system/telnet.socket; enabled
   Active: active (listening) since Wed 2022-07-13 04:11:28 KST;
     Docs: man:telnetd(8)
   Listen: [::]:23 (Stream)
 Accepted: 0; Connected: 0

Jul 13 04:11:28 localhost.localdomain systemd[1]: Listening on Te
[root@localhost ~]# _
```

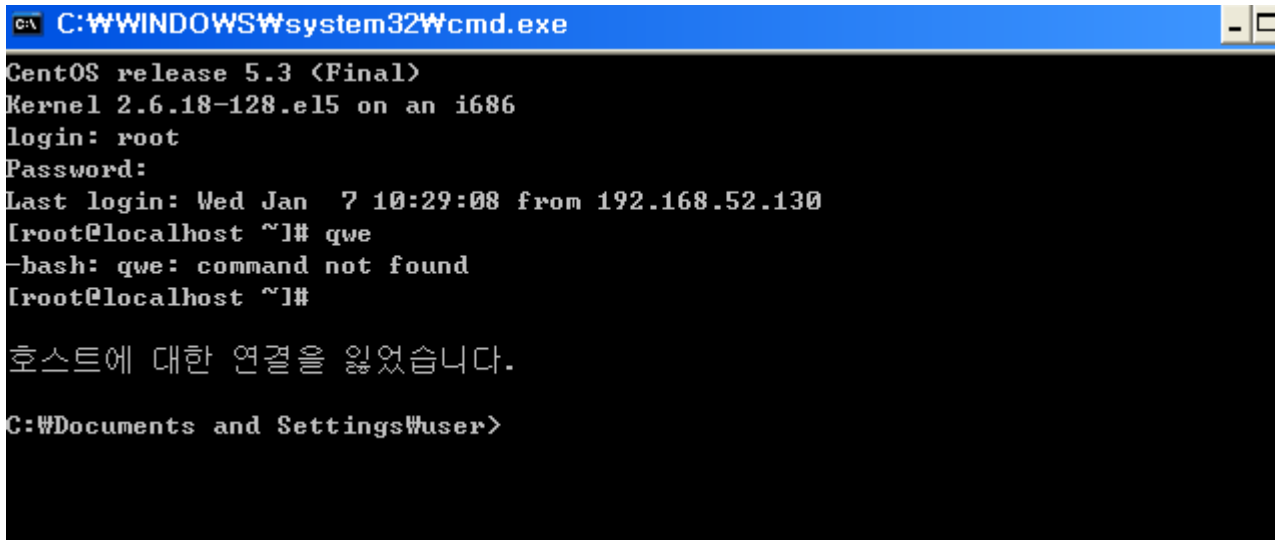


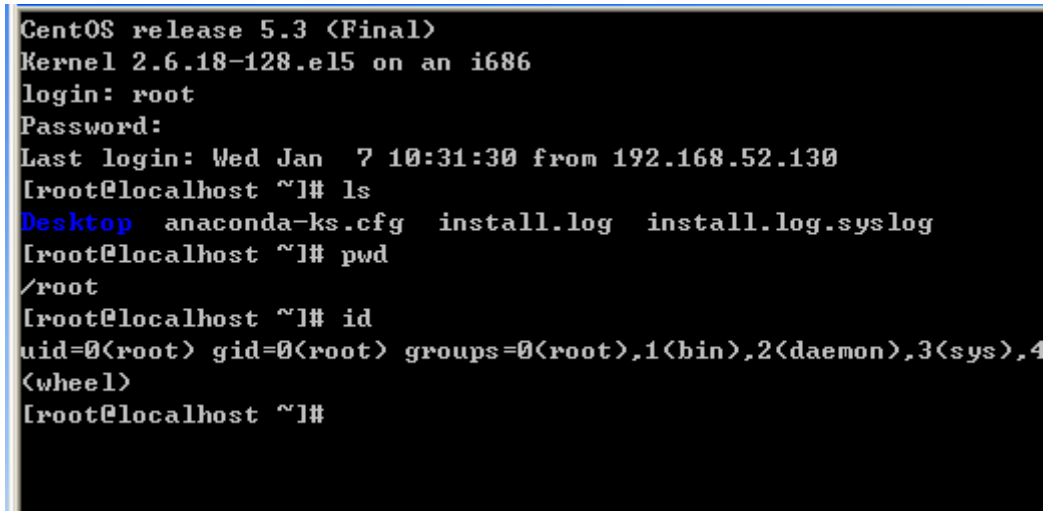
실습

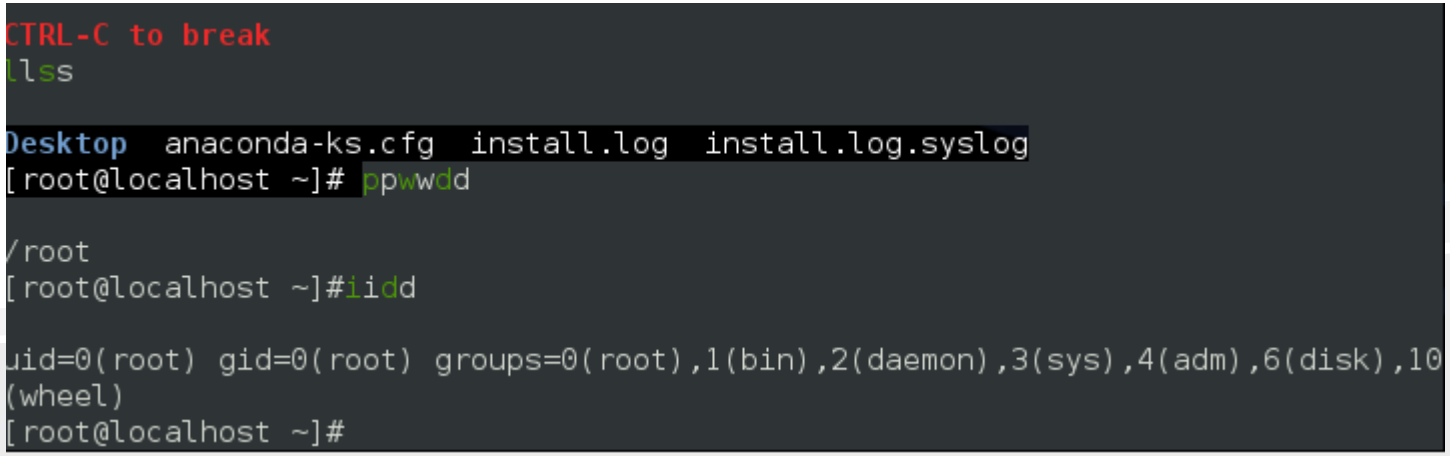

root@kali: /usr/sbin
File Actions Edit View Help
┌──(kali㉿kali)-[~]
└─$ sudo -s
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# /
┌──(root㉿kali)-[/]
└─# /usr/sbin
┌──(root㉿kali)-[/usr/sbin]
└─# ./hunt -i eth0
/*
* hunt 1.5
* multipurpose connection intruder / sniffer for Linux
* (c) 1998-2000 by kra
*/
starting hunt
── Main Menu ── rcvpkt 0, free/alloc 64/64 ──
l/w/r) list/watch/reset connections
u) host up tests
a) arp/simple hijack (avoids ack storm if arp used)
s) simple hijack
d) daemons rst/arp/sniff/mac
o) options
x) exit
→


root@kali: /usr/sbin
File Actions Edit View Help
┌──(root㉿kali)-[/usr/sbin]
└─# ./hunt -i eth0
/*
*l/w/r) list/watch/reset connections
*u) host up tests
*a) arp/simple hijack (avoids ack storm if arp used)
s) simple hijack
d) daemons rst/arp/sniff/mac
o) options
x) exit
*> l
0) 192.168.52.130 [1061] --> 192.168.52.129 [23]
x) exit
→ l
no connections are available
── Main Menu ── rcvpkt 190, free/alloc 63/64 ──
l/w/r) list/watch/reset connections
u) host up tests
a) arp/simple hijack (avoids ack storm if arp used)
s) simple hijack
d) daemons rst/arp/sniff/mac
o) options
x) exit
→

실습


C:₩WINDOWS₩system32₩cmd.exe
CentOS release 5.3 (Final)
Kernel 2.6.18-128.el5 on an i686
login: root
Password:
Last login: Wed Jan 7 10:29:08 from 192.168.52.130
[root@localhost ~]# qwe
-bash: qwe: command not found
[root@localhost ~]#
호스트에 대한 연결을 잃었습니다.
C:₩Documents and Settings₩user>


CentOS release 5.3 (Final)
Kernel 2.6.18-128.el5 on an i686
login: root
Password:
Last login: Wed Jan 7 10:31:30 from 192.168.52.130
[root@localhost ~]# ls
Desktop anaconda-ks.cfg install.log install.log.syslog
[root@localhost ~]# pwd
/root
[root@localhost ~]# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4
(wheel)
[root@localhost ~]#


CTRL-C to break
llss
Desktop anaconda-ks.cfg install.log install.log.syslog
[root@localhost ~]# ppwwdd
/root
[root@localhost ~]#iidd
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10
(wheel)
[root@localhost ~]#

실습

```
-- press any key> Enter the command string you wish executed or [cr]> ls
```

```
R

-- press any key> Enter the command string you wish executed or [cr]> ls
ls
Desktop  anaconda-ks.cfg  install.log  install.log.syslog
[root@localhost ~]# Enter the command string you wish executed or [cr]> id
id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),1
(wheel)
[root@localhost ~]# Enter the command string you wish executed or [cr]> pwd
pwd
/root
[root@localhost ~]# Enter the command string you wish executed or [cr]>
```

예방대책

패킷의 유실 및 재전송 패킷 증가를 탐지한다.

공격자가 중간에 끼어서 서버와 클라이언트 간 통신을 하므로 패킷의 유실과 재전송이 발생하여 서버와의 응답시간이 길어진다.

네트워크 트래픽을 감시하며 ACK Storm 이 발생하는지 탐지한다.

클라이언트와 서버는 서로 시퀀스 넘버가 잘못되었음을 확인하였으므로 과도한 ACK패킷을 보내기 때문에 ACK Storm이 발생할 수 있다.

예방대책

데이터를 전송할 때 암호화 방식을 사용한다.

데이터 전송의 암호화는 스니핑에 대한 대응책이 되기도 한다. 데이터를 전송할 때 암호화를 하면 공격자가 시퀀스 넘버를 추측하기 어렵게 한다.

지속적인 인증을 한다.

대부분의 시스템이 최초 로그인 시 패스워드를 입력한 후에는 다시 재인증 과정을 하지 않는데 어떤 수상한 행동을 하거나 최초 접속 후 일정 시간이 지나면 다시 패스워드를 물어보는 등 접속자가 정당한 과정을 통해 서버에 접속했는지 확인한다.

예방대책

수상한 접속에서 RST 패킷이 발생하는지 탐지한다.

세션에 대한 공격 시도에서 세션이 멈추거나 RST 패킷이 지속적으로 수신된다면 의심해볼 필요가 있다.