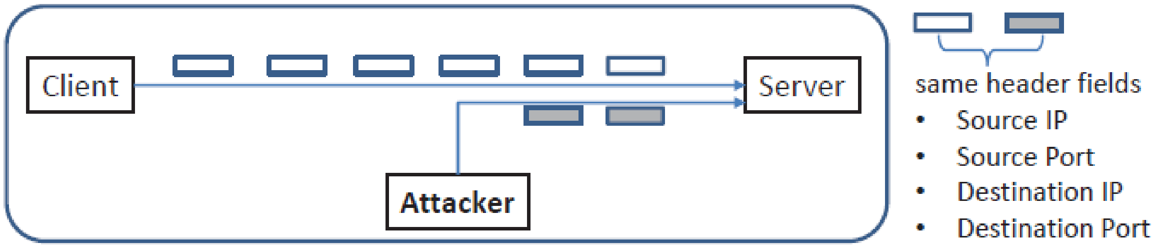
TCP Session Hijacking

Definition:

TCP session hijacking is a process in which an attacker can intercept a TCP session between two machines. Since the authentication check is performed only during session initialization the attacker can perform the attack after some duration. The attacker gets the current value of the absolute sequence and acknowledgement number of the TCP session and forges a TCP packet with the next sequence and acknowledgement number and sends it to one of the two machines.



Toplogy diagram:

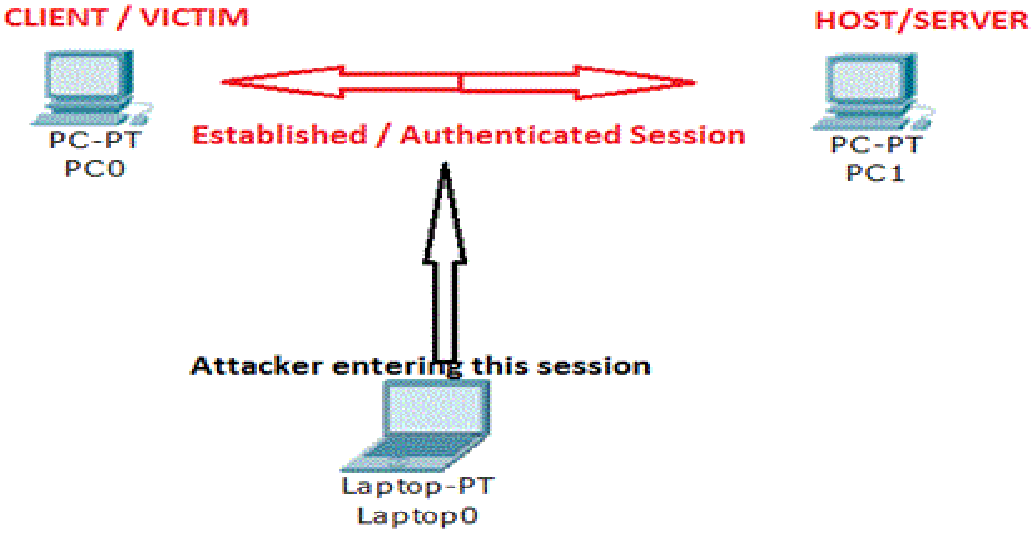
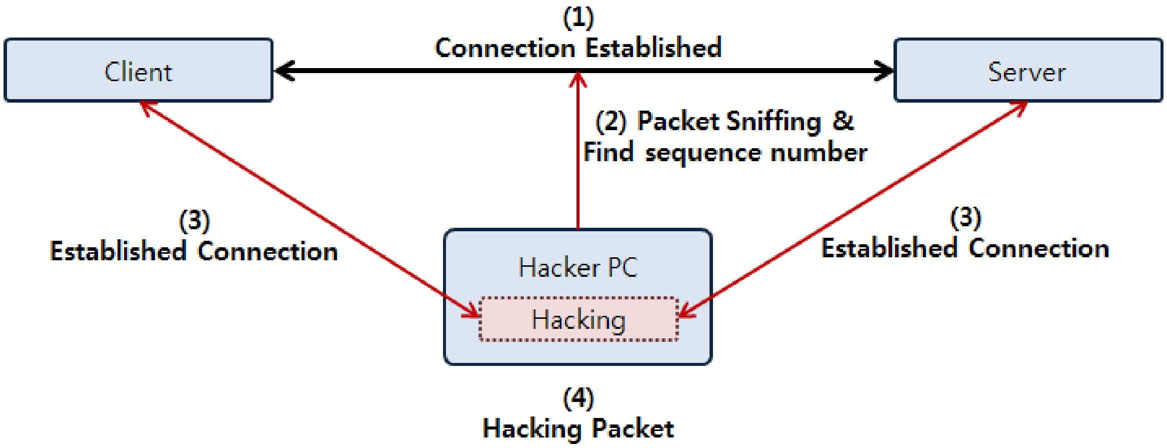

CLIENT / VICTIM
PC-PT
PC0
Established / Authenticated Session
HOST/SERVER
PC-PT
PC1
Attacker entering this session
Laptop-PT
Laptop0

Fig: Session hijacking using sniffing

Hijacking using sniffing:

This attack can be carried out when the attacker can sniff the packets between client and host, where the attacker monitors the traffic and after a while hijacks the connection with appropriate packet injection

Hijacking using man in the middle attack:



A more elaborate scheme can be achieved by a man in the middle attack, where attacker hijacks the connection and tricks both the server and the client to send their data to the attacker. This can be achieved using ARP cache poisoning for both the server and the client. In this attack, the attacker can carry out potentially more damage both to server and client by crafting malicious data for both of them.

Timing Diagram:

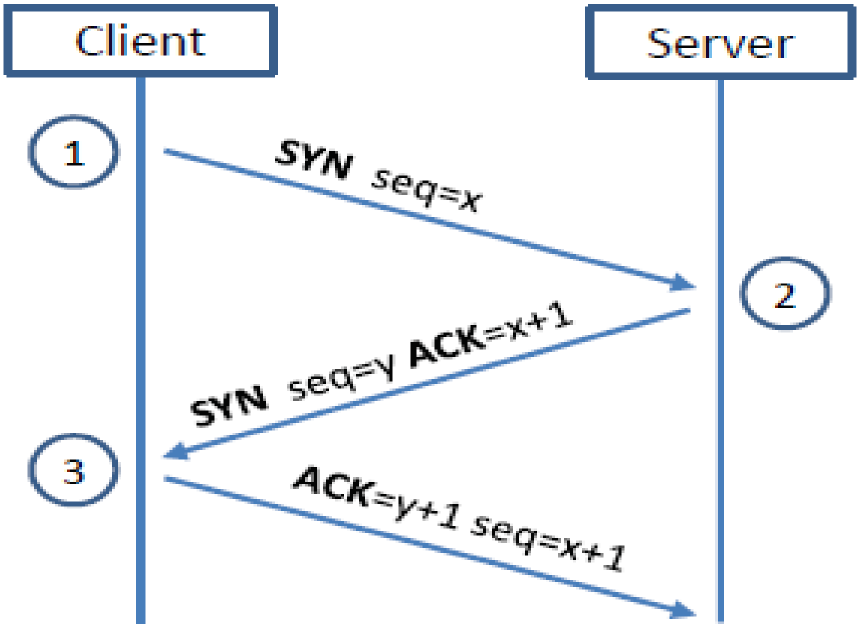


Fig: Timing diagram of the original TCP protocol

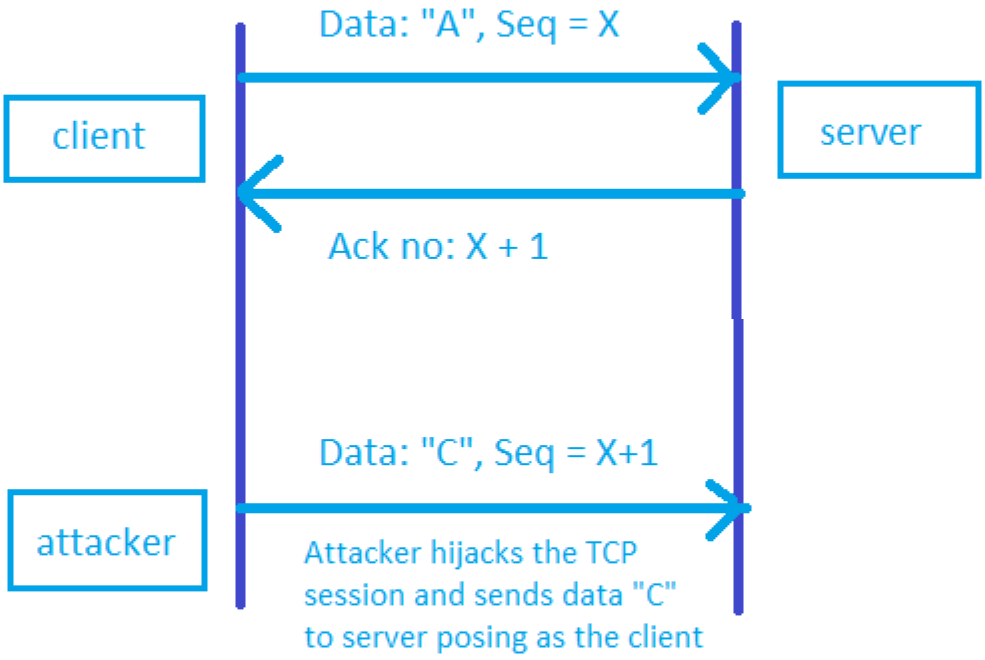

Data: "A", Seq = X
client
server
Ack no: X + 1
Data: "C", Seq = X+1
attacker
Attacker hijacks the TCP session and sends data "C" to server posing as the client

Fig: Timing diagram of the TCP session hijacking

Frame/Packet Details:

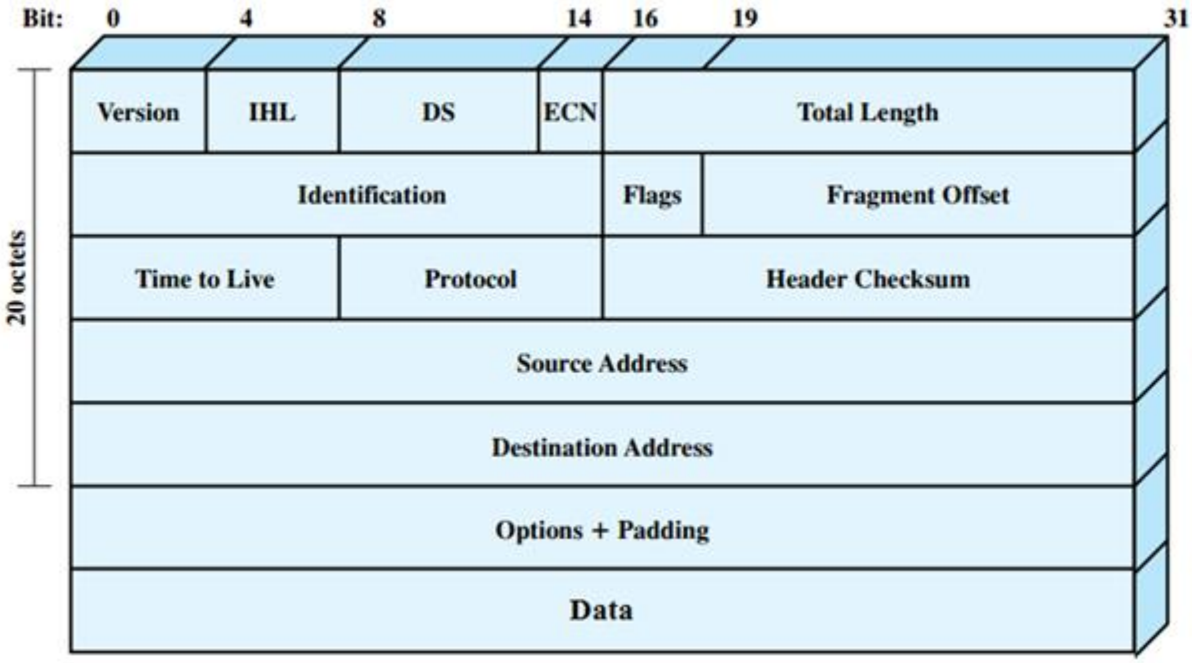


Figure IPv4 Header

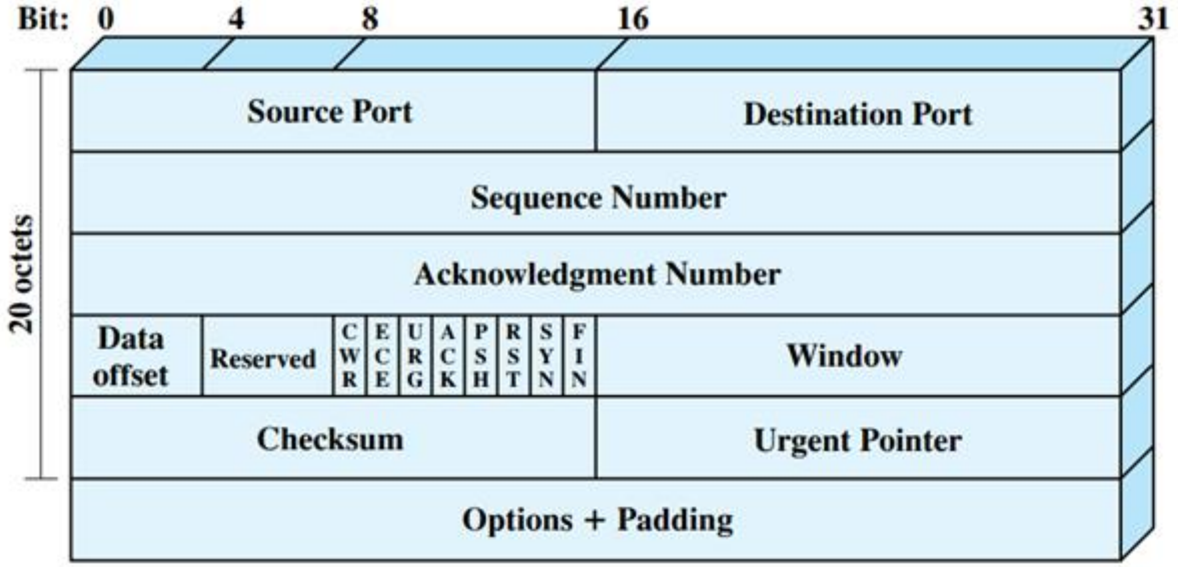


Figure TCP Header

Modifications needed in the forged packet for hijacking:

In IPv4 Header:

- Source Address – address of the original client
- Destination Address – address of the server
- TTL(Time to Live) – Value obtained from the spoofed packets

In TCP header:

- Source Port – port no of the telnet connection in client PC
- Destination Port – 23 for telnet
- Sequence no – Predicted using spoofed packets
- Window size – Suitable size
- Acknowledgement Number – Obtained from spoofed packets
- ACK bit - 1
- TCP data – Payload attacker wants to send

Rest of the fields can be set to their default values. Further headers need to be changed if man in the middle techniques such as ARP cache poisoning are used.

Attacking Strategy:

The attack tool will be designed to hijack an existing telnet connection between the victim and the server, where the attacker will on the same LAN as victim and server. The steps carried out in the basic attack are as follows:

- First, attacker will run a sniffer tool like Wireshark on his machine to sniff the packets between the victim and server. If Wireshark can't be used, an arpspoof attack will be carried out from the attacker to both the client and server. In either case, attacker now has access to the packets sent between server and client.
- From the sniffed packet(s), attacker gets the necessary information about the session such as the sequence and acknowledgement numbers, source and destination port numbers.
- Using these information, attacker will spoof a packet from victim to server incorporating the next sequence number, acknowledgement number and port numbers using the tool.
- Now, by configuring the TCP payload, attacker can run arbitrary commands on the server machine posing as the victim.

Justification:

After the initial authorization between victim and server, a session gets set up between the two machines. TCP uses the following elements of a TCP packet to uniquely identify a session and its data

- Source IP address
- Destination IP address
- Source Port number
- Destination Port number
- Sequence number

i.e. after the initial authorization, we don't need to authorize using password for every other packet.

Here, just matching the source and destination IP and ports isn't enough, because TCP maintains a window of sequence numbers and if the incoming packets sequence number is outside the current window the packet will be discarded. To be precise, attacker needs the current sequence number that the server accepts to be sent within the forged packet.

Hence, a spoofed packet from the attacker with the appropriate values of these fields can't be distinguished from a valid packet from the authorized sender and therefore using this attack strategy attacker can effectively hijack the telnet TCP session.