# Session Hijacking
# Exploiting TCP, UDP and HTTP Sessions

Shray Kapoor
shray.kapoor@gmail.com

## Preface

With the emerging fields in e-commerce, financial and identity information are at a higher risk of being stolen. The purpose of this paper is to illustrate a common cum valiant security threat to which most systems are prone to i.e. *Session Hijacking*. Sensitive user information are constantly transported between sessions after authentication and hackers are putting their best efforts to steal them .In this paper I will discuss mechanics of the act of session hijacking in TCP and UDP sessions i.e. hijacking at the network level and at Application levels i.e. hijacking HTTP sessions.

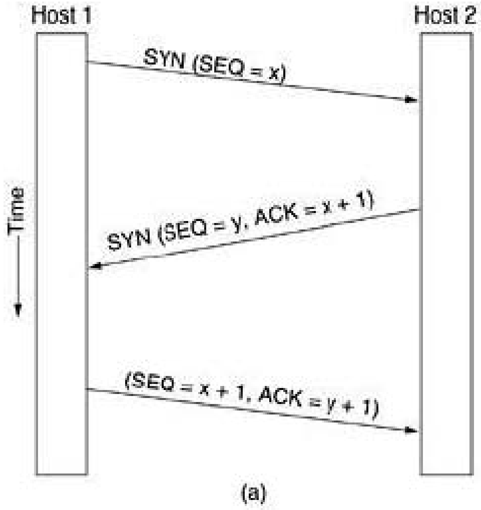
**Table of Contents**

## Background

Session hijacking can be done at two levels: *Network Level* and *Application Level*. Network layer hijacking involves TCP and UDP sessions, whereas Application level session hijack occurs with HTTP sessions. Successful attack on network level sessions will provide the attacker some critical information which will than be used to attack application level sessions, so most of the time they occur together depending on the system that is attacked. Network level attacks are most attractive to an attacker because they do not have to be customized on web application basis; they simply attack the data flow of the protocol, which is common for all web applications.

## Introduction to TCP

TCP an abbreviation for ***Transmission Control Protocol,*** one of the main connections oriented protocol in a TCP/IP network. TCP was formally defined in RFC 793 (while extensions are given in RFC 1323), as a protocol for providing a reliable end -to-end communication on a non-reliable network. To establish a session or a connection with a TCP server, a client must have to follow a structured system for session management; this system is known as "*Three Way Handshake*". For two machines to communicate via TCP they must have to synchronize their session through Synchronize and Acknowledgement Packets. Every single packet is given a sequence number which helps the receiving host to synchronize and reassemble the stream of packets back into their original and intended order. TCP session establishment is shown in figure:



(Figure and session establishment summary taken from **Computer Networks by Andrew S. Tanenbaum, Prentice hall***)*

1.        1. Client sends a SYN request to server with initial sequence number X.
2.        2. Server sends the SYN/ACK packet publishing its own Sequence number SEQ y and Acknowledgement number ACK for the client's original SYN  packet. The ACK indicates the next SEQ number expected from client by the server.
3.        3. Client acknowledges the receipt of the SYN/ACK packet from the server by sending the ACK number which will be the next sequence number expected from the server, y+1 in this case.

The following example shows the three-way handshake, using TCP dump to display the exchange:

```
tclient.net.39904 > telnet.com.23: S 733381829:733381829(0) win 8760 <mss
1460> (DF)
```

```
telnet.com.23 > tclient.net.39904: S 1192930639:1192930639(0) ack 733381830
win 1024 <mss 1460> (DF)
tclient.net.39904 > telnet.com.23: . ack 1 win 8760 (DF)
```

(Reference from **New Riders Intrusion Detection** 3rd edition)

tclient at port 39904 attempting to establish session with telnet.com at 23 port with SEQ number marked by S (start:end(bytes)) flag ; publishing its Window size which is the buffer size i.e. 8760 in this case and also publishing Maximum Segment Size(mss).

Rest all communication follows the standard handshake mechanism After the session establishment its mere a matter of sending and receiving packets and increasing the sequence and the acknowledgement numbers accordingly.

## Introduction to UDP

UDP is a User Datagram Protocol, unlike TCP, it does not provide connection oriented service. UDP does not use sequencing for session establishment and sending packets instead it is used for broadcasting messages across the network or for DNS or ARP queries. UDP is our second hijacking stage in Network level hijack attacks.

## Introduction to HTTP

Hyper Text Transfer Protocol (HTTP) is a stateless protocol used by World Wide Web ; which defines how messages are formatted and transmitted between client and servers, and what actions Web servers and browsers should take in response to various commands. For establishing a connection with a server over HTTP: one has to establish a TCP connection on port 80 on the servers machine. Every session maintains a unique Session ID for the current live session with the server; which can be the target for stealing sessions. This is the last stage of session hijacking.

# Hijacking at Network levels

Network level session attacks are done with TCP and UDP sessions, which are discussed in detail in the following sections.

## TCP Session Hijack

TCP hijacks are meant to intercept the already established TCP sessions between any two communicating parties and than pretending to be one of them, finally redirecting the TCP traffic to it by injecting spoofed IP packets so that your commands are processed on behalf of the authenticated host of the session. It desynchronizes the session between the actual communicating parties and by intruding itself in between. As authentication is only required at the time of establishing connection , an already established connection can be easily stolen without going through any sort of authentication or security measures concerned. TCP session hijacks can be implemented in two different ways: Middle Man Attack (suggested by Lam, LeBlanc, and Smith) and the Blind attack. Before moving further there is need to understand IP spoofing which is discussed in the next subsection.

## IP Spoofing: Assuming the identity

Spoofing is pretending to be someone else. This is a technique used to gain unauthorized access to the computer with an IP address of a trusted host. The trusted host in case of session hijacking is the client with whose IP address we will spoof our packets so that our packets will become acceptable to the server maintaining the session with the client. In implementing this technique session hijacker has to obtain the IP address of the client and inject his own packets spoofed with the IP address of client into the TCP session, so as to fool the server that it is communicating with the victim i.e. the original host.

What remains untouched is how to alter the sequence and the acknowledgement numbers of the spoofed packets which the server is expecting from the client. Once it is altered, hijacker injects its own forged packet in the established session before the client can respond , ultimately desynchronizing the original session , because now our server will expect a different sequence number , so the original packet will be trashed. Based on the anticipation of sequence numbers there are two types of TCP hijacking: Man in the Middle and Blind hijacking.

## Man in the Middle attack using Packet Sniffers

This technique involves using a packet sniffer to intercept the communication between client and the server. Packet sniffer comes in two categories: Active and Passive sniffers. Passive sniffers monitors and sniffs packet from a network having same collision domain

i.e. network with a hub, as all packets are broadcasted on each port of hub. Active sniffers works with Switched LAN network by ARP spoofing (For more information on Active Sniffers refer *Ethical Hacking and Countermeasures EC Council Exam 312 50 (OSB-2004)*).Once the hijacker reads the TCP header, he can know the sequence number expected by the server , the acknowledgement number, the ports and the protocol numbers ; so that hijacker can forge the packet and send it to the server before the client does so.

Another way of doing so is to change the default gateway of the client's machine so that it will route its packets via the hijacker's machine. This can be done by ARP spoofing (i.e. by sending malicious ARP packets mapping its MAC address to the default gateways address so as to update the ARP cache on the client , to redirect the traffic to hijacker).

**Blind Attack**

If you are not able to sniff the packets and guess the correct sequence number expected by server, you have to implement "*Blind Session Hijacking*". You have to brute force 4 billion combinations of sequence number which will be an unreliable task.

### UDP Session Hijacking

Since UDP does not use packet sequencing and synchronizing; it is easier than TCP to hijack UDP session. The hijacker has simply to forge a server reply to a client UDP request before the server can respond. If sniffing is used than it will be easier to control the traffic generating from the side of the server and thus restricting server's reply to the client in the first place.

## Hijacking Application Levels

At this level a hijacker can not only hijack already existing sessions but can also create new sessions from the stolen data.

### HTTP Session Hijack

Hijacking HTTP sessions involves obtaining Session ID's for the sessions, which is the only unique identifier of the HTTP session. Session ID's can be found at three places

1. 1. In the URL received by the browser for the HTTP GET request.
2. 2. With cookies which will be stored in clients computer.
3. 3. Within the form fields.

**Obtaining Session ID's**

One way to obtain the Session ID is by sniffing, which is same as the ***Man in middle attack***. Cookies and URL's can be sniffed from the packets and if unencrypted can provide critical user logon information.

Another way is by ***Brute Forcing*** the Session ID's which involves trying a set of session id's based on some pattern. Brute forcing is a time consuming task but worked on some algorithm can produce results rather quickly.

## Countermeasures

To defend your network with session hijacking, a defender has to implement both security measures at Application level and Network level. Network level hijacks can be prevented by **Ciphering the packets** so that the hijacker cannot decipher the packet headers, to obtain any information which will aid in spoofing. This encryption can be provided by using protocols such as ***IPSEC ,SSL, SSH*** etc. Internet security protocol (IPSEC) has the ability to encrypt the packet on some shared key between the two parties involved in communication. IPsec runs in two modes: Transport and Tunnel. In Transport Mode only the data sent in the packet is encrypted while in Tunnel Mode both packet headers and data are encrypted, so it is more restrictive.

To prevent your Application session to be hijacked it is recommended to use **Strong Session ID's** so that they cannot be hijacked or deciphered at any cost. **SSL (Secure Socket layer) and SSH (Secure Shell)** also provides strong encryption using SSL certificates so that session cannot be hijacked, but tools such as Cain & Bell can spoof the SSL certificates and decipher everything! **Expiring sessions** after a definite period of time requires re-authentication which will futile the hacker's tricks.

## Summary

Session hijacking is a serious threat to Networks and Web applications on web as most of the systems are vulnerable to it. Although above explanation and countermeasures will give insight to the defender to protect his /her network, but it will also raise the security bar and will force the hijackers to apply more complex attacks to compromise the system. Networks should be tested and monitored continuously in order to make them impenetrable by the intruders.

The *'Strange Attractors and TCP/IP Sequence Number Analysis'* gives a great insight on Pseudo-Random Number Generators(PRNG) to avoid Blind attacks by guessing sequence numbers. This is available at: http://www.bindview.com/Services/Razor/Papers/2001/tcpseq.cfm