

Session Hijacking Packet Analysis

Lee Lawson

Introduction

'Session Hijacking' is a high level attack vector which many systems are completely open to. Most systems are vulnerable to this type of attack as most systems use Transmission Control Protocol (TCP), the standard communication protocol used on the Internet and internal Local Area Networks (LANs). This paper assumes a level of network competency by the reader to being equivalent to that of a network engineer or experienced administrator.

Background

To establish a session with a TCP server, a client must follow a structured system of packet transmissions; this system is known as the '3 Way Handshake'. For two TCP enabled machines to talk to each other, they must synchronize, specifically they must inform the other machine of their communication settings such as Sequence Number (SEQ) and Window size (WIN). ALL packets transmitted in a TCP connection must have a sequence number as TCP is a connection oriented protocol; every single packet has to be assigned a session unique number that will enable the receiving machine to reassemble the stream of packets back into their original and intended order. If the packets arrive out of order, as can happen regularly over the internet, then the SEQ is used to stream them correctly.

The 3 Way Handshake

The synchronization of two TCP computers has to follow a defined process, a handshake. Both machines must inform the other of communication specific settings vital to the successful transmission of data. These settings are used so that each machine knows of the other's capabilities in handling TCP packets. The 3 way handshake, as the name suggests, has 3 parts. The following diagram shows the 3 steps in establishing a handshake, and therefore a TCP session.



CLIENT



SERVER

SYN <Clt ISN> <WIN>



SYN <Svr ISN> <WIN> / ACK (Clt ISN +1)



ACK (Svr ISN +1)



The computer wishing to initiate the TCP session, the Client in the above example, transmits a packet with the SYN control bit set, a synchronize packet. This packet includes the clients 'Initial Sequence Number' (ISN) and 'Window' size (WIN).

The ISN is a pseudo-randomly generated number. It is essential to remember that the actual

Session Hijacking Packet Analysis

Lee Lawson

sequence number space is finite, although very large. This space ranges from 0 to $2^{32} - 1$, which equates to 4,294,967,295 (over 4 Billion) possible combinations.

Every TCP terminal has a Window size that tells the sender how many bytes it can send before the receiver will have to toss it away due to fixed input buffer size. Imagine it as a bucket of water, if you pour too much water into my bucket, it will overflow. The Window size tells both machines what the size of bucket the other has.

You may have noticed that the acknowledgement (ACK) by each machine is the received packet sequence number plus one increment. This method of acknowledgement will tell the sender the next expected TCP packet sequence number. When within the 3 way handshake, the increment value is literally 1. When inside normal data communications, the increment value is that of the data size in bytes, e.g. you transmit 38 bytes of data, the increment goes up by 38 to ACK the 38 bytes.



SYN <Clt ISN 4000> <WIN 512>



SYN <Svr ISN 5000> <WIN 1024> / ACK 4001



ACK 5001



DATA=128 <Clt SEQ 4001>



ACK (Clt SEQ + Data) 4129



DATA=91 <Clt SEQ 4129>



ACK (Clt SEQ + Data) 4220



Again, the ACK is the next expected packet to be sent by the client.

Great! So that's how TCP sessions are established and normal TCP communications takes place. So what is 'Session Hijacking'?

Session Hijacking Packet Analysis

Lee Lawson

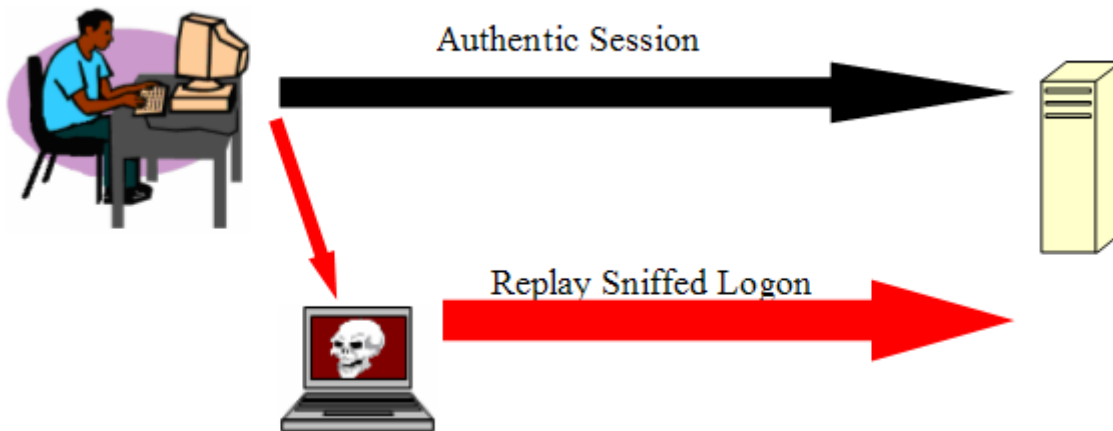
Session Hijacking

“Session Hijacking - A method of attack which involves a third party intercepting communications in a session, or series of communications, and pretending to be one of the parties involved in the session.”

<http://www.cryptnet.net/fdp/crypto/crypto-dict.html#S>

I do not quite agree with the above definition by cryptnet.net. As far as the receiving computer is concerned, you ARE the other party in the session. This comes down to the subtle difference between spoofing and hijacking.

Spoofing is pretending to be someone else. This could be achieved by sniffing a logon/authentication process and replaying it to the server after the user has logged off. The server may then assume you are the user that the sniffed process actually belongs to.

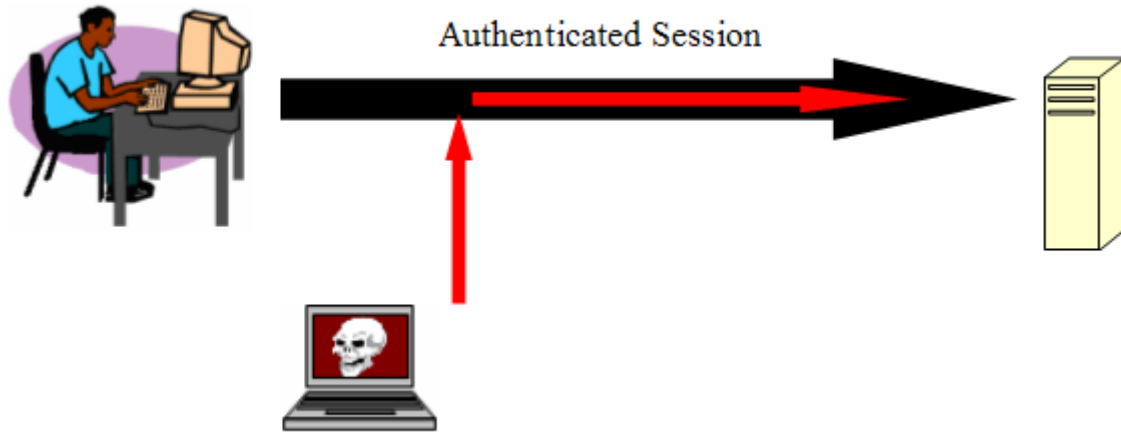


Hijacking is taking over an already established TCP session and injecting your own packets into that stream so that your commands are processed as the authentic owner of the session.

One problem with TCP is that it was not built with security in mind. Any TCP session is identified by the (client IP address + client port number) and (server IP address + server port number). Any packets that reach either machine that have those identifiers are assumed to be part of the existing session. So if an attacker can spoof those items, they can pass TCP packets to the client or server and have those packets processed as someone else!

Session Hijacking Packet Analysis

Lee Lawson



To complete a hijack you must perform 3 actions.

- Monitor or track a session
- Desynchronize the session
- Inject your own commands

To monitor a session, you simply sniff the traffic. How do we achieve the de-synchronization of a session?? By 'Sequence Packet Prediction'.

If we can predict the next sequence number to be used by a client (or server), we can use that sequence number before the client (or server) gets a chance to. Predicting the number may seem a difficult task to do as we have a possible 4 billion combinations, but remember that the ACK packet actually tells us the next expected sequence number. If we have access to the network and can sniff the TCP session, we don't have to guess, we are told what sequence number to use! This is known as 'Local Session Hijacking'. More security tools are also available.

If you do not have the ability to sniff the TCP session between client and server, you will have to attempt 'Blind Session Hijacking'. This attack vector is much less reliable as you are transmitting data packets with guessed sequence numbers. 4 billion possible combinations then becomes a very big pool to choose from!

Below is a packet analysis of a local session hijack.

Session Hijacking Packet Analysis

Lee Lawson



SYN <Clt ISN 4000> <WIN 512>



SYN <Svr ISN 5000> <WIN 1024> / ACK 4001



ACK 5001



DATA=128 <Clt SEQ 4001>



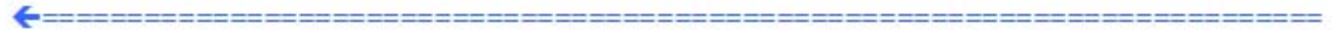
ACK (Clt SEQ + Data) 4129



DATA=91 <Clt SEQ 4129>



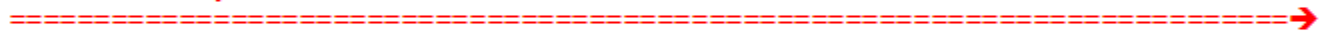
ACK (Clt SEQ + Data) 4220



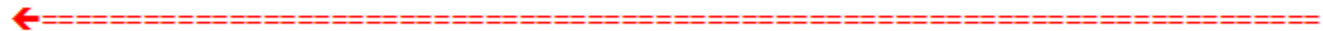
We now know the next expected sequence number. If we transmit that packet sequence number before the client, we will desynchronize the connection; basically we will bump the server up by one increment.



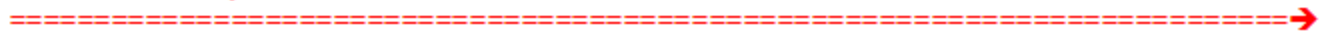
DATA=10 <SEQ 4220>



ACK 4230



DATA=512 <SEQ 4230>

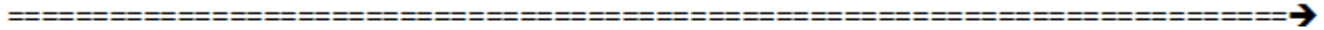


What happens when the real client sends the next packet that it has?

Session Hijacking Packet Analysis

Lee Lawson

DATA=145 <SEQ 4220>



The server treats it as a resent packet as it has already received that SEQ number. So, now the client is unable to communicate with the server, the hacker is still able to communicate as they know the correct sequence number. This dropping of packets can create a problem with the network. Because the client is not receiving an ACK for his TCP packet, he assumes that it did not make it to the server and resends it, only to have it dropped by the server.

It is important to note at this point that to carry out this attack correctly, you must employ an 'ARP Cache poisoning' attack on both machines. The reason you must redirect the packets through the Hacker machine is so that the server does not reply directly to the client. By transferring all packets through the Hacker machine, the Hacker is able to filter out what they want the server or client to be able to see and control any communications between the two. The hijacking tools will do this for you!

If the flow of data is not completely controlled, packets may get through to either party. The server may be able to send an ACK packet to the client, this packet will contain a sequence number that the client is not expecting, so when the client receives this packet, it will try to resynchronize the TCP session with the server by sending it an ACK packet with the sequence number that it is expecting. This ACK packet will in turn contain a sequence number that the server is not expecting, and so the server will resend its last ACK packet. This cycle goes on and on and on, and this rapid passing back and forth of ACK packets creates an 'ACK Storm'. This 'ACK Storm' can quite quickly grind a network to a halt, so any attack tends to be carried out rapidly by the hacker.

DATA=145 <SEQ 4220>



DATA=145 <SEQ 4220>



DATA=145 <SEQ 4220>



ACK 4742



ACK 5001



ACK 4742



ACK 5001



ACK 4742



ACK 5001



To clear the ACK storm, the Hacker could send TCP packets to both parties with the control bit set to RST, reset. This will essentially tear down the established session, disconnecting all connected machines. However, if the attacker is performing an 'ARP Cache Poison' of the two machines, the

Session Hijacking Packet Analysis

Lee Lawson

'ACK Storm' should not occur as neither machine can directly communicate with each other.

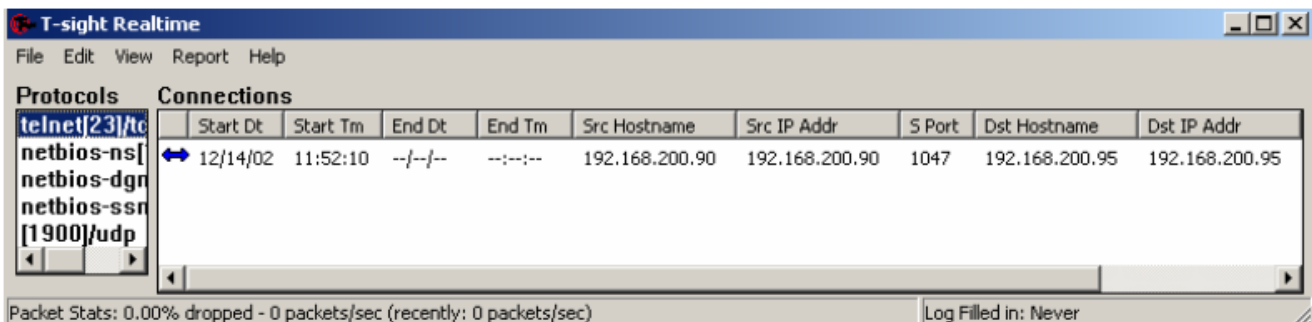
Available Hacking Tools

There are a number of tools available to conduct TCP Session Hijacking, some open source, some commercial applications. In the interest of operating system independence, I will mention two.

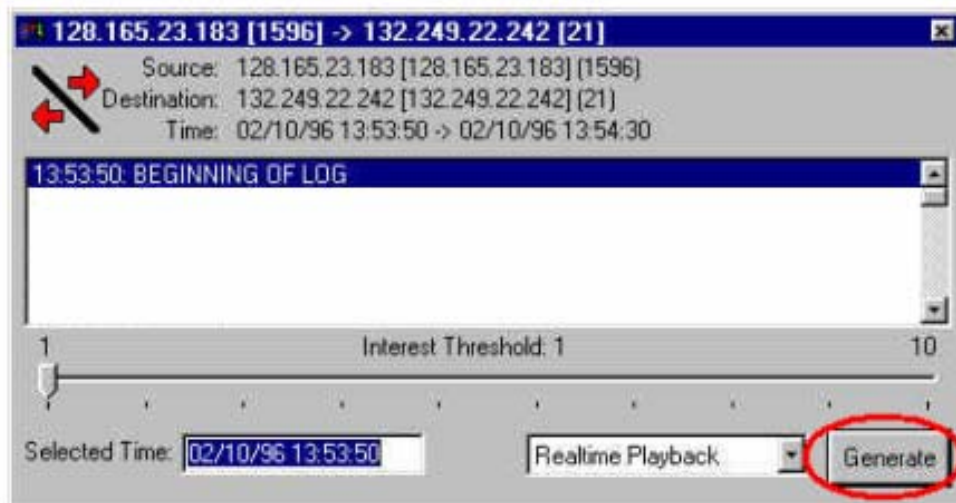
T-Sight

T-Sight, by Engarde Systems is a Windows based 'Local Session Hijacking' application. It is actually much more than just that, it is a post-mortem and real-time network analysis tool.

T-Sight is really easy to use, I do not know of another session hijacking tool that is as easy to use! Its main screen is a connection monitoring window. This will display on the left hand 'Protocols' pane all of the connections that T-Sight can 'see'. This means that if you are on a hub based network, you can see plenty of connections. If you are on a switch based network, you will need to perform some attack, such as 'Switch Flooding' or 'ARP Cache Poisoning' to be able to sniff or see other user's connections.



Once you see a connection you wish to monitor, you double click on the entry in the right hand pane, the 'connections pane'. This opens a new window with which you can analyze post-mortem packets, or by clicking on 'Generate', you can move into the real-time monitoring window.



Session Hijacking Packet Analysis

Lee Lawson

sent in quick succession to the server, enabling the Hacker to carry out the attack and get out before anyone notices.

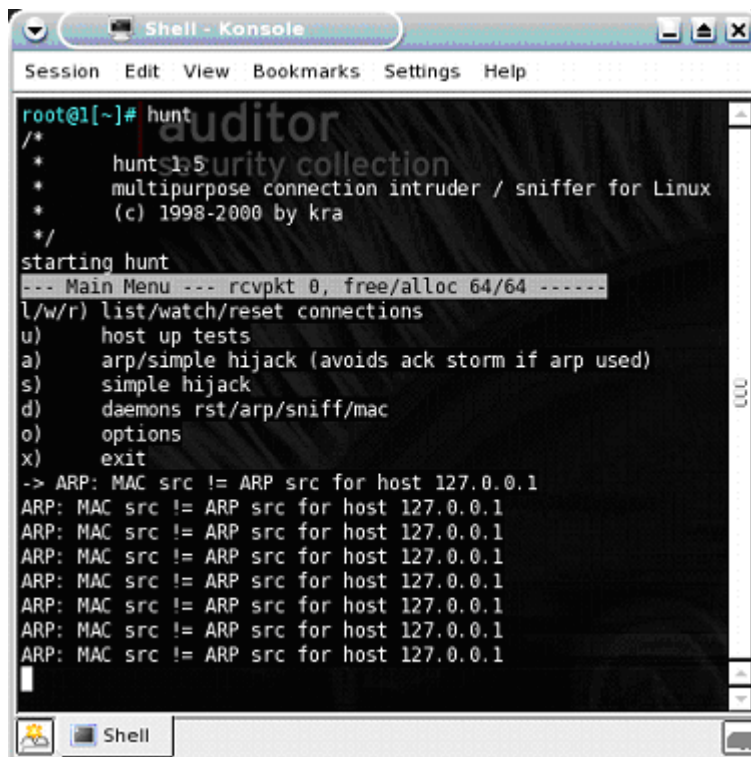
As for the poor Administrator? The command prompt window that they are using to create a Telnet session hangs while the session is being hijacked. After a minute or so the window will display a message 'Connection to host lost.' What would most Administrators do? Blame Windows and re-establish the session!! What will you do from now on?

Hunt

Hunt, by Pavel Krauz is a Linux open source attack tool that performs session hijacking. The fundamental difference between Hunt and most other hijackers is that it can hand back the victim session. To hand back the session, the attacker needs to resynchronize the client sequence number to match the server. Hunt tries to solve this problem by sending a message to the logged on client user. One of those messages is shown below:

```
#msg from root: power failure - try to type 17 chars
```

The number of characters that needs to be entered is entirely dependant on the difference between the client sequence number and the server sequence number. Hunt will replace this value with whatever number of bytes is required, 1 character = 1 byte. The crux of this is if the user will obey the instruction. When the user has typed enough characters and therefore transmitted enough bytes to synchronize, Hunt will then transmit ARP update packets to restore the correct values to the ARP table entries it modified on the client and server. This technique will probably not work against well-educated users or any protocol other than Telnet or possibly FTP, both text based unencrypted protocols.



```
root@l[~]# hunt
/*
 *   hunt: security collection
 *   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
-> ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
ARP: MAC src != ARP src for host 127.0.0.1
```

Session Hijacking Packet Analysis

Lee Lawson

Countermeasures

To defend against session hijack attacks, a network should employ several defenses. The most effective is encryption such as 'IPSEC'. Internet Protocol Security has the ability to encrypt your IP packets based on a Pre-Shared Key or with more complex systems like a Public Key Infrastructure 'PKI'. This will also defend against many other attack vectors such as sniffing. The attacker may be able to passively monitor your connection, but they will not be able to read any data as it is all encrypted. There might be actions an attacker could take against an IPSEC enabled network, depending on if they use IKE-PSK or PKI to manage the encryption keys, but this would require an experienced hacker. Don't think that IPSEC is the panacea to all your ills, there are IPSEC cracking tools available on the internet that will attempt to guess the PSK and decrypt packets.

Other countermeasures include encrypted applications like SSH (Secure SHell, an encrypted Telnet) or SSL (Secure Sockets Layer, HTTPS traffic). Again this reflects back to using encryption, but a subtle difference being that you are using the encryption within an application. Be aware though that there are known attacks against SSH and SSL. OWA, Outlook Web Access uses SSL to encrypt data between an internet client browser and the Exchange mail server, but tools like Cain & Abel (my favorite Windows based attack tool!) can spoof the SSL certificate and mount a Man-In-The-Middle (MITM) attack and decrypt everything!

Reducing your 'Attack Surface' (the potential methods of gaining access to your network) will help, e.g. eliminate remote access to the internal systems. By cutting out authorized remote connections, you have removed the potential for somebody to attack those remote connections. If you have remote users that need to connect to carry out their duties, then use 'Virtual Private Networks' that have been secured with tunneling protocols and encryption, L3TP/PPTP & IPSEC.

Again, a defense in depth approach is always the best countermeasure to any potential threat. Employing any one countermeasure may not be enough, but using them together to secure your enterprise will make the success rate of any attack minimal to anyone but the most professional and dedicated attacker. Remember, no computer system is every 100% secure! (Unless it is powered off and dropped in the ocean!!)

Summary

TCP session hijacking is a very dangerous attack vector because most systems are vulnerable to it, as most systems use TCP/IP as their primary communication protocol. Newer operating systems have attempted to secure themselves from session hijacking by using pseudo-random number generators to calculate the Initial Sequence Number, making it harder to guess. Any security measure in randomly generating an ISN is ineffective if the attacker is able to sniff ACK packets, as they give all the information required to perform this attack. The 'Strange Attractors and TCP/IP Sequence Number Analysis' document gives great information about the PRNG, Pseudo-Random Number Generators that different Operating System's use.

Tools such as T-Sight, Hunt, Juggernaut and TTY Watcher have been released to mount these attack vectors and all are effective in the right environment.

Session Hijacking Packet Analysis

Lee Lawson

T-Sight is an easy to use, point and click Windows application for local session hijacking. It is limited to the IP address range that you purchase from Engarde.

Hunt is a *nix based tool for monitoring, hijacking and resetting TCP sessions.

Juggernaut is a session hijacker that can be configured to listen for keywords, such as an administrator username.

IP-Watcher is a network security monitor for UNIX with active countermeasures. It allows the administrator to monitor and control intruders in real-time.

TTY-Watcher is a host security monitor with active countermeasures. It allows the administrator to monitor and control users in real-time on a single host. Based on IP-Watcher, this program is limited to a single (Solaris) machine rather than an entire network.

Reduce your attack surface. Use encryption. Do not use Telnet!