Chapter 11: Session Hijacking

Introduction

The session hijacking concept is an interesting topic for several different scenarios. It is the hijacking of sessions by intercepting the communication between hosts. The attacker usually interrupts communications to undertake the role of an authenticated user or to carry out a "Man-in-the-Middle" attack.

In hijacking attacks, a hacker downloads malicious code to a site that the original user frequently visits and then forces the victim's system to transfer session cookie data to the hacker's server. Once the attacker has obtained a user's session ID, they can impersonate a valid user on any number of web services that successfully handshake with the session ID.

The impact of a session hijacking attack can be severe, depending on the sensitivity of the data revealed and the significance of the application being accessed. A successful attack could result in financial fraud, identity theft, data breaches, etc. There are several rules, techniques, and best practices to prevent these attacks for securing applications as the threat landscape constantly evolves.

In this chapter, you will explore:

- Session hijacking concepts and their role in cyberattacks
- Common techniques used to intercept and hijack active sessions
- Methods attackers use to steal or guess session IDs
- Tools commonly employed in session hijacking attacks
- Impacts of session hijacking on organizations and users
- Best practices and countermeasures to defend against session hijacking

Session Hijacking Concept

Familiarization with basic concepts related to session hijacking is important to attain a comprehensive understanding. This section explains what session hijacking is as well as the reasons why session hijacking succeeds. It also discusses the session hijacking process, packet analysis of a local session hijack, types of session hijacking, session hijacking in an Open Systems Interconnection (OSI) model, and differences between spoofing and hijacking.

What Is Session Hijacking?

After successful authentication, a web server sends a session identification token or key to a web client. These session tokens differentiate multiple sessions that the server establishes with clients. Web servers use various mechanisms to generate random tokens and controls to secure the tokens during transmission.

Session hijacking is an attack in which an attacker takes over a valid Transmission Control Protocol (TCP) communication session between two computers. Because most types of authentication are performed only at the start of a TCP session, an attacker can access a machine while a session is in

progress. Attackers can sniff all the traffic from established TCP sessions and perform identity theft, information theft, fraud, etc.

A session hijacking attack exploits a session-token generation mechanism or token security controls to establish an unauthorized connection with a target server. The attacker can guess or steal a valid session ID, which identifies authenticated users, and use it to establish a session with the server. The web server responds to the attacker's requests under the impression that it is communicating with an authenticated user.

Attackers can use session hijacking to launch various kinds of attacks, such as Man-In-The-Middle (MITM) and Denial-of-Service (DoS) attacks. In an MITM attack, an attacker places themselves between an authorized client and a server by performing session hijacking to ensure that information flowing in either direction passes through them. However, the client and server believe they are directly communicating with each other. Attackers can also sniff sensitive information and disrupt sessions to launch a DoS attack.

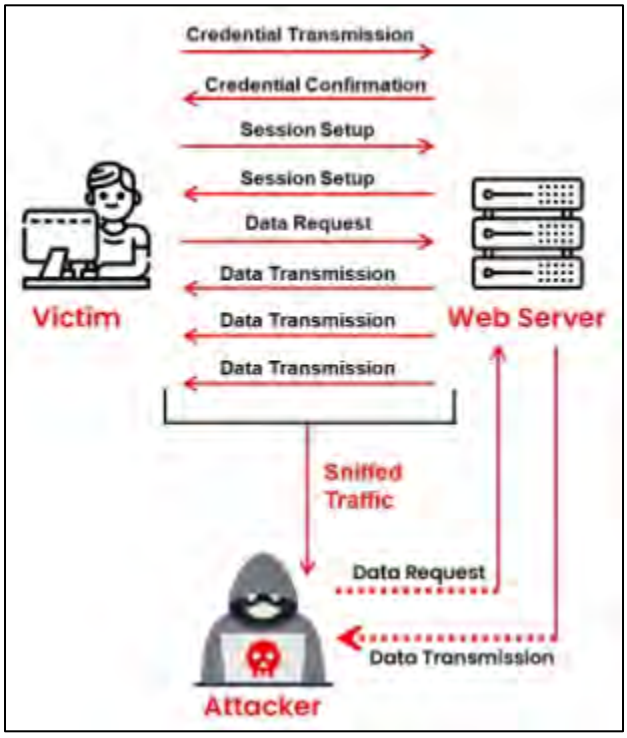

Figure 11-01: Session Hijacking Concept

Why Is Session Hijacking Successful?

Session hijacking succeeds because of the following factors.

Absence of Account Lockout for Invalid Session IDs

If a website does not implement account lockout, an attacker can make several attempts to connect with varying session IDs embedded in a genuine URL and continue until the actual session ID is determined. This attack is also known as a brute-force attack. During a brute-force attack, the web server does not display a warning message or complaint, allowing the attacker to determine the valid session ID.

Weak Session-ID Generation Algorithm or Small Session IDs

Most websites use linear algorithms to predict variables such as time or IP address for generating session IDs. By studying the sequential pattern and generating multiple requests, an attacker can easily narrow the search space necessary to forge a valid session ID. Even if a strong session-ID generation algorithm is used, an active session ID can be easily determined if the string is short.

Insecure Handling of Session IDs

An attacker can retrieve stored session-ID information by misleading the user's browser into visiting another site. Before the session expires, the attacker can exploit the information in many ways, such as Domain Name System (DNS) poisoning, cross-site scripting exploitation, and the exploitation of a bug in the browser.

The Session Hijacking Process

The process of session hijacking involves:

- **Sniffing** - An attacker attempts to place themself between the victim and the target to sniff the packet
- **Monitoring** - An attacker monitors the traffic flow between the victim and the target
- **Session Desynchronization** - This is the process of breaking the connection between the victim and the target
- **Session ID** - An attacker takes control of the session by predicting the session ID
- **Command Injection** - After successfully taking control of the session, the attacker starts inserting commands

Session Hijacking Techniques

The following are the techniques of session hijacking:

Stealing

There are various techniques for stealing a session ID, such as Referrer Attack, Network Sniffing, Trojans, etc.

Guessing

Guessing is using tricks and techniques to guess the session ID; for example, observing the variable components of session IDs or calculating the valid session ID by figuring out the sequence.

Brute-Forcing

Brute-Forcing is the process of guessing every possible combination of credentials. It is usually performed when an attacker has obtained information about the session ID range.

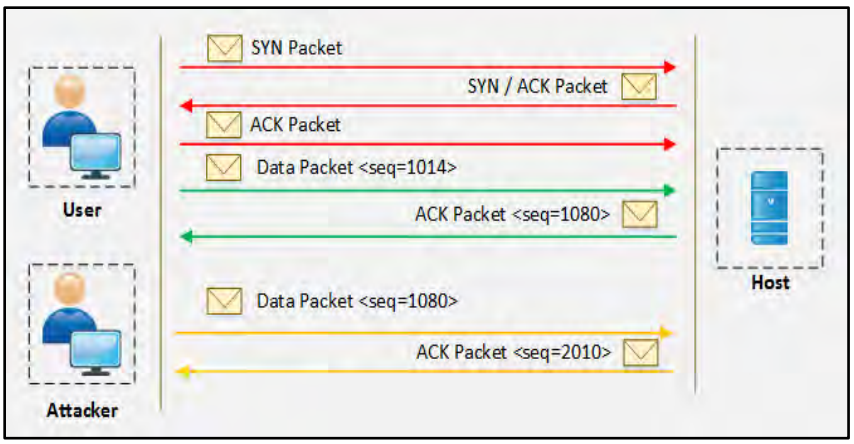

Figure 11-02: Brute Forcing

Packet Analysis of a Local Session Hijack

Session hijacking involves high-level attack vectors that affect many systems. Many systems that establish LAN or Internet connections use TCP to transmit data. The two systems should perform a three-way handshake to connect and transmit data successfully. Session hijacking involves the exploitation of this three-way handshake method to take control of the session.

To conduct a session hijacking attack, the attacker performs three activities:

- Tracking of a session
- Desynchronization of the session
- Injection of commands during the session

By sniffing network traffic, an attacker can monitor or track a session. The next step in session hijacking is to desynchronize the session. It is easy to accomplish this attack if the attacker knows the Next Sequence Number (NSN) the client uses. A session can be hijacked by using that sequence number before the client uses it. There are two possibilities to determine sequence numbers: one is to sniff the traffic, find an ACK packet, and then determine the NSN based on the ACK packet. The other is to transmit data with guessed sequence numbers, which is not a reliable method. If the attacker can access the network and sniff the TCP session, they can easily determine the sequence number. This type of session hijacking is called local session hijacking.

The Session Hijacking Process

The process of session hijacking involves:

- **Sniffing** - An attacker attempts to place themself between the victim and the target to sniff the packet.
- **Monitoring** - An attacker monitors the traffic flow between the victim and the target.
- **Session Desynchronization** - This is the process of breaking the connection between the victim and the target.
- **Session ID** - An attacker takes control of the session by predicting the session ID.
- **Command Injection** - After successfully taking control of the session, the attacker starts inserting commands.

Types of Session Hijacking

Active Attack

An Active Attack involves the attacker actively intercepting the active session. In an active attack, the attacker may send packets to the host. The attacker manipulates legitimate users of the connection. Once the active attack is successful, the legitimate user becomes disconnected from the attacker.

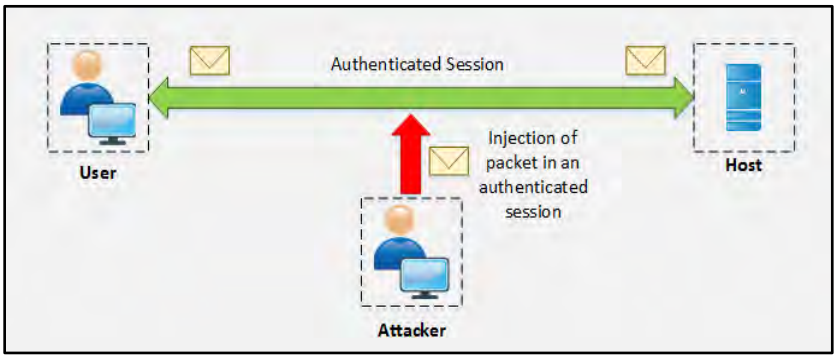

Figure 11-03: Active Attack

Passive Attack

A passive attack involves hijacking a session and monitoring the communication between hosts without sending any packets.

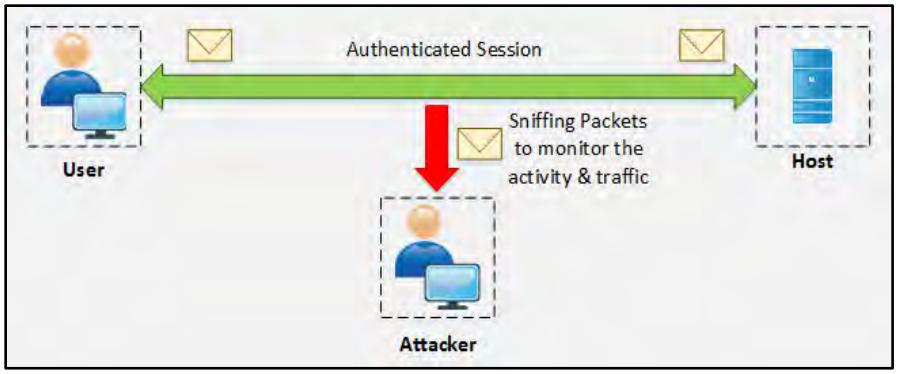

Figure 11-04: Passive Attack

Session Hijacking in OSI Model

Network Level Hijacking

Network Level Hijacking involves hijacking a network layer session, such as a TCP or UDP session.

Application Level Hijacking

Application Level Hijacking involves hijacking an Application layer, such as an HTTPS session.

Network-Level Hijacking and Application-Level Hijacking are discussed in detail later in this chapter.

Spoofing vs. Hijacking

The major difference between Spoofing and Hijacking is an active session. In a spoofing attack, the attacker impersonates another user to gain access. The attacker has no active session but initiates a new session with the target with the help of stolen information.

Hijacking is the process of taking control of an existing active session between an authenticated user and a targeted host. The attacker uses the authenticated, legitimate user's session without initiating a new session with the target.

Application Level Session Hijacking

Session hijacking focuses on the application layer of the OSI model. In the application layer hijacking process, the attacker seeks a legitimate session ID from the victim to access an authenticated session that allows the attacker to use web resources. With application layer hijacking, an attacker can access the website resources secured for authenticated users. The web server may assume that the incoming requests are from a known host when the session has been hijacked by an attacker, usually by predicting the session ID.

How To Predict A Session Token?

Web servers normally use random session ID-generating tools to prevent prediction. However, some web servers use customer-defined algorithms to assign a session ID. Some examples are shown below:

```
http://www.example.com/ABCD0 10 120 17 19 17 10
http://www.example.com/ABCD0 10 120 17 19 1750
http://www.example.com/ABCD0 10 120 17 19 1820
http://www.example.com/ABCD0 10 120 17 1920 10
```

After observing the above session IDs, the constant and variable parts can easily be identified. In the above example, **ABCD** is the constant part, **0 10 120 17** is the date, and the last section is the time. An attacker may attempt the following session ID at 19:25:10

```
http://www.example.com/ABCD0 10 120 17 1925 10
```

Compromising Session IDs

Compromising Session IDs Using Sniffing

Session sniffing is a technique in which an attacker sniffs for the session ID/Token. Once the attacker finds the session ID, they can gain access to the resources.

Compromising Session IDs Using a Man-In-The-Middle Attack

Compromising the session ID using a Man-In-The-Middle attack requires splitting the connection between the victim and web server into two connections, one between the victim and attacker and another between the attacker and the server.

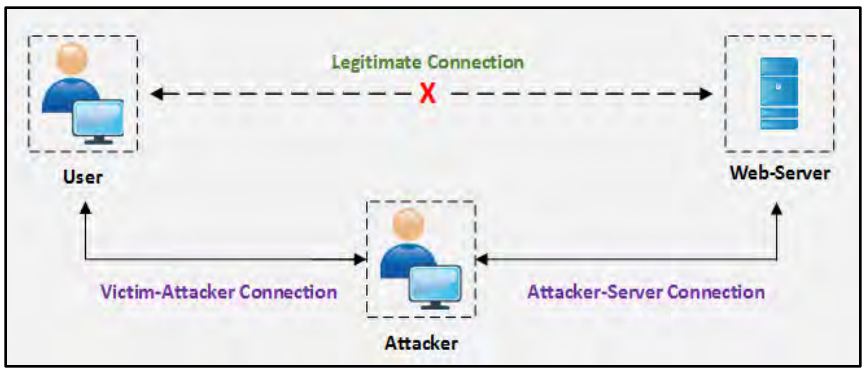

Figure 11-05: MITM Process

Note: Ettercap is a comprehensive suite for Man-In-The-Middle attacks. It helps sniff time connections, content filtering, and active and passive dissection of many protocols. Ettercap includes many networks and host analysis features.

Compromising Session IDs Using a Man-in-the-Browser Attack

Compromising a session ID using a Man-in-the-Browser attack requires a Trojan deployed on the target machine. The Trojan can change the proxy settings or redirect all traffic through the attacker. Another Trojan technique is intercepting the process between the browser and its security mechanism.

Steps to Performing a Man-in-the-Browser Attack

The attacker first infects the victim's machine using a Trojan to launch a Man-in-the-Browser Attack. The Trojan installs malicious code on the victim's machine as an extension that modifies the browser's configuration upon boot. When a user logs in to a site, the URL is checked against a known list of the targeted websites. The event handler registers the event upon detection. Using a DOM interface, an attacker can extract and modify the values when the user clicks the button. The browser will send the form with modified entries to the webserver. The user cannot identify interception as the browser shows the original transaction details.

Compromising Session IDs By Predicting Session Token

Predicting session ID is observing a client's currently occupied session ID. An attacker can guess the next session key by observing its common and variable parts.

Compromising Session IDs Using Client-side Attacks

Session IDs can be compromised easily by using Client-side attacks such as:

1. Cross-Site Scripting (XSS)
2. Malicious JavaScript Code
3. Trojans

Cross-site Script Attacks

An attacker performs a Cross-site Scripting Attack by sending a crafted link with a malicious script. When the user clicks the malicious link, the script is executed. This script might be coded to extract and send the session IDs to the attacker.

Cross-site Request Forgery Attack

A Cross-site Request Forgery (CSRF) attack is obtaining a legitimate user's session ID and exploiting the active session with a trusted website to perform malicious activities.

Compromising Session IDs Using Session Replay Attack

Another technique for session hijacking is the Session Replay Attack. Attackers capture users' authentication tokens intended for the server and replay the request to the server, resulting in unauthorized access to the server.

Compromising Session IDs Using Session Fixation

Session Fixation is an attack permitting the attacker to hijack the session. The attacker has to provide a valid session ID and make a victim's browser use it. The following techniques do this:

1. Session Token in the URL argument
2. Session Token in hidden form
3. Session ID in a cookie

Consider the scenario of a Session Fixation Attack where an attacker, a victim, and the web server are connected to the internet. The attacker initiates a legitimate connection with the webserver, issues a session ID, or uses a new one. The attacker then sends the link to the victim with the established session ID to bypass the authentication. When the user clicks the link and attempts to log in to the website, the web server continues the session as it is already established and authenticated. Now, the attacker has the session ID information and continues using a legitimate user account.

Session Hijacking

Session Hijacking Using Proxy Servers

An attacker lures the victim to click on a fake link, which appears legitimate but redirects the user to the attacker's server. The attacker then forwards the request to the legitimate server on behalf of the victim and serves as a proxy for the entire transaction. Acting as a proxy, the attacker captures the session information during the interaction between the legitimate server and the user.

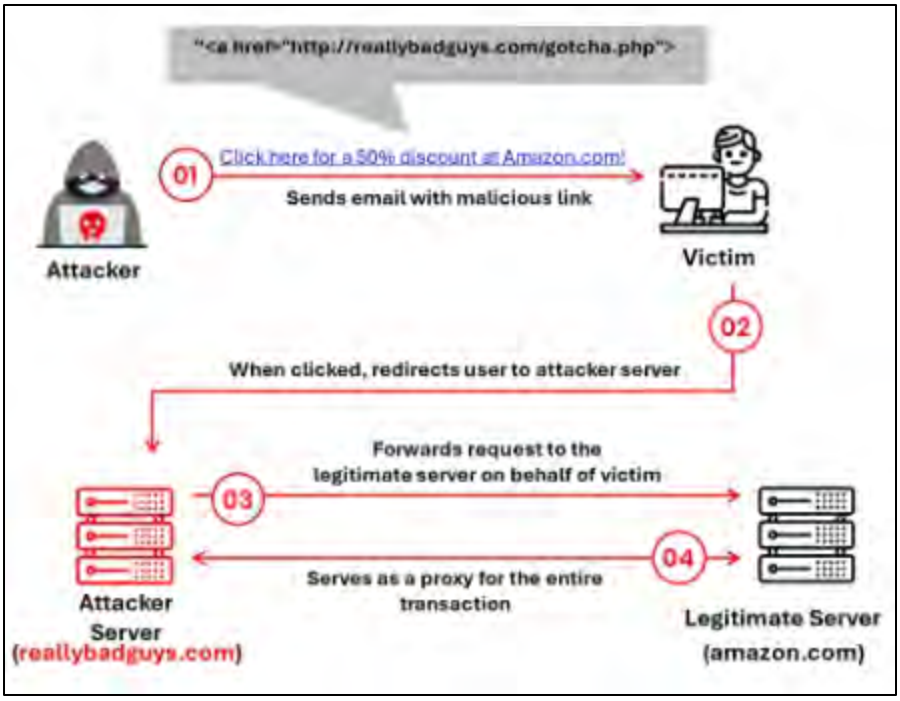

Figure 11-06: Session Hijacking Using Proxy Servers

Session Hijacking Using CRIME Attack

Compression Ratio Info-Leak Made Easy (CRIME) is a client-side attack that exploits vulnerabilities in the data-compression feature of protocols such as SSL/Transport Layer Security (TLS), SPDY, and HTTP Secure (HTTPS). The possibility of mitigation against HTTPS compression is low, which makes this vulnerability even more dangerous than other compression vulnerabilities.

When two hosts on the Internet establish a connection using HTTPS, a TLS session is established, and the data are transmitted in an encrypted form. Hence, it is difficult for an attacker to read or modify the messages between the two hosts. Authentication data are stored in a cookie when a user logs into a web application. Whenever the browser sends an HTTPS request to the web application, the stored cookie is used for authentication. In this attack, the attacker attempts to access the authentication cookie to hijack the victim's session.

In HTTPS, cookies are compressed using a lossless data compression algorithm (DEFLATE) and then encrypted. Hence, it is difficult for an attacker to obtain the cookie's value with simple sniffing.

To perform a CRIME attack, an attacker must use social engineering techniques to trick the victim into clicking on a malicious link. When the victim clicks on the malicious link, it either injects malicious code into the victim's system or redirects the victim to a malicious website. If the victim has already established an HTTPS connection with a secured web application, the attacker sniffs the victim's HTTPS traffic using techniques such as ARP spoofing. Through sniffing, the attacker captures the cookie value from the HTTPS messages and sends multiple HTTPS requests to the web application, with that cookie prepended with a few random characters. Subsequently, the attacker monitors the traffic between the victim and the web application to obtain the compressed and encrypted value of the cookie. After capturing the cookie, the attacker analyzes the cookie length

and predicts the actual value of the authentication cookie. After obtaining the authentication cookie, the attacker impersonates the victim and hijacks the victim's session with the secure web application to steal confidential information such as passwords, social security numbers, and credit card numbers. Attackers use tools such as CrimeCheck to detect whether a web server has TLS or HTTP compression enabled and are thus vulnerable to CRIME attacks.

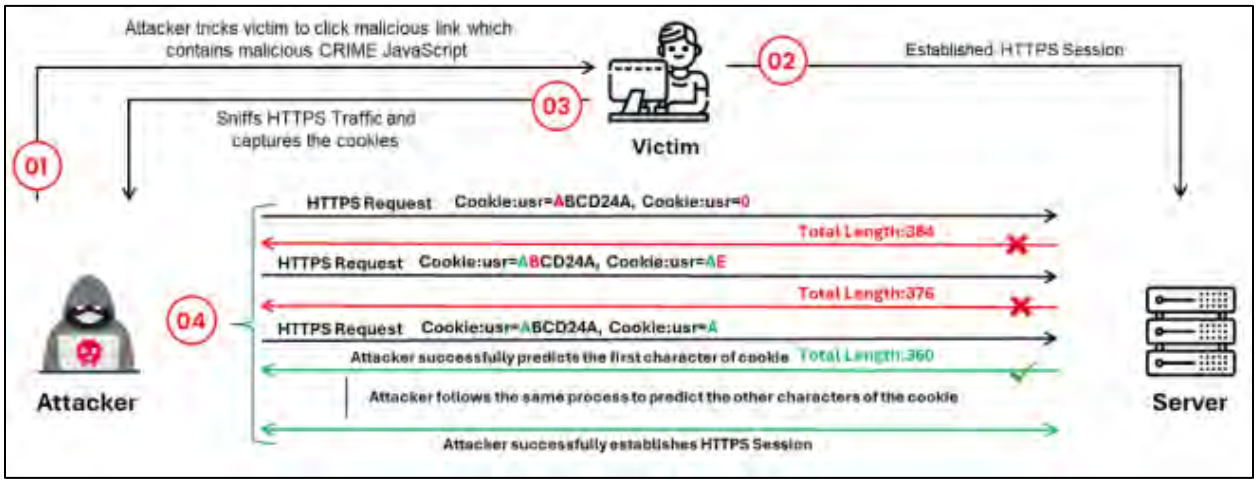

Figure 11-07: Session Hijacking Using CRIME Attack

Session Hijacking Using Forbidden Attack

A forbidden attack is an MITM attack that can be executed when a cryptographic nonce is reused while establishing an HTTPS session with a server. According to the TLS specification, these arbitrary pieces of data must be used once. This attack exploits the vulnerability that the TLS implementation incorrectly reuses the same nonce when data are encrypted using the Advanced Encryption Standard–Galois/Counter Mode (AES-GCM) during the TLS handshake. Attackers exploit this vulnerability to perform an MITM attack by generating cryptographic keys used for authentication. Repeating the same nonce during the TLS handshake allows an attacker to monitor and hijack the connection. After hijacking the HTTPS session and bypassing the protection, attackers inject malicious code and forged content into the transmission, such as JavaScript code or web fields that prompt users to disclose passwords, social security numbers, or other confidential information. A forbidden attack involves the following steps.

- The attacker monitors the connection between the victim and web server and sniffs the nonce from the TLS handshake messages
- The attacker generates authentication keys using the nonce and hijacks the connection
- All the traffic between the victim and web server flows through the attacker's machine
- The attacker injects JavaScript code or web fields into the transmission towards the victim
- The victim reveals sensitive information such as bank account numbers, passwords, and social security numbers to the attacker

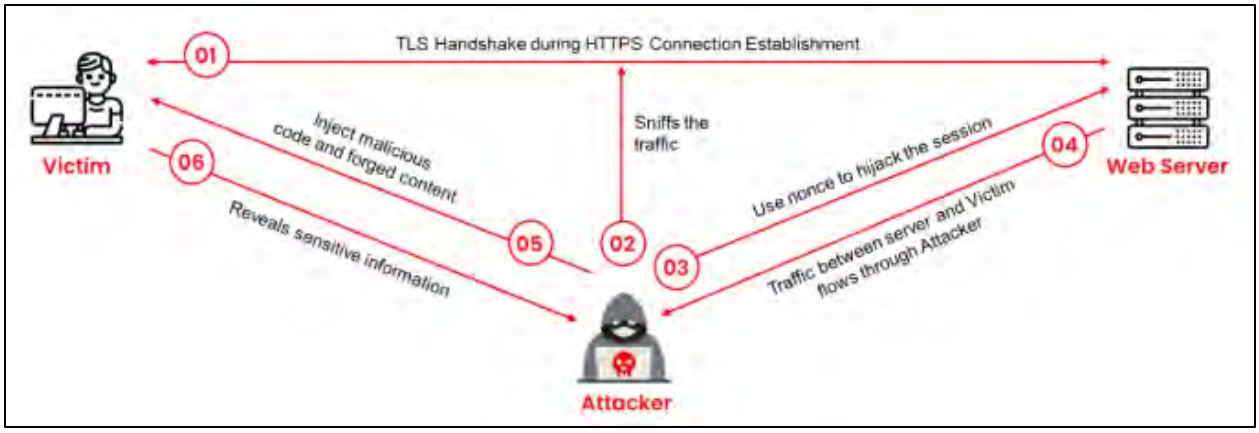

Figure 11-08: Session Hijacking Using Forbidden Attack

Session Hijacking Using Session Donation Attack

In a session donation attack, the attacker donates their own session ID to the target user. In this attack, the attacker first obtains a valid session ID by logging into a service and later feeds the same session ID to the target user. This session ID links a target user to the attacker's account page without disclosing any information to the victim. When the target user clicks on the link and enters the details (username, password, payment details, etc.) in a form, the entered details are linked to the attacker's account. To initiate this attack, the attacker can send their session ID using techniques such as cross-site cooking, an MITM attack, and session fixation.

A session donation attack involves the following steps.

- First, the attacker logs into a service, establishes a legitimate connection with the target web server, and deletes the stored information
- The target web server (e.g., http://citibank.com/) issues a session ID, say 0D6441FEA4496C2, to the attacker
- The attacker then donates their session ID, say http://citibank.com/?SID=0D6441FEA4496C2, to the victim and lures the victim to click on it to access the website
- The victim clicks on the link, believing it to be a legitimate link sent by the bank. This opens the server's page in the victim's browser with SID=0D6441FEA4496C2. Finally, the victim enters their information in the page and saves it.
- The attacker can now login as themselves and acquire the victim's information

Network Level Session Hijacking

Network Level Hijacking focuses on the Transport layer and Internet layer protocols used by the application layer. A network-level attack extracts information that might be helpful for the application layer session.

There are several types of network-level hijacking, including:

- Blind Hijacking
- UDP Hijacking
- TCP/IP Hijacking

- RST Hijacking
- MITM
- IP Spoofing

The Three-Way Handshake

TCP communication initiates with a three-way handshake between the requesting and the target host. This handshake communicates Synchronization (SYN) packets and Acknowledgment (ACK) packets. Figure 11-09 illustrates the flow of a three-way handshake.

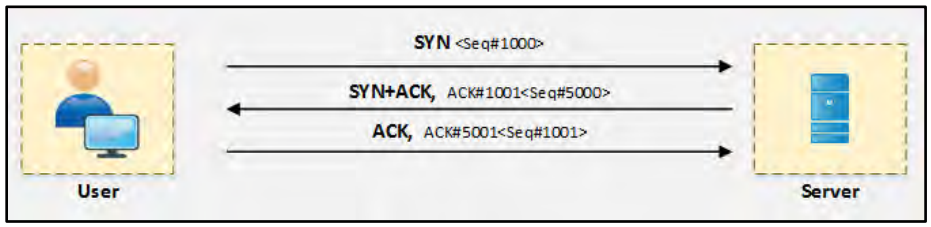

Figure 11-09: The Three-way Handshake

TCP/IP Hijacking

The TCP/IP Hijacking process is a network-level attack on a TCP session in which an attacker predicts the sequence number of packets flowing between the victim and host. To perform a TCP/IP attack, the attacker must be on the same network as the victim. Usually, the attacker uses sniffing tools to capture the packets and extract the sequence number. By injecting the spoofed packet, the attacker can interrupt a session. A denial-of-service attack or a reset connection can disrupt communication with the legitimate user.

Source Routing

Source routing is a technique of sending a packet via a selected route. In session hijacking, this technique is used to attempt IP spoofing as a legitimate host with the help of source routing to direct traffic through a path identical to the victim's path.

RST Hijacking

RST hijacking is the process of sending a Reset (RST) packet to the victim with a spoofed source address. The acknowledgment number used in this reset packet is also predicted. When the victim receives this packet, they will not be aware that it is spoofed. The victim resets the connection assuming that an actual source requested the connection reset request. An RST packet can be crafted using packet designing tools.

Blind Hijacking

Blind Hijacking is a technique used when an attacker is unable to capture the return traffic. In blind hijacking, the attacker captures a packet coming from the victim and heading toward the server, injects a malicious packet, and forwards it to the targeted server.

Forged ICMP and ARP Spoofing

A man-in-the-middle attack can also be carried out using a Forged ICMP Packet and ARP Spoofing techniques. Forged ICMP packets, such as *destination unavailable* or *high latency messages,* are sent to fool the victim.

UDP Hijacking

The UDP Session Hijacking process is simpler than TCP session hijacking. Since the UDP is a connectionless protocol, it does not require any sequence packet between the requesting client and host. UDP session hijacking is all about sending a response packet before the destination server responds. There are several techniques to intercept the coming traffic from the destination server.

Session Hijacking Tools

Attackers can use tools such as Hetty, OWASP ZAP, and bettercap to hijack a session between a client and a server. The following are some additional session hijacking tools:

- Burp Suite (https://portswigger.net)
- OWASP ZAP (https://www.zaproxy.org)
- WebSploit Framework (https://sourceforge.net)
- sslstrip (https://pypi.org)
- JHijack (https://sourceforge.net)

Hetty

Source: https://github.com

Hetty is an HTTP toolkit for security research. It provides the following features:

- Machine-In-The-Middle (MITM) HTTP proxy with logs and advanced search
- HTTP client for manually creating/editing requests and replaying proxied requests
- Intercepting requests and responses for a manual review (edit, send/receive, and cancel)

Caido

Source: https://caido.io

Caido is a web security auditing toolkit that security professionals can use to intercept and view HTTP requests in real time while browsing. It allows customization and testing of requests against large wordlists, automatically modifies incoming requests using Regex rules, and resends requests to manually test endpoints.

bettercap

Source: https://www.bettercap.org

bettercap is a portable framework written in Go that allows security researchers, red teamers, and reverse engineers to perform reconnaissance and various attacks on Wi-Fi networks, Bluetooth low energy devices, wireless HID devices, and IPv4/IPv6 networks.

Session Hijacking Countermeasures

Several detection techniques and countermeasures can be implemented to mitigate session hijacking attacks. These can be manual or automated. Deployment of defense-in-depth technology and network monitoring devices such as the Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) are automated detection processes. Several packet sniffing tools are available that can be used for manual detection.

In addition, encrypted sessions and communication using Secure Shell (SSH), HTTPS instead of HTTP, random and lengthy strings as session IDs, session timeout, and strong authentication like Kerberos can help prevent and mitigate session hijacking. IPsec and SSL can also be used to provide stronger protection against hijacking.

Several detection techniques and countermeasures can be implemented to mitigate session hijacking attacks. These can be manual or automated. Deployment of defense-in-depth technology and network monitoring devices such as the Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) are automated detection processes. Several packet sniffing tools are available that can be used for manual detection.

In addition, encrypted sessions and communication using Secure Shell (SSH), HTTPS instead of HTTP, random and lengthy strings as session IDs, session timeout, and strong authentication like Kerberos can help prevent and mitigate session hijacking. IPsec and SSL can also provide stronger protection against hijacking.

Session Hijacking Detection Methods

Session hijacking attacks are exceptionally difficult to detect, and users often overlook them unless the attacker causes severe damage.

The following are some symptoms of a session hijacking attack:

- A burst of network activity for some time, which decreases the system performance
- Busy servers resulting from requests sent by both the client and hijacker

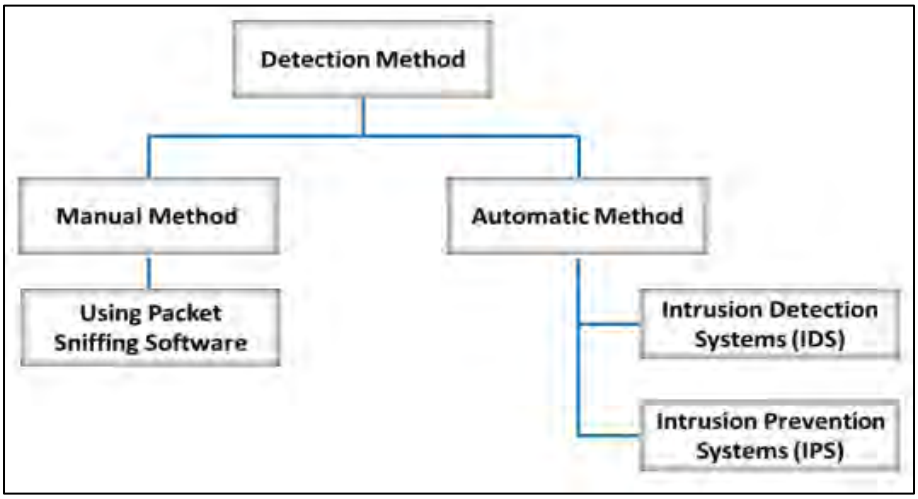

Figure 11-10: Session Hijacking Detection Methods

Manual Method

The manual method involves the use of packet sniffing software such as Wireshark and SteelCentral Packet Analyzer to monitor session hijacking attacks. The packet sniffer captures packets in transit across the network, which is then analyzed using various filtering tools.

Forced ARP Entry

A forced ARP entry involves replacing the MAC address of a compromised machine in the ARP cache of the server with a different one in order to restrict network traffic to the compromised machine.

A forced ARP entry should be performed in the case of the following:

- Repeated ARP updates o Frames sent between the client and server with different MAC addresses
- ACK storms

Automatic Method

The automatic method involves the use of an Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) to monitor incoming network traffic. If the packet matches any of the attack signatures in the internal database, the IDS generates an alert, whereas the IPS blocks the traffic from entering the database.

Session Hijacking Detection Tools

Session hijacking attacks are difficult to detect, and in most cases, attacks go unnoticed, causing severe leakage of confidential data. Tools such s packet sniffers, IDSs, and Security Information And Event Management (SIEM) can be used to detect session hijacking attacks.

The following are some additional session hijacking detection tools:

- Quantum Intrusion Prevention System (IPS) (https://www.checkpoint.com)
- SolarWinds Security Event Manager (https://www.solarwinds.com)
- IBM Security Network Intrusion Prevention System (https://www.ibm.com)
- LogRhythm (https://logrhythm.com)

USM Anywhere

Source: https://cybersecurity.att.com

USM Anywhere offers powerful threat detection, incident response, and compliance management across cloud, on-premises, and hybrid environments. Security professionals can use this tool for detecting session hijacking attempts and perform asset discovery, intrusion detection, security automation, SIEM and log management, endpoint detection and response, threat detection, threat intelligence, and vulnerability assessment.

Wireshark

Source: https://www.wireshark.org

Wireshark allows users to capture and interactively browse the traffic on a network. This tool uses Winpcap to capture packets. Therefore, it can only capture packets on the networks supported by Winpcap. It captures live network traffic from Ethernet, IEEE 802.11, Point-to-Point Protocol/High-level Data Link Control (PPP/HDLC), Asynchronous Transfer Mode (ATM), Bluetooth, Universal Serial Bus (USB), Token Ring, Frame Relay, and Fiber Distributed Data Interface (FDDI) networks. Security professionals use Wireshark to monitor and detect session hijacking attempts.

Protecting Against Session Hijacking

- Use the Secure Shell (SSH) to create a secure communication channel
- Pass authentication cookies over HTTPS connections
- Implement the log-out functionality for the user to end the session
- Generate a session ID after a successful login and accept session IDs generated by the server only
- Ensure that data in transit is encrypted and implement the defense-in-depth mechanism
- Use strings or long random numbers as session keys
- Use different usernames and passwords for different accounts
- Educate employees and minimize remote access
- Implement timeout mechanism to destroy sessions when expired
- Avoid including the session ID in the URL or query string
- Switch from a hub network to a switch network to reduce the risk of ARP spoofing and other session hijacking attacks
- Ensure that client-side and server-side protection software are in the active state and up-to-date
- Use strong authentication (such as Kerberos) or peer-to-peer virtual private networks (VPNs)
- Configure appropriate internal and external spoof rules on gateways
- Use IDS products or ARPwatch for monitoring ARP cache poisoning
- Use encrypted protocols available in OpenSSH suite
- Use firewalls and browser settings to confine cookies
- Protect authentication cookies with Secure Sockets Layer (SSL)
- Regularly update platform patches to fix TCP/IP vulnerabilities (e.g., predictable packet sequences)
- Use IPsec to encrypt session information
- Enable browsers to verify website authenticity using network notary servers
- Implement DNS-based authentication of named entities
- Disable compression mechanisms of HTTP requests
- Restrict cross-site scripts to prevent Cross-Site Request Forgery (CSRF) from the client side
- Upgrade web browsers to the latest versions
- Use vulnerability scanners to detect any insecure configuration of HTTPS session settings on sites
- Enable the HTTPOnly property to prohibit user scripts from accessing the cached cookies
- Use SSH File Transfer Protocol (SFTP), Applicability Statement 2 (AS2)-managed file transfer, or FTP Secure (FTPS) to send data using encryption and digital certificates
- Employ the Microsoft-based solution (SMB signing) to enable traffic signing
- Apply Secure Socket Layer (SSL) or Transport Layer Security (TLS) to decrease the likelihood of successful hijacks
- Implement IPsec to secure IP communications

- Use encrypted Virtual Private Networks (VPNs), such as PPTP and Layer 2 Protocol Tunneling (L2PT) for remote connections
- Use Multifactor Authentication (MFA) to reduce the chances of unauthorized access, even if a session token is compromised
- Use the SameSite cookie attribute to prevent the browser from sending cookies along with cross-site requests
- Monitor session activity for unusual patterns such as multiple simultaneous logins from different geographic locations, which may indicate a hijacked session
- Educate users on the importance of logging out of applications, especially on public or shared computers, and the need to use strong and unique passwords
- Bind the session to the user's IP address
- Leverage behavioral biometrics such as typing rhythms, mouse movements, and navigation patterns for continuous authentication
- Implement a challenge-response mechanism that requires solving a puzzle (e.g., CAPTCHA) when suspicious activity is detected
- Implement an absolute timeout regardless of session timeout when the user is inactive

Web Development Guidelines to Prevent Session Hijacking

An attacker usually hijacks a session by exploiting the vulnerabilities in mechanisms used for session establishment. Web developers often ignore security. During the development process, they should consider the following guidelines to minimize/eliminate the risk of session hijacking.

- Create session keys with lengthy strings or random numbers so that it is difficult for an attacker to guess a valid session key
- Regenerate the session ID after a successful login to prevent session fixation attacks
- Encrypt the data and session key transferred between the user and web servers
- Implement SSL to encrypt all information in transit via the network
- Make the session expire as soon as the user logs out
- Prevent eavesdropping within the network
- Reduce the life span of a session or cookie
- Use restrictive cache directives for all the web traffic through HTTP and HTTPS, such as the “Cache-Control: no-cache, no-store” and “Pragma: no-cache” HTTP headers and/or equivalent META tags on all or (at least) sensitive web pages
- Do not create sessions for unauthenticated users unless necessary
- Ensure HTTPOnly while using cookies for session IDs
- Use a secure flag to send cookies in HTTPS requests and encrypt them before sending across the network
- Check whether all the requests received for the current session originate from the same IP address and user agent
- Implement continuous device verification to identify whether the user who established the session is still in control
- Implement risk-based authentication at different levels before granting access to sensitive information

- Perform authentication and integrity verification between VPN endpoints
- Destroy the associated sessions on the server-side itself instead of simply depending on the session expiration when a user is deauthenticated
- Ensure that the web application is able to redirect HTTP requests to HTTPS using either server settings or redirection techniques
- Incorporate user re-authentication and generation of new sessions before allowing any sensitive functions
- Rely on web frameworks that provide highly secure session IDs to generate sessions instead of using own session management
- Enforce HTTPS on all pages of the web application, not just login pages

Web User Guidelines to Prevent Session Hijacking

The following are some guidelines for web users to defend against session hijacking.

- Do not click on links received through emails or Instant Messages (IMs)
- Use firewalls to prevent malicious content from entering the network
- Use firewalls and browser settings to restrict cookies
- Ensure that the website is certified by appropriate certifying authorities
- Ensure that the history, offline content, and cookies are cleared from the browser after every confidential and sensitive transaction
- Give preference to HTTPS, a secure transmission protocol, over HTTP when transmitting sensitive and confidential data
- Logout from the browser by clicking on the logout button instead of closing the browser
- Verify and disable add-ons from untrusted sites. Enable add-ons only if necessary
- Practice using a one-time password for critical data transactions (e.g., credit card transactions)
- Frequently update anti-virus signatures to prevent the automatic installation of malware that attempts to steal cookies
- Avoid accessing sensitive accounts or conducting financial transactions over public Wi-Fi
- Disable auto-connect to open Wi-Fi networks
- Ensure that your operating system, web browsers, and installed plugins are updated
- Use encrypted messaging and email services for sensitive communication
- Avoid saving passwords in browsers
- Use incognito mode on shared computers
- Be cautious about granting apps access to sensitive information or features on your device
- Use custom session handlers for storing and handling session tokens

Approaches To Prevent Session Hijacking

HTTP Strict Transport Security (HSTS)

HTTP Strict Transport Security (HSTS) is a web security policy that protects HTTPS websites against MITM attacks. The HSTS policy helps web servers force web browsers to interact with them using HTTPS. With the HSTS policy, all insecure HTTP connections are automatically converted into HTTPS connections. This policy ensures that all the communication between a web server and

web browser is encrypted and that all responses that are delivered and received originate from an authenticated server.

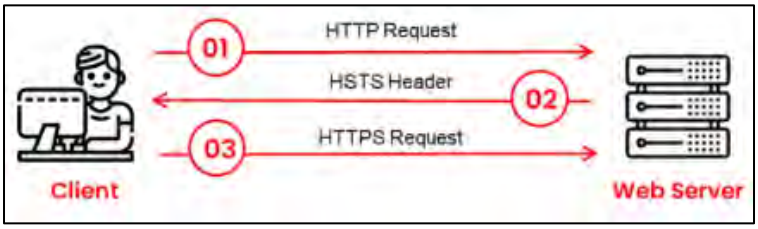

Figure 11-11: HTTP Strict Transport Security (HSTS)

Token Binding

When a user logs into a web application, a cookie with a session ID, called a token, is generated. The user utilizes this random token to send requests to the server and access resources. An attacker can impersonate the user and hijack the connection by capturing and reusing a valid session ID. Token binding protects client–server communication against session hijacking attacks. The client creates a public–private key pair for every connection to a remote server. When a client connects to the server, it generates a signature using a private key and sends this signature along with its public key to the server. The server verifies the signature using the client's public key. This ensures that the message was sent by an authentic client because only the client has its private key. Even if an attacker captures the signature, it is not possible for them to regenerate the signature or reuse it for another connection. For every new connection, a new pair of public and private keys are used.

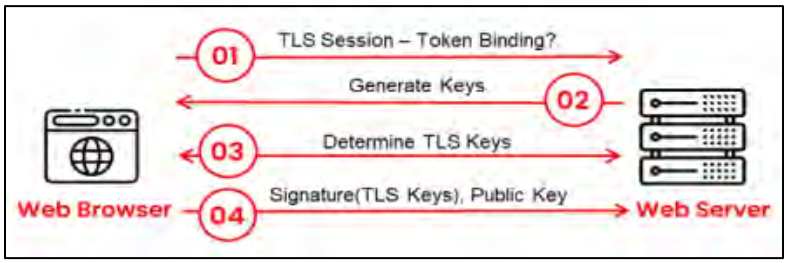

Figure 11-12: Token Binding

Session Hijacking Prevention Tools

To prevent session hijacking, the security testing of web applications and the analysis of static code to identify vulnerabilities in web applications are required. Identifying vulnerabilities at an early stage helps in implementing security measures to protect against session hijacking attacks.

The following are some additional session hijacking prevention tools:

- Nessus (https://www.tenable.com)
- Invicti (https://www.invicti.com)
- Wapiti (https://wapiti-scanner.github.io)

Checkmarx One SAST

Source: https://checkmarx.com

Checkmarx One SAST is a unique source-code analysis solution that provides tools for identifying, tracking, and repairing technical and logical flaws in source code, such as security vulnerabilities,

compliance issues, and business logic problems. CxSAST supports Open-Source Analysis (CxOSA), enabling licensing and compliance management, vulnerability alerts, policy enforcement, and reporting. This tool supports a wide range of OS platforms, programming languages, and frameworks.

Security professionals can use this tool to prevent various session hijacking attacks such as MITM attacks, session fixation attacks, and XSS attacks.

Fiddler

Source: https://www.telerik.com

Fiddler is used for performing web-application security tests such as the decryption of HTTPS traffic and manipulation of requests using an MITM decryption technique. Fiddler is a web debugging proxy that logs all HTTP(S) traffic between a computer and the Internet.

Security professionals can use Fiddler to test web applications by debugging the traffic from systems as well as manipulating and editing web sessions.

Approaches To Prevent MITM Attacks

Man-In-The-Middle (MITM) attacks are the most common type of attack, wherein the attackers intercept the traffic between two endpoints. The victim may not realize the effect of this attack, because it is mostly passive in nature. Because the detection of MITM attacks is difficult, they can only be prevented using various measures. The following are some approaches to prevent MITM attacks:

DNS Over HTTPS

DNS over HTTPS (DoH) is an enhanced version of the DNS protocol that is used to prevent the peeking or snooping of user's web activities or DNS queries during the DNS lookup process. The protocol is different than the conventional DNS protocol since the web queries and the traffic is sent through a secured or encrypted HTTPS tunnel via port 443. Implementing DNS over HTTPS makes the traffic undetectable by the attackers or ISPs since it gets hidden within the normal traffic passing through the HTTPS port.

Unlike the traditional DNS lookup process, the DoH sends a segment of a necessary domain name to fetch the results instead of sending the complete domain name entered by a user. This protocol helps in ensuring user's privacy and security as the web traffic is directed only between DoH supported clients and a resolver avoiding MITM and session hijacking attacks. Web browsers such as Chrome, Mozilla, and Microsoft Edge have been implementing this protocol for the past few years and Mozilla had already adopted this protocol as default from 2020 for its US clients.

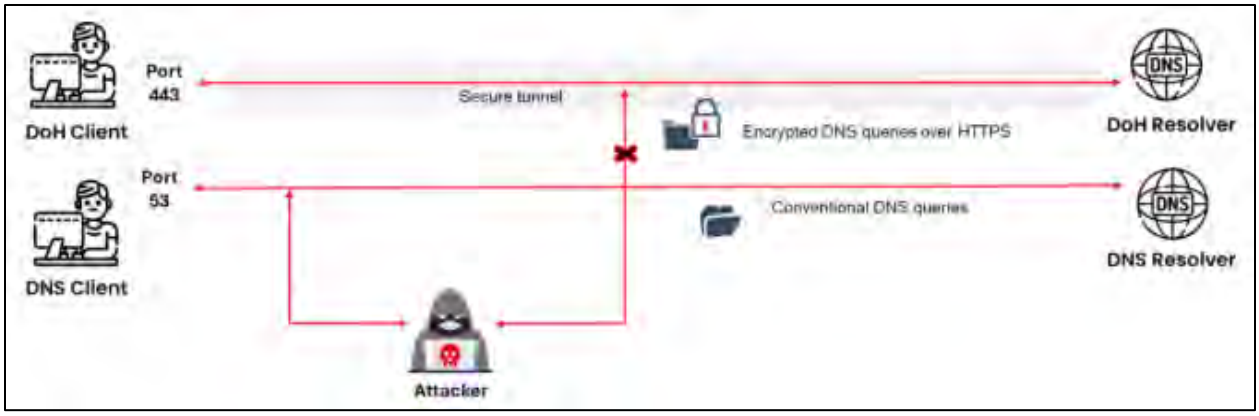

Figure 11-13: DNS Over HTTPS

WPA3 Encryption

Wireless Protected Access 3 (WPA3) is a wireless protocol intended to protect the traffic sent and received by users over a wireless network. The implementation of this protocol can prevent attempts by unwanted users to connect to a network. A weak encryption mechanism enables attackers to brute-force credentials and enter a target network to perform MITM attacks.

VPN

A VPN creates a safe and encrypted tunnel over a public network to securely send and receive sensitive information. It creates a subnet by using key-based encryption for secure communication between endpoints. The implementation of a VPN in the network prevents attackers from decrypting the data flowing between the endpoints.

Two-Factor Authentication

Two-factor authentication provides an extra layer of protection because it serves as a vector of authentication in addition to a user's password. Therefore, the implementation of two-factor authentication can prevent attackers from performing session hijacking and brute forcing to compromise a user's account.

Password Manager

Password Manager is an application or tool used to protect and manage individual credentials. The tool can also help in producing unique and complex passwords for web applications. Using the password manager, passwords can be stored in a secure location under the database and encapsulated using a master key to prevent MITM attacks.

Zero-trust Principles

Zero-trust principles constitute a set of standardized user pre-verification procedures that requires all users (inside or outside) to be authenticated before providing access to any resource. These principles are based on the famous phrase, "Trust but verify." Even though the request is made from the internal network, the authentication process is similar to that for an outsider.

Public Key Infrastructure (PKI)

Public key infrastructure (PKI) is a framework that manages, distributes, and validates digital certificates for secure communication. This ensures that the entities involved in the communication are those they claim to be involved in. Certificates are issued by trusted Certificate Authorities (CAs), and any attempt to present a false certificate can be detected.

Network Segmentation

Network segmentation is the practice of dividing a computer network into smaller sub-networks or segments to enhance security. It can help prevent Man-In-The-Middle (MITM) attacks by restricting an attacker's ability to intercept and manipulate communication between devices, move laterally within the network, and gain access to sensitive information.

IPsec

Internet Protocol Security (IPsec) is a set of protocols that the Internet Engineering Task Force (IETF) developed to support the secure exchange of packets at the IP layer. It ensures interoperable cryptographically based security for IPv4 and IPv6, and it supports network-level peer authentication, data origin authentication, data integrity, data confidentiality (encryption), and replay protection. It is widely used to implement VPNs and for remote user access through dial-up connections to private networks. It supports transport and tunnel encryption modes, although sending and receiving devices must share a public key.

IPsec policies can be assigned through the Group Policy configuration of Active Directory domains, organizational units, and IPsec deployment policies at the domain, site, or organizational unit level. The security services offered by IPsec include the following:

- Rejection of replayed packets (a form of partial sequence integrity)
- Data confidentiality (encryption)
- Access control
- Connectionless integrity
- Data origin authentication
- Data integrity
- Limited traffic-flow confidentiality
- Network-level peer authentication
- Replay protection

At the IP layer, IPsec provides all the above-mentioned services, offering the protection of IP and/or upper-layer protocols such as TCP, UDP, ICMP, and Border Gateway Protocol (BGP).

Components of IPsec

IPsec driver: Software that performs protocol-level functions required to encrypt and decrypt packets.

- **Internet Key Exchange (IKE):** A protocol that produces security keys for IPsec and other protocols

- **Internet Security Association and Key Management Protocol (ISAKMP):** Software that allows two computers to communicate by encrypting the data exchanged between them
- **Oakley:** A protocol that uses the Diffie–Hellman algorithm to create a master key and a key that is specific to each session in IPsec data transfer
- **IPsec Policy Agent:** A service included in Windows OS that enforces IPsec policies for all the network communications initiated from that system

The following are the steps involved in the IPsec process.

- A consumer sends a message to a service provider
- The consumer's IPsec driver attempts to match the outgoing packet's address or the packet type against the IP filter
- The IPsec driver notifies ISAKMP to initiate security negotiations with the service provider
- The service provider's ISAKMP receives the security negotiation request
- Both principles initiate a key exchange, establishing an ISAKMP Security Association (SA) and a shared secret key
- Both principles discuss the security level for the information exchange, establishing both IPsec SAs and keys
- The consumer's IPsec driver transfers packets to the appropriate connection type for transmission to the service provider
- The provider receives the packets and transfers them to the IPsec driver
- The provider's IPsec uses the inbound SA and key to check the digital signature and begin decryption
- The provider's IPsec driver transfers decrypted packets to the OSI transport layer for further processing.

Modes of IPsec

The configuration of IPsec involves two different modes: the tunnel mode and transport mode. These modes are associated with the functions of two core protocols: the Encapsulation Security Payload (ESP) and Authentication Header (AH). The model selection depends on the requirements and implementation of IPsec.

Transport Mode

In transport mode, IPsec encrypts only the payload of the IP packet and not the IP header. IP headers remain intact, and only the data payload is encrypted or authenticated. This mode is used for end-to-end communications between two hosts.

In the transport mode (for ESP), IPsec encrypts only the payload of the IP packet, leaving the header untouched. It authenticates two connected computers and provides the option of encrypting data transfer. It is compatible with Network Address Translation (NAT); therefore, it can be used to provide VPN services for networks utilizing NAT.

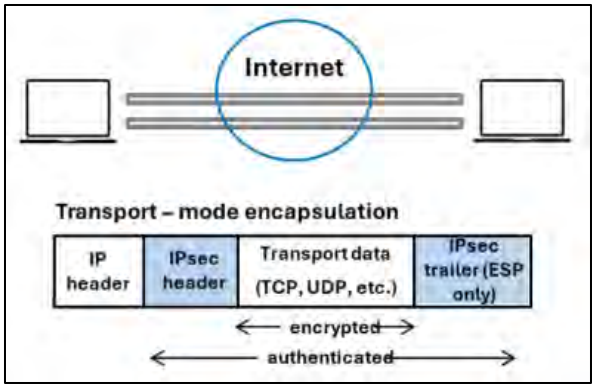

Figure 11-14: Transport Mode

Tunnel Mode

In tunnel mode, IPsec encapsulates the entire IP packet (payload and IP header) and then encrypts the entire packet. This encapsulated packet becomes the payload for a new IP packet with a new IP header.

In the tunnel mode (for AH), the IPsec encrypts both the payload and header. Hence, in the tunnel mode has higher security than the transport mode. After receiving the data, the IPsec-compliant device performs decryption. The tunnel model is used to create VPNs over the Internet for network-to-network communication (e.g., between routers and link sites), host-to-network communication (e.g., remote user access), and host-to-host communication (e.g., private chat). It is compatible with NAT and supports NAT traversal.

In the tunnel mode, ESP encrypts and optionally authenticates entire inner IP packets, whereas AH authenticates entire inner IP packets and selected fields of outer IP headers. The tunnel mode is usually useful between two gateways or between a host and gateway.

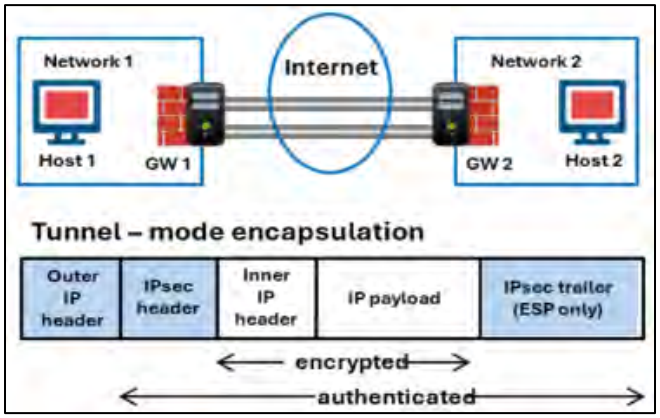

Figure 11-15: Tunnel Mode Encapsulation

IPsec Architecture

IPsec offers security services at the network layer. This allows the freedom to select the required security protocols and algorithms. If necessary, the corresponding cryptographic keys can be employed to provide the requested services. Security services offered by IPsec include access control, data origin authentication, connectionless integrity, anti-replay, and confidentiality. To

meet these objectives, IPsec uses two traffic security protocols, AH and ESP, as well as cryptographic key management protocols and procedures.

The protocol structure of the IPsec architecture is as follows.

- **Authentication Header (AH):** It offers integrity and data origin authentication, with optional anti-replay features
- **Encapsulating Security Payload (ESP):** It offers all the services offered by AH as well as confidentiality
- **IPsec Domain Of Interpretation (DOI):** It defines the payload formats, types of exchange, and naming conventions for security information such as cryptographic algorithms or security policies. IPsec DOI instantiates ISAKMP for use with IP when IP uses ISAKMP to negotiate security associations.
- **Internet Security Association and Key Management Protocol (ISAKMP):** It is a key protocol in the IPsec architecture that establishes the required security for various communications over the Internet, such as government, private, and commercial communications, by combining the security concepts of authentication, key management, and security associations.
- **Policy:** IPsec policies are useful in providing network security. They define when and how to secure data, as well as security methods to use at different levels in the network. One can configure IPsec policies to meet the security requirements of a system, domain, site, organizational unit, and so on.

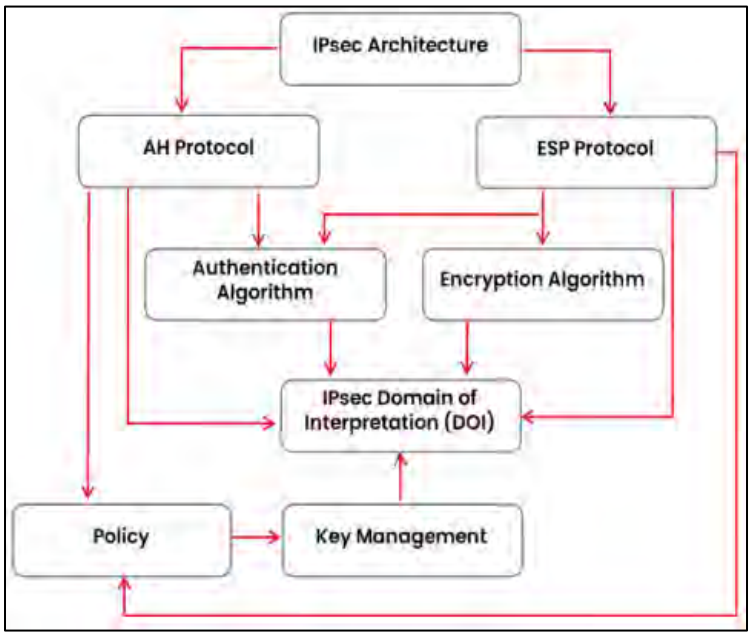

Figure 11-16: IPsec Architecture

IPsec Authentication and Confidentiality

IPsec uses two different security services for authentication and confidentiality.

- **Authentication Header (AH):** It is useful in providing connectionless integrity and data origin authentication for IP datagrams and anti-replay protection for the data payload and

some portions of the IP header of each packet. However, it does not support data confidentiality (no encryption). A receiver can select the service to protect against replays, which is an optional service on establishing a Security Association (SA).

- **Encapsulation Security Payload (ESP):** In addition to the services (data origin authentication, connectionless integrity, and anti-replay service) provided by AH, the ESP protocol offers confidentiality. Unlike AH, ESP does not provide integrity and authentication for the entire IP packet in the transport mode. ESP can be applied alone, in conjunction with AH, or in a nested manner. It protects only the IP data payload in the default setting. In the tunnel mode, it protects both the payload and IP header.

CRIME Attack

Compression Ratio Info-leak Made Easy (CRIME) is a vulnerability and a security flaw against secret web cookies across HTTPS and SPDY protocols. It is a data compression-enabled security exploit in which the attacker sends several HTTP requests to the web application with an appended cookie value to the victim. At first, the attacker listens to the conversation to get the compressed and encrypted cookie value and then analyzes the result to get the actual cookie value. The content recovery of secret authentication cookies enables an attacker to execute session hijacking on an authenticated web session, enabling the launch of future attacks.

CRIME Attack Prevention

CRIME is a client-side attack that can be defeated by preventing compression. Compression can be disabled either at the client end by the browser to prevent HTTPS requests from being compressed or by the website to prevent the use of data compression on such transactions utilizing the protocol negotiation elements of the TLS protocol.

Summary

In this chapter, we explored the concept of session hijacking, including its various types and the differences between application-level and network-level session hijacking attacks. We examined the tools commonly used to execute these attacks and discussed methods for detecting, preventing, and mitigating them. Additionally, we covered various session hijacking detection and prevention tools that enhance security against such threats. The chapter concluded with an in-depth discussion on effective countermeasures organizations and individuals can implement to protect their systems from session hijacking attempts by threat actors.

Mind Map

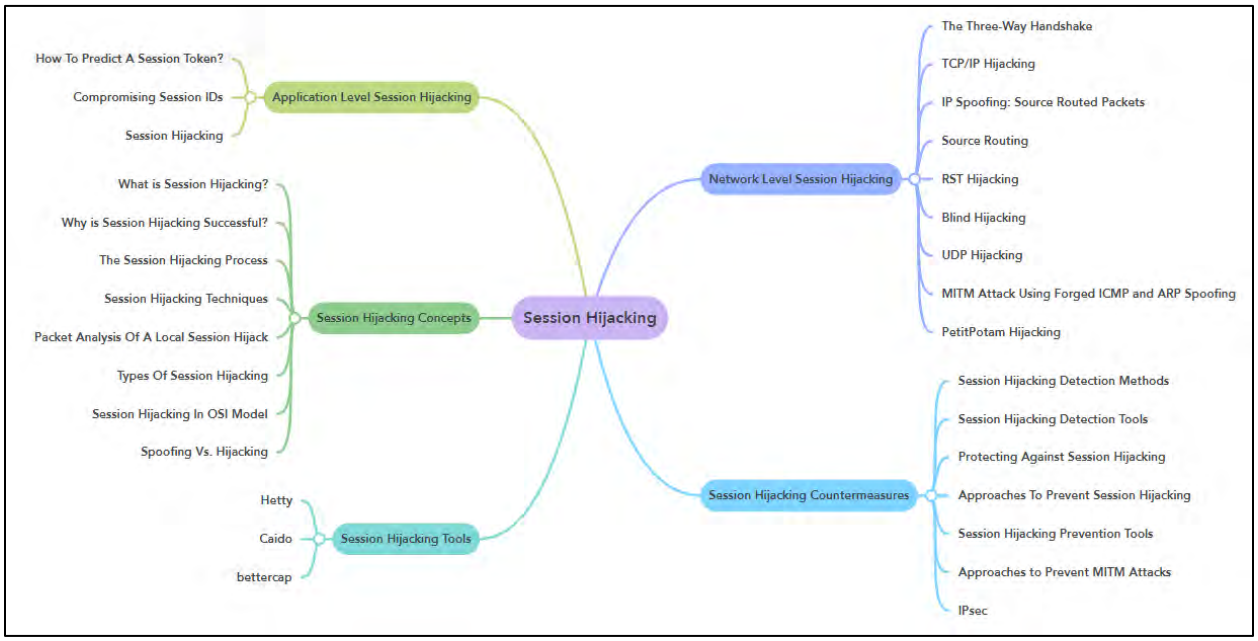

Figure 11-17: Mind Map

Practice Questions

1. What is the primary goal of session hijacking?

A. Encrypting user data

B. Taking over an authenticated user session

C. Performing DNS spoofing

D. Brute-forcing passwords

2. Which technique is commonly used in a Man-In-The-Middle (MITM) attack?

A. Brute-forcing

B. Sniffing packets

C. SQL injection

D. Social engineering

3. Which protocol is typically targeted in network-level session hijacking?

A. HTTP

B. TCP

C. FTP

D. ICMP

4. What is the first step in the session hijacking process?

A. Command injection

B. Predicting the session ID

C. Sniffing packets

D. Session desynchronization

5. What vulnerability does session fixation exploit?

A. Weak password policies

B. Mismanagement of session IDs

C. Unencrypted communication channels

D. SQL injections

6. Which of these is not a type of session hijacking?

A. Active attack

B. Passive attack

C. Spoofing attack

D. Reflected attack

7. What is a key characteristic of an active session hijacking attack?

A. Monitoring traffic without intervention

B. Actively injecting packets into the session

C. Relying on social engineering techniques

D. Using pre-recorded session data

8. What does a brute-force session hijacking attack involve?

A. Stealing user credentials

B. Trying multiple session ID combinations

C. Redirecting traffic to the attacker's server

D. Exploiting a buffer overflow

9. Which tool can be used for packet sniffing in session hijacking?

A. Nessus

B. Ettercap

C. Metasploit

D. Wireshark

10. What type of session hijacking uses a Trojan to intercept browser processes?

A. Man-in-the-Browser attack

B. Passive attack

C. TCP/IP hijacking

D. UDP hijacking

11. Which attack involves exploiting the reuse of cryptographic nonces?

A. CRIME attack

B. Forbidden attack

C. Session replay attack

D. Cross-site scripting attack

12. What protocol does DNS over HTTPS (DoH) enhance?

A. TCP

B. ICMP

C. DNS

D. FTP

13. Which of these methods helps prevent session hijacking?

A. Encrypting session cookies

B. Using small session IDs

C. Disabling HTTPS

D. Allowing session tokens in URLs

14. What is a characteristic of a CRIME attack?

A. Exploits the reuse of session cookies

B. Uses compression ratio vulnerabilities

C. Redirects traffic using forged DNS responses

D. Relies on outdated encryption algorithms

15. Which type of hijacking targets the application layer?

A. TCP/IP hijacking

B. Session fixation

C. Application-level hijacking

D. Blind hijacking

16. What technique is used in source routing for session hijacking?
A. Forging DNS responses
B. Modifying IP headers
C. Directing packets through a specific route
D. Encrypting traffic with a public key

17. What is the purpose of the SameSite cookie attribute?
A. Restricting cross-site request forgery attacks
B. Enabling session persistence
C. Allowing cookies to be shared across domains
D. Encrypting session cookies

18. What tool is commonly used to analyze hijacked sessions?
A. OWASP ZAP
B. Wireshark
C. Fiddler
D. Burp Suite

19. Which layer of the OSI model is targeted in network-level hijacking?
A. Application
B. Transport
C. Data Link
D. Network

20. What is a characteristic of a blind hijacking attack?
A. The attacker cannot capture return traffic
B. The attacker monitors the entire session
C. The attack relies on pre-captured session data
D. The attacker exploits weak password policies

21. Which session hijacking technique does not involve direct packet manipulation?
A. Passive attack
B. Active attack

C. TCP/IP hijacking

D. Man-in-the-Browser attack

22. What is a major countermeasure against ARP spoofing?

A. Using HTTPS

B. Configuring static ARP tables

C. Encrypting session tokens

D. Using small session IDs

23. What distinguishes spoofing from hijacking?

A. Spoofing targets active sessions

B. Hijacking creates a new session

C. Spoofing impersonates another user

D. Hijacking relies on stolen credentials

24. Which protocol can be used to establish a secure session?

A. HTTP

B. FTP

C. SSH

D. ICMP

25. What does IPsec's tunnel mode protect?

A. Only the payload

B. Only the header

C. Both payload and header

D. Neither payload nor header

Answers

1. **Answer: B**

Explanation: The main goal of session hijacking is to gain unauthorized access by taking control of an authenticated session. By stealing or predicting a valid session ID, attackers can impersonate the legitimate user. This allows them to perform malicious activities, such as stealing sensitive data, executing unauthorized transactions, or accessing restricted resources, without needing the user's login credentials.

2. **Answer: B**

Explanation: Packet sniffing is the process of capturing and analyzing network traffic to identify sensitive information such as session IDs. In MITM attacks, sniffing enables attackers to eavesdrop on communication between the client and server, gaining access to confidential data like cookies, credentials, or tokens used for session authentication.

3. **Answer: B**

Explanation: TCP is a connection-oriented protocol widely used for reliable data communication over networks. Network-level session hijacking often targets TCP sessions because it involves sequence numbers that can be predicted or manipulated. By interfering with the three-way handshake or ongoing communication, attackers can take control of a session.

4. **Answer: C**

Explanation: Sniffing packets is the initial step in most session hijacking attacks. Attackers monitor network traffic to identify vulnerabilities, such as unencrypted session tokens or predictable patterns, which can then be exploited to hijack sessions. Tools like Wireshark are commonly used for packet sniffing.

5. **Answer: B**

Explanation: Session fixation occurs when an attacker provides a predetermined session ID to the victim, forcing their browser to use it during authentication. Once the victim logs in, the attacker can hijack the session using the same session ID. This attack exploits poor session management practices, such as not regenerating session IDs after login.

6. **Answer: D**

Explanation: Reflected attacks are a form of Cross-Site Scripting (XSS) where malicious scripts are injected and reflected off a web application. Unlike session hijacking, reflected attacks do not involve taking control of an existing session but instead aim to execute unauthorized scripts in the victim's browser.

7. **Answer: B**

Explanation: Active session hijacking involves the attacker actively interfering in a session by injecting packets or commands. This allows the attacker to manipulate or disrupt communication, impersonate the user, or redirect traffic. This type of attack often results in the disconnection of the legitimate user.

8. **Answer: B**

Explanation: Brute-forcing session IDs involves systematically trying all possible combinations until the correct session ID is found. This technique is effective against systems with weak or predictable session ID generation algorithms, especially if there is no account lockout mechanism in place.

9. **Answer: D**

Explanation: Wireshark is a network protocol analyzer that captures and examines data packets in real time. It helps attackers and security professionals analyze network traffic, identify vulnerabilities, and locate session tokens, making it an essential tool in both session hijacking and its prevention.

10. Answer: A

Explanation: Man-in-the-Browser (MitB) attacks involve the use of Trojans or malicious browser extensions that intercept browser processes. These attacks modify user inputs or session data on the client side without detection, enabling attackers to steal session IDs or redirect transactions.

11. Answer: B

Explanation: Forbidden attacks exploit the improper reuse of cryptographic nonces during secure communication. By hijacking a session through nonce reuse, attackers can bypass encryption mechanisms, inject malicious content, or access sensitive data exchanged in the session.

12. Answer: C

Explanation: DNS over HTTPS (DoH) encrypts DNS queries to prevent attackers from intercepting or altering them during the lookup process. This enhancement adds a layer of security to the otherwise plaintext DNS protocol, mitigating risks like DNS spoofing or MITM attacks.

13. Answer: A

Explanation: Encrypting session cookies protects them from being read or modified during transmission. Even if an attacker intercepts the cookies, encryption prevents them from using the stolen data to hijack the session. Implementing secure transmission protocols like HTTPS further enhances this protection.

14. Answer: B

Explanation: CRIME attacks exploit vulnerabilities in compression algorithms (e.g., DEFLATE) used in SSL/TLS protocols. By analyzing compressed and uncompressed data, attackers can deduce sensitive information like session cookies, enabling them to hijack secure sessions.

15. Answer: C

Explanation: Application-level hijacking focuses on compromising communication at the application layer of the OSI model. This type of hijacking often targets session tokens or cookies, allowing attackers to impersonate authenticated users and access protected web resources.

16. Answer: C

Explanation: Source routing involves directing packets along a specified path instead of relying on standard routing methods. Attackers use this technique to intercept or manipulate network traffic by ensuring packets pass through a compromised or controlled route.

17. Answer: A

Explanation: The SameSite cookie attribute prevents cookies from being sent with cross-site requests. This reduces the risk of Cross-Site Request Forgery (CSRF) attacks, where attackers exploit authenticated sessions to perform unauthorized actions on behalf of users.

18. Answer: B

Explanation: Wireshark is a critical tool for detecting and analyzing network traffic. It allows security professionals to monitor session activities, identify unusual patterns, and trace session hijacking attempts by examining captured packets in detail.

19. Answer: B

Explanation: Network-level hijacking targets the transport layer, which manages data transfer and reliability. By exploiting protocols like TCP or UDP, attackers can intercept, modify, or inject data into ongoing sessions.

20. Answer: A

Explanation: Blind hijacking occurs when attackers cannot capture return traffic but still inject malicious packets into the session. This relies on guessing sequence numbers and sending spoofed packets to disrupt or control communication.

21. Answer: A

Explanation: Passive attacks involve monitoring communication without actively interfering. These attacks are often used for reconnaissance, allowing attackers to gather sensitive information like session tokens for use in future active attacks.

22. Answer: B

Explanation: Configuring static ARP tables ensures that each device on the network communicates with trusted MAC addresses. This prevents ARP spoofing attacks, where attackers redirect traffic by sending forged ARP replies.

23. Answer: C

Explanation: Spoofing impersonates a user or device by faking identity credentials (e.g., IP or MAC address). Unlike hijacking, which involves taking over an active session, spoofing establishes a new connection under false pretenses.

24. Answer: C

Explanation: Secure Shell (SSH) establishes encrypted sessions, protecting data integrity and confidentiality during communication. It is widely used for secure remote logins, file transfers, and tunneling protocols.

25. Answer: C

Explanation: IPsec's tunnel mode encrypts both the IP header and payload, ensuring end-to-end security. This mode is ideal for VPNs, as it secures the entire packet during transmission, protecting data from interception or tampering.