Module 11

Session Hijacking

Learning Objectives

01 Summarize Session Hijacking Concepts

02 Explain Application-Level Session Hijacking

03 Explain Network-Level Session Hijacking

04 Explain Session Hijacking Countermeasures

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

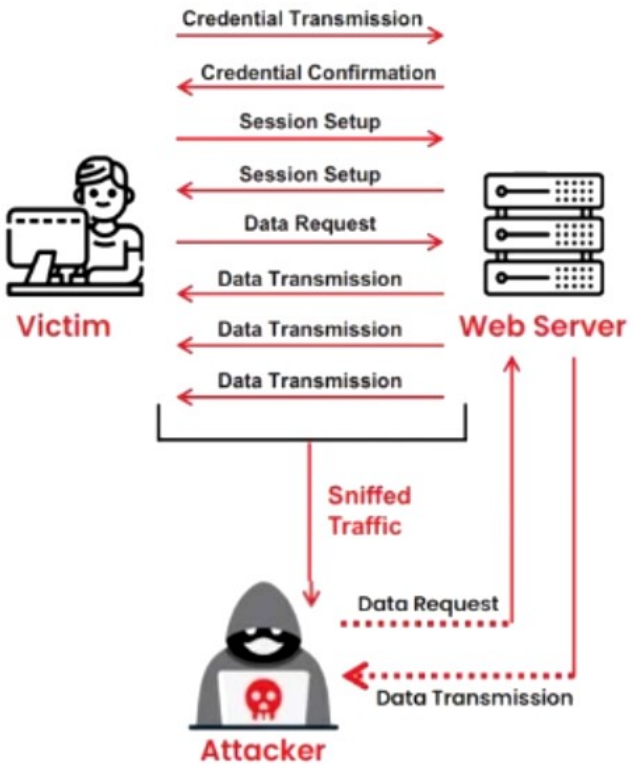
Objective 01

Summarize Session Hijacking Concepts

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

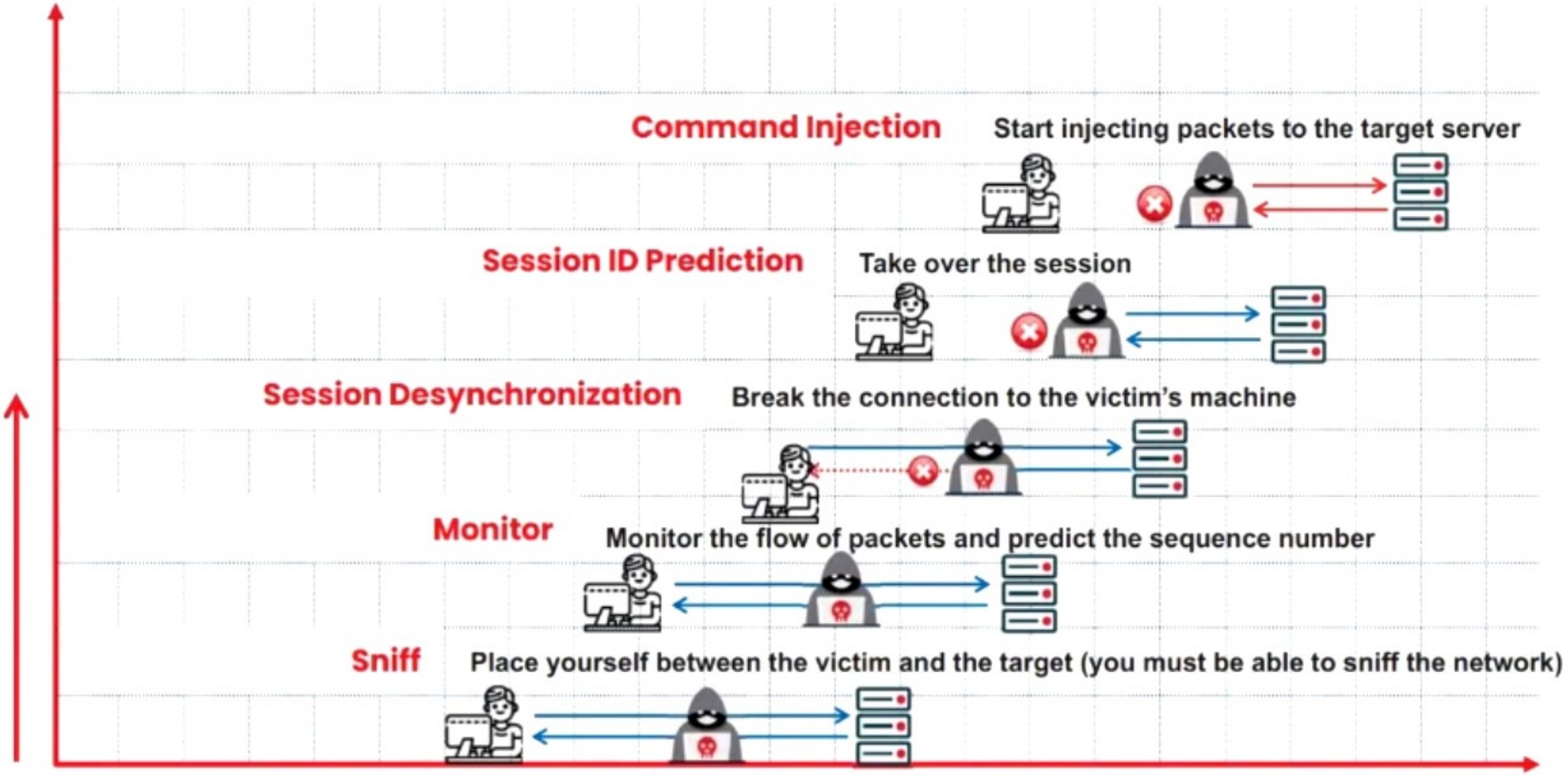
What is Session Hijacking?

- Session hijacking refers to an attack in which an attacker seizes control of a **valid TCP communication session** between two computers
- As most **authentications only occur at the start of a TCP session**, this allows the attacker to gain access to a machine
- Attackers can sniff all the traffic from the established TCP sessions and perform **identity theft**, **information theft**, **fraud**, etc.
- The attacker steals a valid session ID and uses it to **authenticate himself with the server**



Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Process



Command Injection Start injecting packets to the target server

Session ID Prediction Take over the session

Session Desynchronization Break the connection to the victim's machine

Monitor Monitor the flow of packets and predict the sequence number

Sniff Place yourself between the victim and the target (you must be able to sniff the network)

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking in OSI Model

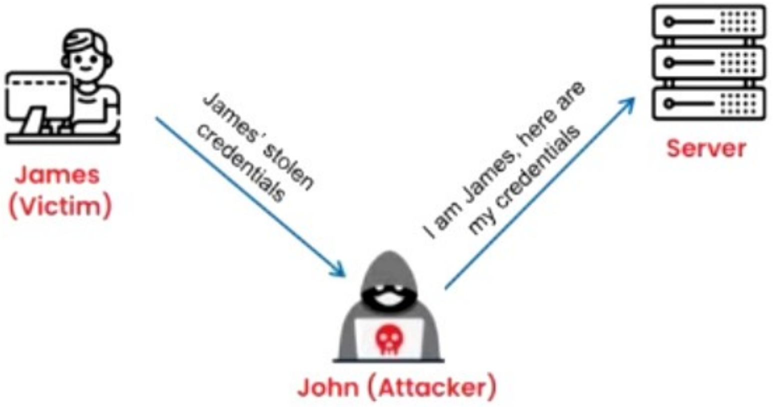
Network-Level Hijacking	Network-level hijacking can be defined as the **interception of packets** during the transmission between a client and the server in a TCP or UDP session
Application-Level Hijacking	Application-level hijacking refers to **gaining control over the HTTP's user session** by obtaining the session IDs

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Spoofing vs. Hijacking

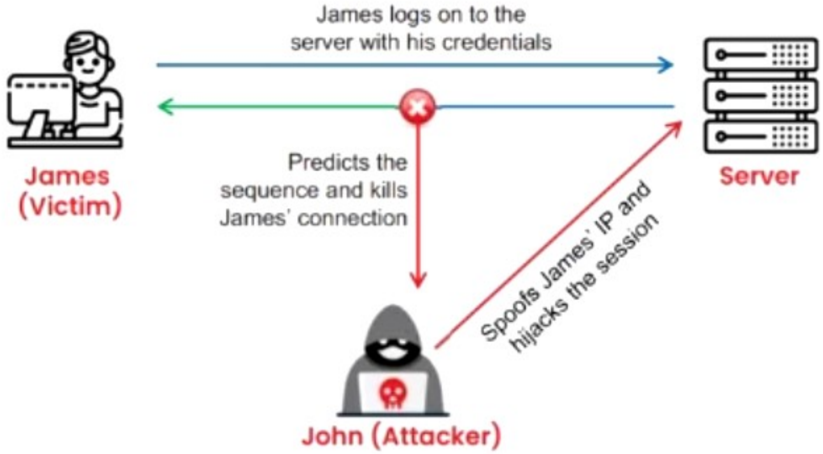
Spoofing Attack

- An attacker **pretends to be another user** or machine (victim) to gain access
- The attacker does not seize control of an existing active session; instead, he or she initiates a new session using the victim's **stolen credentials**


James (Victim)
James' stolen credentials
I am James, here are my credentials
Server
John (Attacker)

Hijacking

- Session hijacking is the process of seizing control of an **existing active session**
- The attacker relies on the **legitimate user** to create a connection and authenticate


James logs on to the server with his credentials
James (Victim)
Predicts the sequence and kills James' connection
Spoofs James' IP and hijacks the session
Server
John (Attacker)

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Objective 02

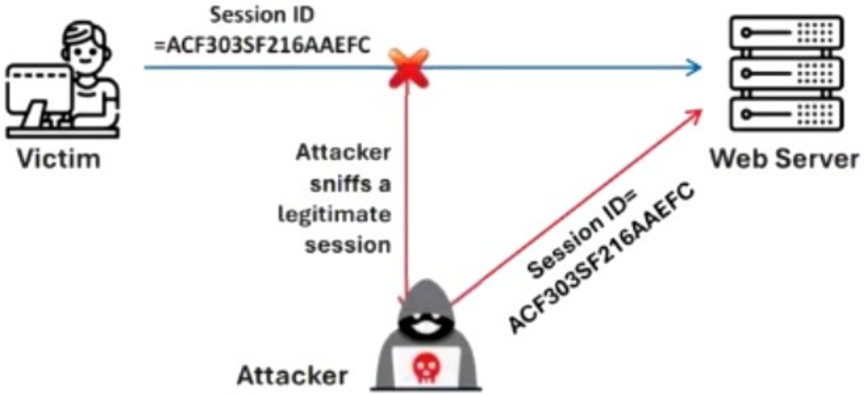
Explain Application-Level Session Hijacking

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Compromising Session IDs using Sniffing and by Predicting Session Token

Compromising Session IDs using Sniffing

- An attacker uses a sniffer to **capture a valid session token** or **session ID**
- The attacker then uses the valid token session to **gain unauthorized access** to the web server



Compromising Session IDs by Predicting Session Token

- Attackers can **predict session IDs** generated by weak algorithms and **impersonate a website user**
- Attackers analyze variable sections of session IDs to **determine a pattern**
- The analysis is performed **manually** or **using various cryptanalytic tools**
- Attackers **collect a high number of simultaneous session IDs** to gather samples in the same time window and keep the variable constant

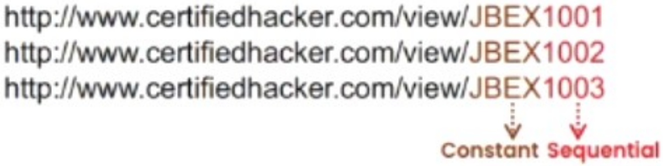
Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

How to Predict a Session Token

Analyzing Token Patterns

Sequential Tokens

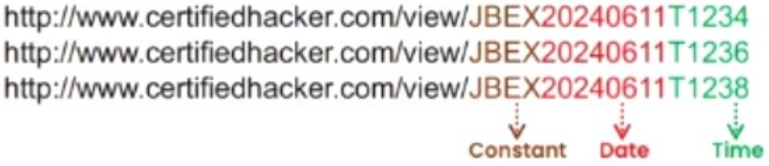
Tokens can be predicted by attackers if they follow an identifiable pattern



http://www.certifiedhacker.com/view/JBEX1001
http://www.certifiedhacker.com/view/JBEX1002
http://www.certifiedhacker.com/view/JBEX1003

Timestamp-based Tokens

Tokens are easier to predict if they include a timestamp

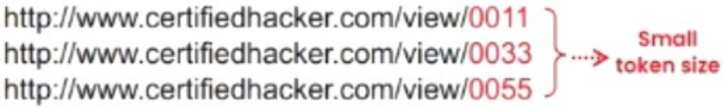


http://www.certifiedhacker.com/view/JBEX20240611T1234
http://www.certifiedhacker.com/view/JBEX20240611T1236
http://www.certifiedhacker.com/view/JBEX20240611T1238

Brute Force Attacks

Small Token Space

A small token space allows attackers to use brute-force attacks to guess all possible tokens



http://www.certifiedhacker.com/view/0011
http://www.certifiedhacker.com/view/0033
http://www.certifiedhacker.com/view/0055

Lack of Rate Limiting

Without **rate limiting**, attackers can make numerous token guesses without being blocked

Weak Random Number Generators

Predictable PRNG

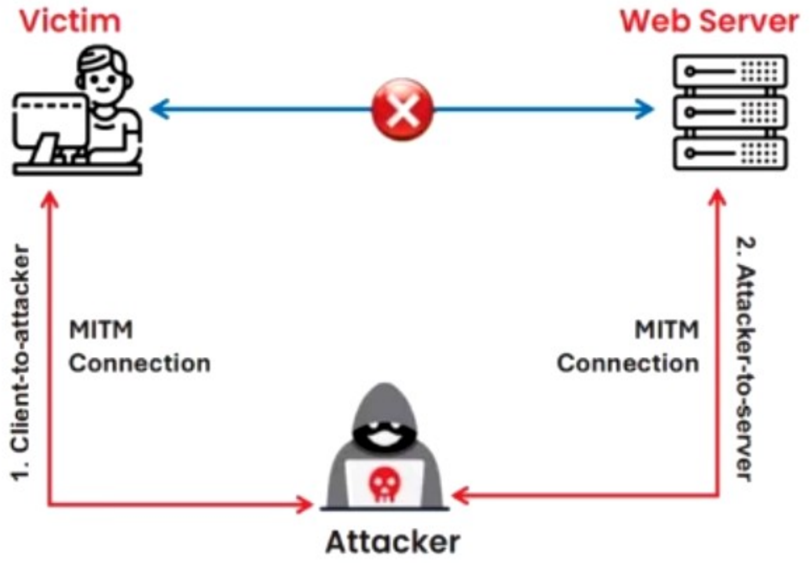
Predictable PRNGs can produce token sequences that attackers can guess if they know the seed or algorithm

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Compromising Session IDs Using Man-in-the-Middle/ Manipulator-in-the-Middle Attack

The man-in-the-middle/manipulator-in-the-middle attack is used to **intrude into an existing connection** between systems and intercept the messages being exchanged

- Attackers use different techniques and **split the TCP connection** into two connections:
  - Client-to-attacker connection
  - Attacker-to-server connection
- After the interception of the TCP connection, an attacker can read, modify, and insert fraudulent data into the **intercepted communication**
- In the case of an **http transaction**, the TCP connection between the client and the server becomes the target



Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Compromising Session IDs Using Man-in-the-Browser /Manipulator-in-the-Browser Attack

The man-in-the-browser/manipulator-in-the- browser attack **uses a Trojan horse** to intercept the calls between the browser and its security mechanisms or libraries

01 The Trojan first infects the **OS** or **application**

02 The Trojan installs malicious code (extension files) and saves it in the **browser configuration**

03 When the user restarts the browser, it loads the **malicious extension** files

04 The extension files register a **handler** for every visit to a webpage

05 When a page is loaded, the extension matches the **URL** with a list of **targeted sites**

06 The user **logs in securely** to the website

07 The extension registers a **button event handler** for specific page loads

08 When the user clicks the button; the extension uses **DOM** to extract and modify **form data**

09 The browser sends the **form** and **modified values** to the server

10 The server **receives modified values** but cannot distinguish from the original

11 After the server performs the transaction, a **receipt is generated**

12 Now, the browser receives the receipt for the **modified transaction**

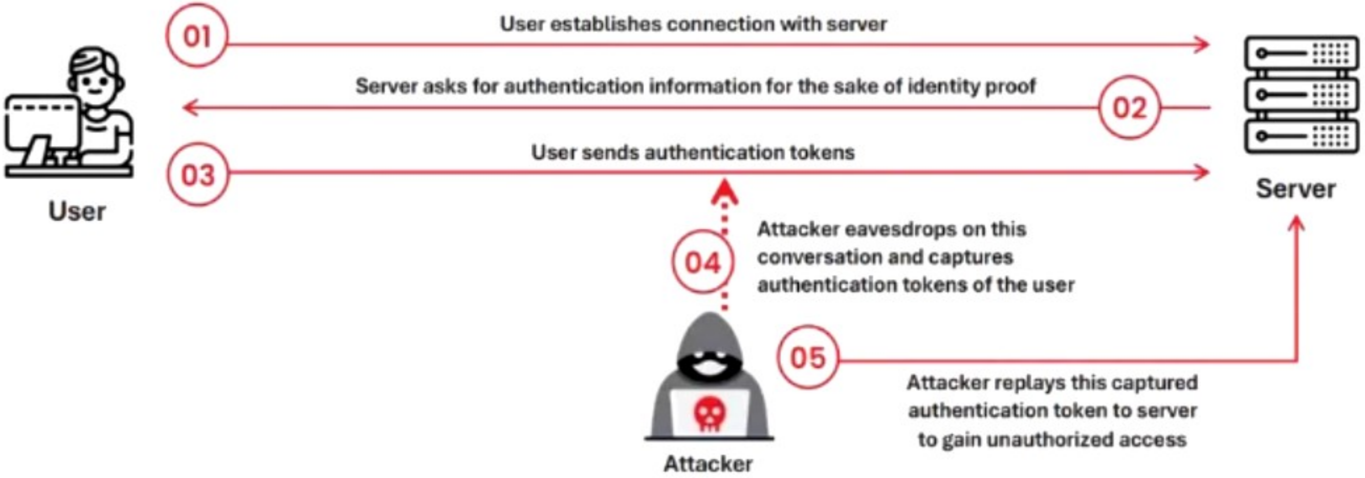
13 The browser displays the receipt with the **original details**

14 The user believes the original transaction was processed **without interception**

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Compromising Session IDs Using Session Replay Attacks

- In a session replay attack, the attacker listens to the conversation between the **user and the server** and captures the **authentication token** of the user
- Once the authentication token is captured, the attacker **replays the request to the server** with the captured **authentication token** and gains **unauthorized access** to the server

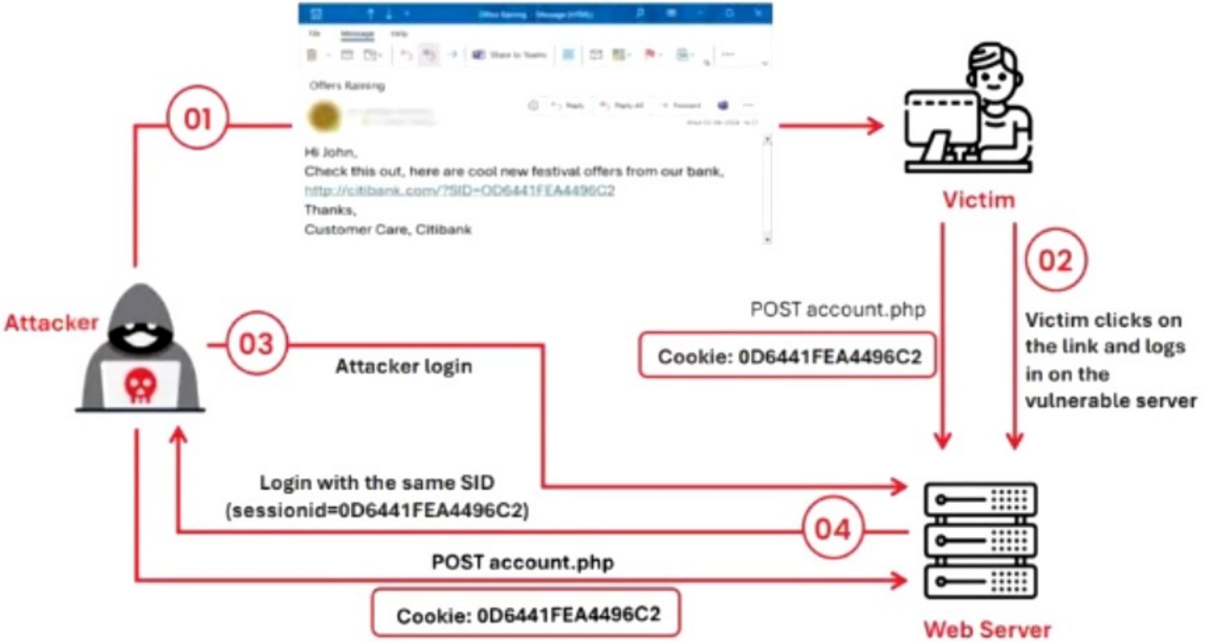

01
User establishes connection with server
02
Server asks for authentication information for the sake of identity proof
03
User sends authentication tokens
04
Attacker eavesdrops on this conversation and captures authentication tokens of the user
05
Attacker replays this captured authentication token to server to gain unauthorized access
User
Server
Attacker

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Compromising Session IDs Using Session Fixation

- Session fixation is an attack that allows an attacker to hijack a **valid user session**
- An attacker attempts to lure a user to authenticate himself or herself with a known session ID and then hijacks the **user-validated session** with the knowledge of the used session ID
- The attacker has to provide a **legitimate web application session ID** and attempt to lure the victim's browser to use it
- Some techniques for executing session fixation attacks are as follows:
  - Session token in the **URL argument**
  - Session token in a **hidden form field**
  - Session ID in a **cookie**

- The attacker exploits the **vulnerability of a server** that allows a user to use a fixed SID
- The attacker provides a **valid SID** to a victim and lures him or her to **authenticate** using that SID



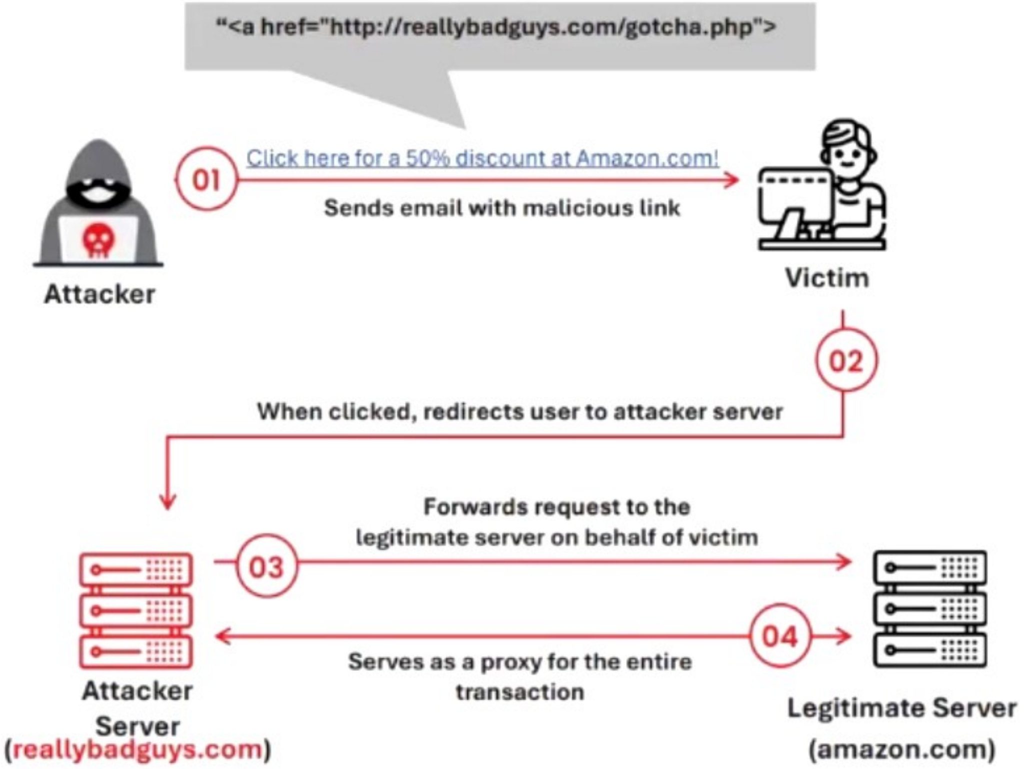
Offers Raining

Hi John,
Check this out, here are cool new festival offers from our bank,
http://citibank.com/?SID=0D6441FEA4496C2
Thanks,
Customer Care, Citibank

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Using Proxy Servers

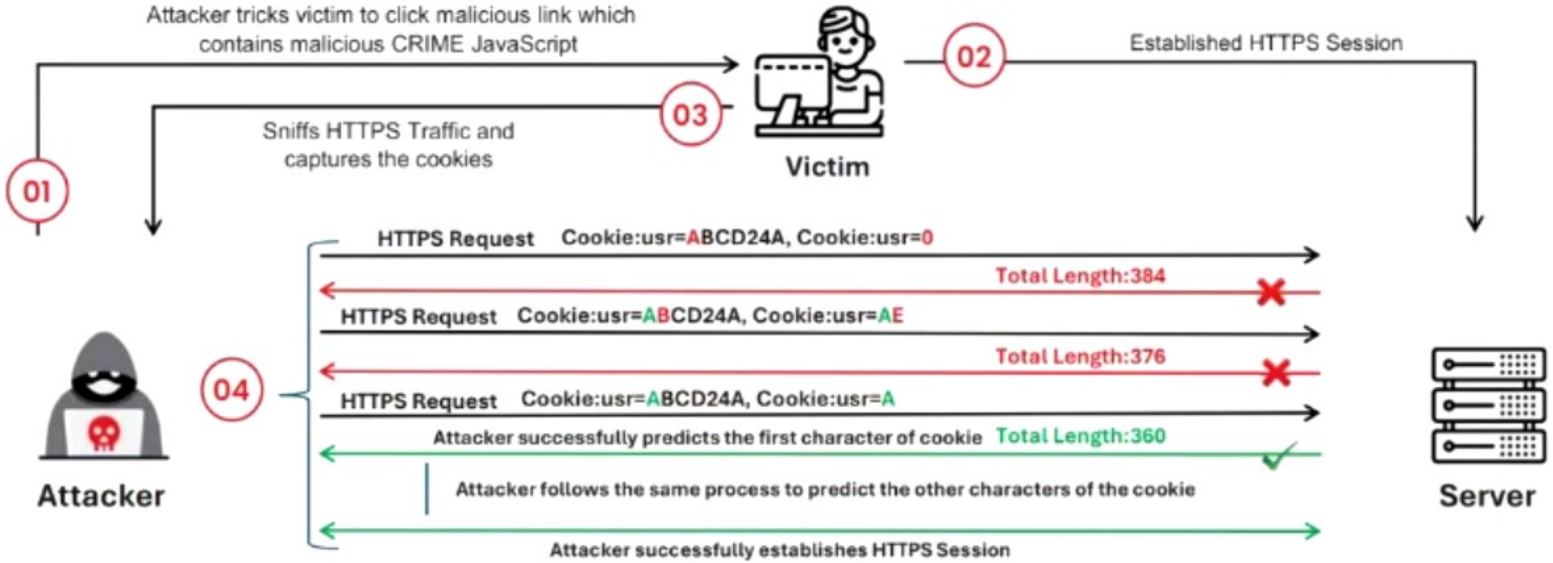
- An attacker lures the victim to **click on a bogus link**, which looks legitimate but redirects the user to the attacker server
- The attacker forwards the request to the legitimate server on behalf of the victim and **serves as a proxy** for the entire transaction
- The attacker then **captures the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


"<a href="http://reallybadguys.com/gotcha.php">
01
Click here for a 50% discount at Amazon.com!
Sends email with malicious link
Attacker
Victim
02
When clicked, redirects user to attacker server
03
Forwards request to the legitimate server on behalf of victim
04
Serves as a proxy for the entire transaction
Attacker Server (reallybadguys.com)
Legitimate Server (amazon.com)

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Using CRIME Attack

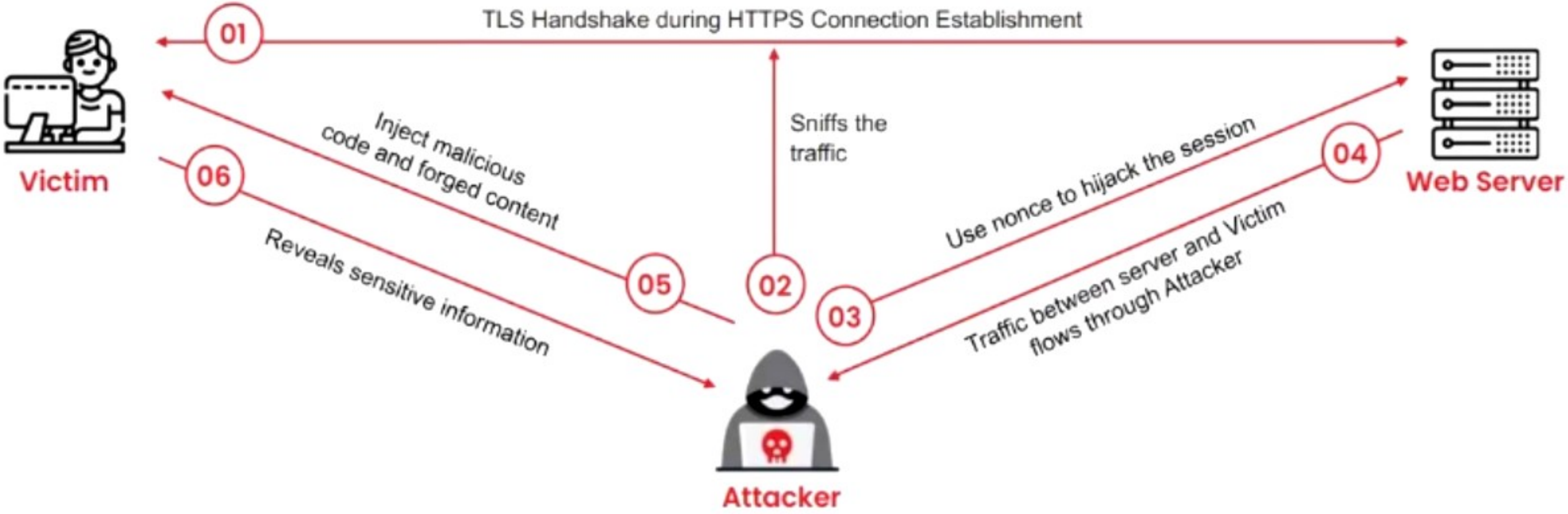
- Compression Ratio Info-Leak Made Easy (CRIME) is a client-side attack that exploits the vulnerabilities present in the **data compression** feature of protocols, such as SSL/TLS, SPDY, and HTTPS
- Attackers hijack the session by decrypting secret **session cookies**
- The authentication information obtained from the session cookies is used to establish a **new session** with the web application


Attacker tricks victim to click malicious link which contains malicious CRIME JavaScript
01
02
Established HTTPS Session
03
Sniffs HTTPS Traffic and captures the cookies
Victim
04
HTTPS Request Cookie:usr=ABCD24A, Cookie:usr=0
Total Length:384
HTTPS Request Cookie:usr=ABCD24A, Cookie:usr=AE
Total Length:376
HTTPS Request Cookie:usr=ABCD24A, Cookie:usr=A
Attacker successfully predicts the first character of cookie
Total Length:360
Attacker follows the same process to predict the other characters of the cookie
Attacker successfully establishes HTTPS Session
Attacker
Server

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Using Forbidden Attack

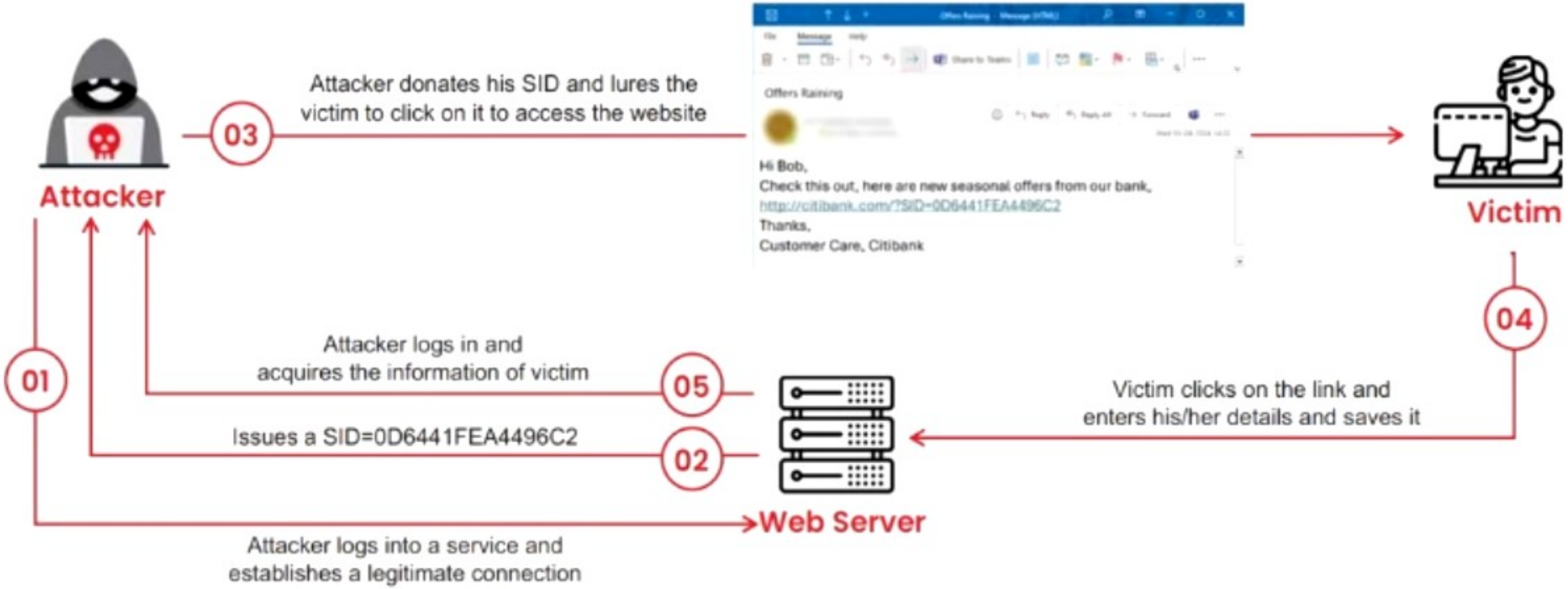
- A forbidden attack is a type of man-in-the-middle attack used to **hijack HTTPS sessions**
- It exploits the reuse of **cryptographic nonce** during the TLS handshake
- After hijacking the HTTPS session, the attackers **inject malicious code** and **forged content** that prompts the victim to disclose sensitive information, such as bank account numbers, passwords, and social security numbers


01
TLS Handshake during HTTPS Connection Establishment
Victim
Web Server
02
Sniffs the traffic
03
Use nonce to hijack the session
04
Traffic between server and Victim flows through Attacker
05
Inject malicious code and forged content
06
Reveals sensitive information
Attacker

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Using Session Donation Attack

- In a session donation attack, an attacker **donates his/her own session identifier (SID)** to the target user
- The attacker first **obtains a valid SID** by logging into a service and later feeds the same SID to the target user
- This SID **links a target user back to the attacker's account page** without any information to the victim


Attacker
Victim
Web Server
01
Attacker logs into a service and establishes a legitimate connection
02
Issues a SID=0D6441FEA4496C2
03
Attacker donates his SID and lures the victim to click on it to access the website
04
Victim clicks on the link and enters his/her details and saves it
05
Attacker logs in and acquires the information of victim

Offers Raining

[redacted]

Hi Bob,
Check this out, here are new seasonal offers from our bank,
http://citibank.com/?SID=0D6441FEA4496C2
Thanks,
Customer Care, Citibank

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Objective 03

Explain Network-Level Session Hijacking

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Network-Level Session Hijacking

- The network-level hijacking relies on hijacking **transport** and **Internet protocols** used by web applications in the application layer

- By attacking the network-level sessions, the attacker gathers some **critical information**, which are used to **attack the application-level sessions**

Network-level hijacking includes:

01 **Blind hijacking**

02 **UDP hijacking**

03 **TCP/IP hijacking**

04 **RST hijacking**

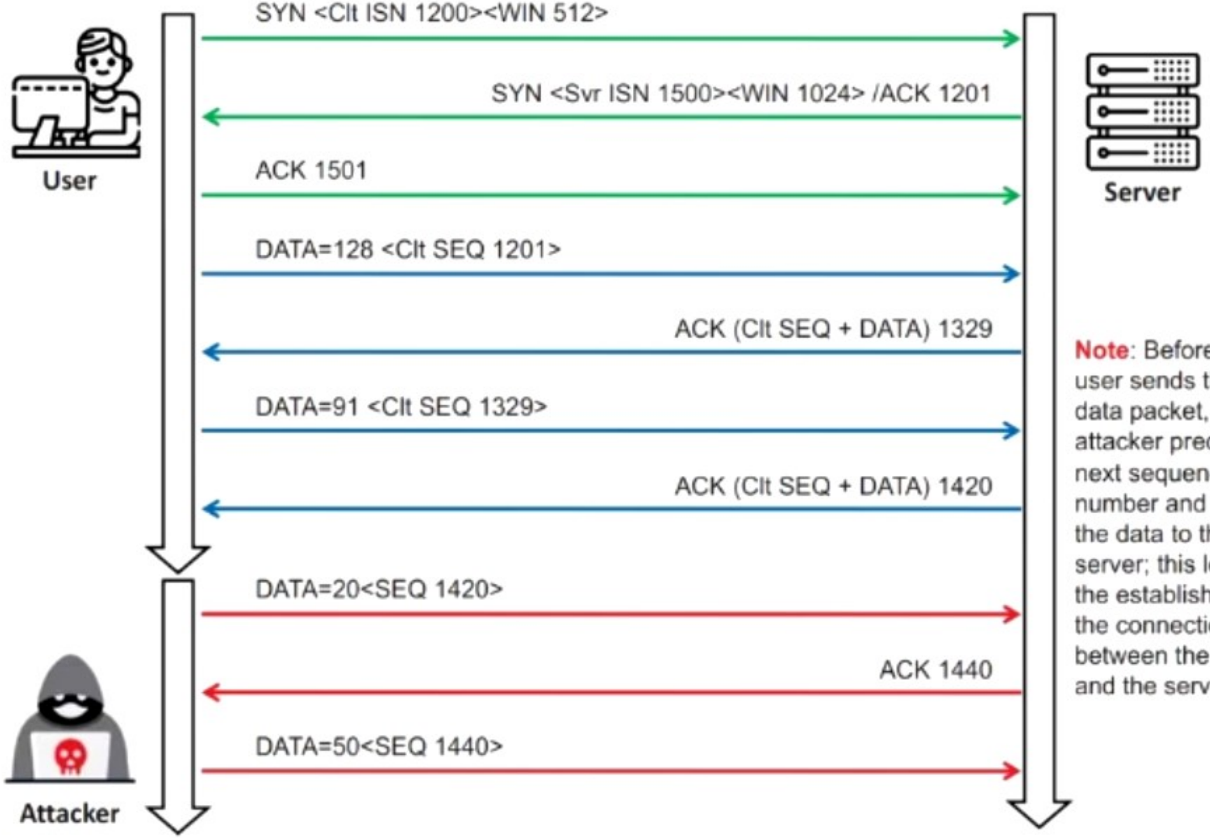
05 **Man-in-the-middle: Packet sniffer**

06 **IP spoofing: Source routed packets**

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

TCP/IP Hijacking

- TCP/IP hijacking involves using **spoofed packets** to seize control of a connection between a victim and target machine
- A victim's connection hangs, and an attacker is then able to **communicate with the host's machine** as if the attacker is the victim
- To launch a TCP/IP hijacking attack, the **attacker must be on the same network as the victim**
- The target server and the victim machines can be located anywhere



SYN <Clt ISN 1200><WIN 512>

SYN <Svr ISN 1500><WIN 1024> /ACK 1201

ACK 1501

DATA=128 <Clt SEQ 1201>

ACK (Clt SEQ + DATA) 1329

DATA=91 <Clt SEQ 1329>

ACK (Clt SEQ + DATA) 1420

DATA=20<SEQ 1420>

ACK 1440

DATA=50<SEQ 1440>

Note: Before the user sends the next data packet, the attacker predicts the next sequence number and sends the data to the server; this leads to the establishment of the connection between the attacker and the server

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

IP Spoofing: Source Routed Packets

01 The packet source routing technique is used for **gaining unauthorized access** to a computer with the help of a trusted host's IP address

02 An attackers spoofs the host's IP address so that the server **managing a session** with the host accepts the packets from the attacker

03 When the session is established, the attacker **injects forged packets** before the host responds to the server

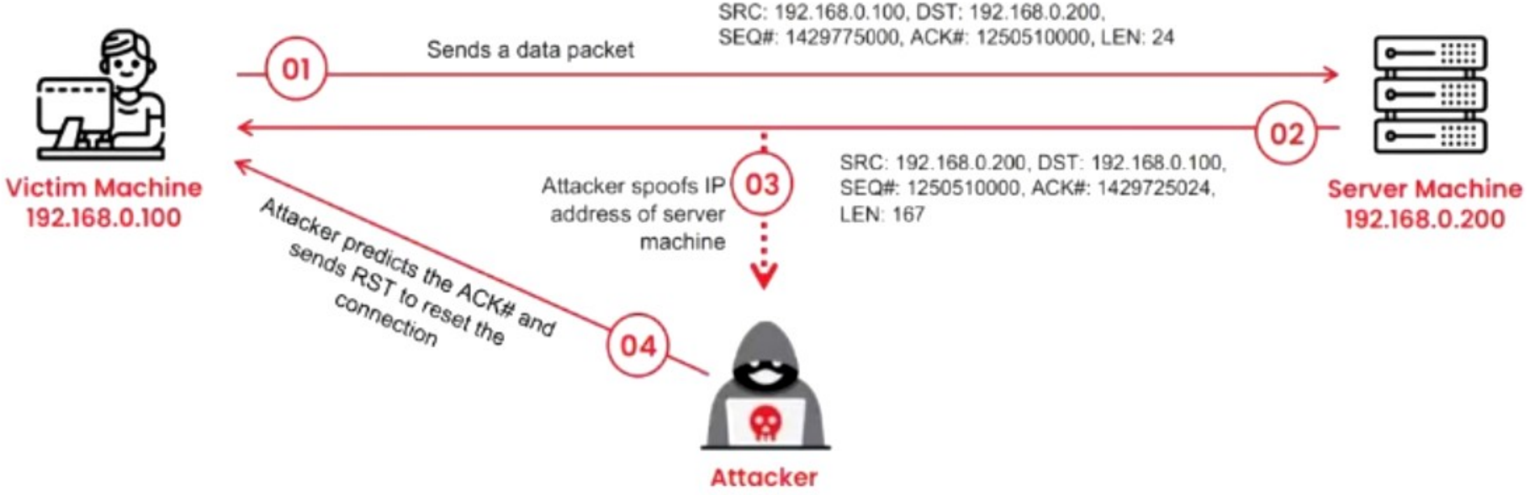
04 The original packet from the host is lost as the server receives the packet with a **sequence number** already used by the attacker

05 The packets from the attacker are source-routed through the host with the **destination IP** specified by the attacker

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

RST Hijacking

- RST hijacking involves injecting an **authentic-looking reset (RST) packet** using a spoofed source address and predicting the acknowledgment number
- A hacker can reset a victim's connection if it uses an **accurate acknowledgment number**
- The victim would believe that the source sent the **reset packet**, and **reset the connection**
- RST Hijacking can be performed using a **packet crafting tool**, such as Colasoft Packet Builder, and TCP/IP analysis tools, such as tcpdump



Victim Machine
192.168.0.100

Server Machine
192.168.0.200

Attacker

01 Sends a data packet

SRC: 192.168.0.100, DST: 192.168.0.200,
SEQ#: 1429775000, ACK#: 1250510000, LEN: 24

02

SRC: 192.168.0.200, DST: 192.168.0.100,
SEQ#: 1250510000, ACK#: 1429725024,
LEN: 167

03 Attacker spoofs IP address of server machine

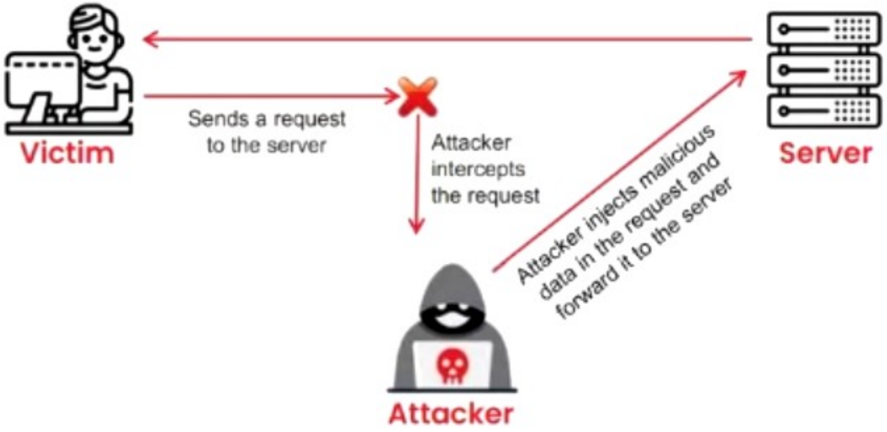
04 Attacker predicts the ACK# and sends RST to reset the connection

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Blind and UDP Hijacking

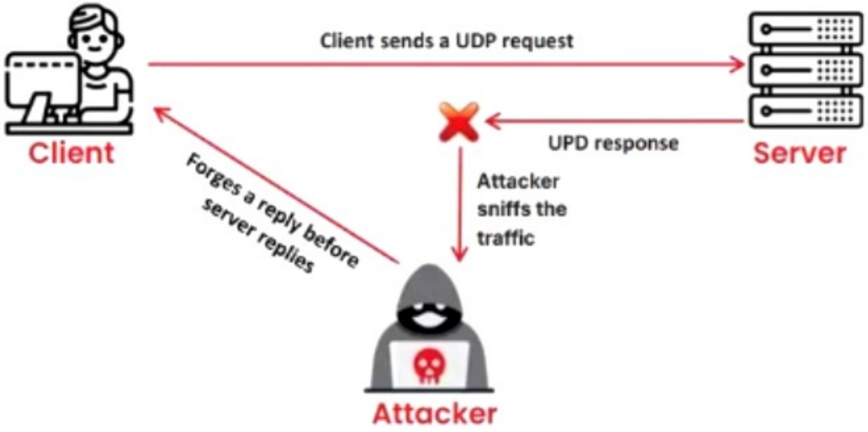
Blind Hijacking

- An attacker can inject **malicious data or commands** into the intercepted communications in the TCP session even if the source-routing is disabled
- The attacker can send the data or commands but has no **access to see the response**


Victim
Sends a request to the server
Attacker intercepts the request
Attacker injects malicious data in the request and forward it to the server
Server
Attacker

UDP Hijacking

- A network-level session hijacking where the attacker sends **forged server reply** to a victim's UDP request before the intended server replies to it
- The attacker uses a **man-in-the-middle** attack to intercept the server's response to the client and sends a forged reply


Client sends a UDP request
Client
Server
UPD response
Attacker sniffs the traffic
Forges a reply before server replies
Attacker

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

MiTM Attack Using Forged ICMP and ARP Spoofing

- In this attack, the packet sniffer is **used as an interface** between the client and server
- An attacker changes the **default gateway** of the client's machine and attempts to reroute packets
- The packets between the client and server are routed through the **hijacker's host** using two techniques, as shown below:

Forged Internet Control Message Protocol (ICMP)

It is an extension of IP to **send error messages** where the attacker can send messages to **fool the client and server**

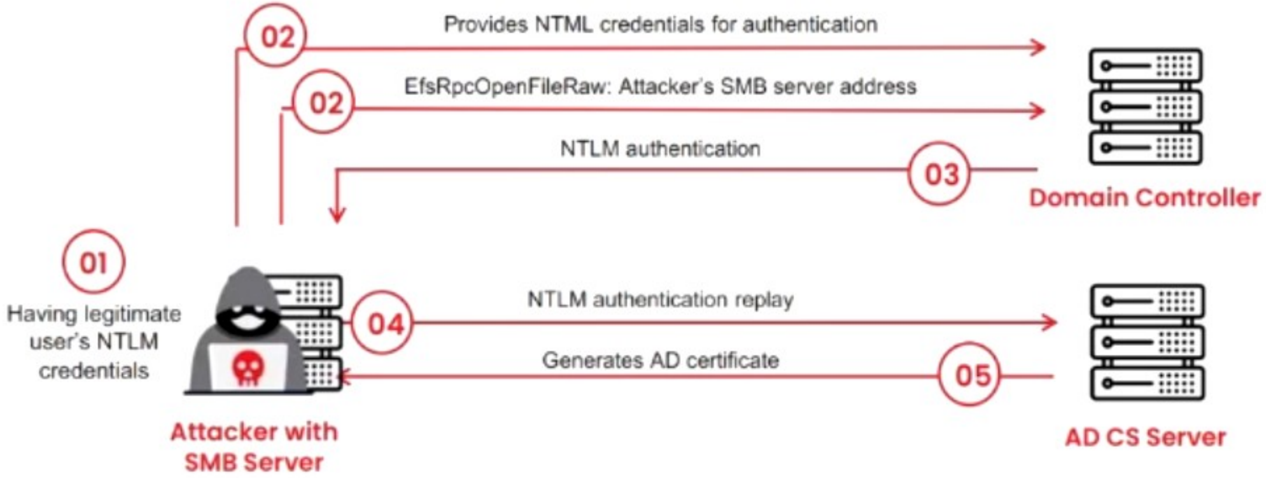
Address Resolution Protocol (ARP) Spoofing

ARP is used to map the **network layer addresses** (IP address) to **link layer addresses** (MAC address)

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

PetitPotam Hijacking

- In a PetitPotam attack, **a domain controller (DC)** is forced by an attacker to initiate authentication to the attacker's server
- The attacker uses **Microsoft's Encrypting File System Remote Protocol (MS-EFSRPC) API** for authentication session hijacking
- The attacker relays the **NTLM authentication** shared by the domain controller to the **Active Directory Certificate Services (AD CS)** server and generates a certificate to acquire **admin-level privileges**

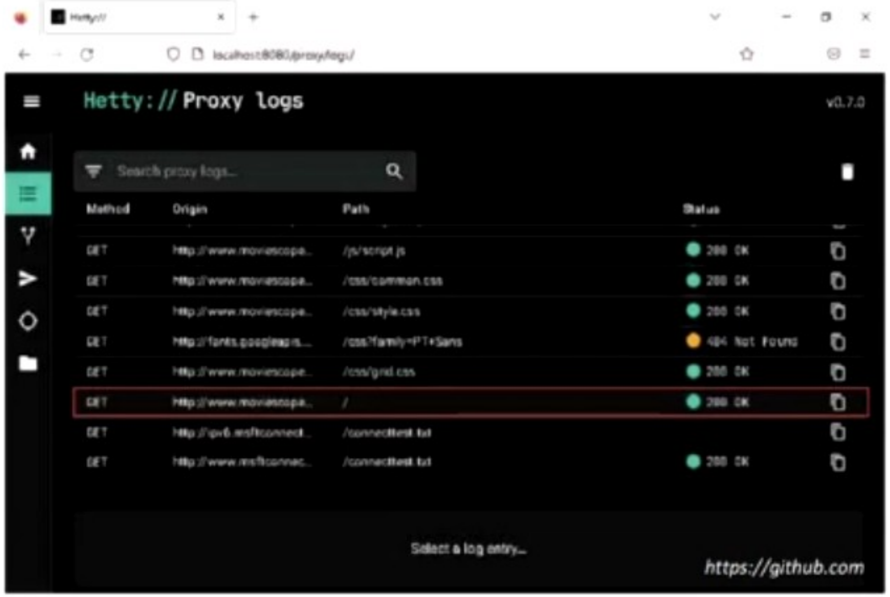

02
Provides NTML credentials for authentication
02
EfsRpcOpenFileRaw: Attacker's SMB server address
NTLM authentication
03
Domain Controller
01
Having legitimate user's NTLM credentials
04
NTLM authentication replay
Generates AD certificate
05
Attacker with SMB Server
AD CS Server

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Session Hijacking Tools

Hetty

Hetty is an HTTP toolkit that allows attackers to perform **machine-in-the-middle (MITM) HTTP proxy** attack using logs and advanced search

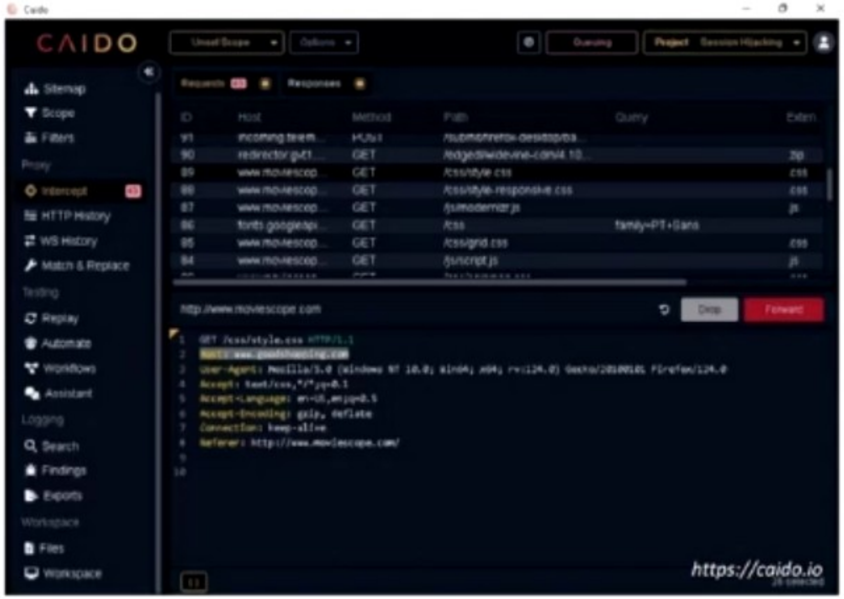


Method	Origin	Path	Status
GET	http://www.moviescope...	/js/script.js	200 OK
GET	http://www.moviescope...	/css/common.css	200 OK
GET	http://www.moviescope...	/css/style.css	200 OK
GET	http://fonts.googleapis...	/css?family=PT+Sans	404 Not Found
GET	http://www.moviescope...	/css/grid.css	200 OK
GET	http://www.moviescope...	/	200 OK
GET	http://ipv6.msftconnect...	/connecttest.txt	
GET	http://www.msftconnec...	/connecttest.txt	200 OK

https://github.com

Caido

Caido is a web **security auditing** toolkit that security professionals can use to intercept and view HTTP requests in **real-time** while browsing



ID	Host	Method	Path	Query	Exten
91	incoming.telem...	POST	/submit/firefox-desktop/ba...		
90	redirector.gvt1...	GET	/edgedl/widevine-cdm/4.10...		zip
89	www.moviescop...	GET	/css/style.css		css
88	www.moviescop...	GET	/css/style-responsive.css		css
87	www.moviescop...	GET	/js/modernizr.js		js
86	fonts.googleapi...	GET	/css	family=PT+Sans	
85	www.moviescop...	GET	/css/grid.css		css
84	www.moviescop...	GET	/js/script.js		js

http://www.moviescope.com

```
GET /css/style.css HTTP/1.1
Host: www.goodshopping.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:124.0) Gecko/20100101 Firefox/124.0
Accept: text/css,*/*;q=0.1
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://www.moviescope.com/
```

https://caido.io

Some more tools are:

- **bettercap** *https://www.bettercap.org*
- **OWASP ZAP** *https://www.zaproxy.org*
- **WebSploit Framework** *https://sourceforge.net*
- **sslstrip** *https://pypi.org*
- **Burp Suite** *https://portswigger.net*

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Objective 04

Explain Session Hijacking Countermeasures

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Protecting against Session Hijacking

01 Use **Secure Shell (SSH)** or **OpenSSH** to create a secure communication channel

02 Implement the **log-out functionality** for the user to end the session

03 Generate a **session ID** after a successful login and accept only session IDs generated by the server only

04 Ensure that data in transit is **encrypted** and implement the **defense-in-depth** mechanism

05 Use **string** or **long random numbers** as session keys

06 Switch from a hub network to a **switch network** to reduce the risk of session hijacking attacks

07 Implement **timeout()** to destroy sessions when expired

08 Avoid including the session ID in the **URL** or **query string**

09 Ensure that **client-side** and **server-side** protection software are in the active state and up-to-date

10 Use **strong authentication** (such as Kerberos) or peer-to-peer virtual private networks (VPNs)

11 Configure appropriate **internal** and **external spoof rules** on gateways

12 Use **IDS products** or **ARPwatch** for monitoring ARP cache poisoning

13 Enable browsers to **verify website authenticity** using network notary servers

14 Use SFTP, AS2 managed file transfer, or FTPS to send data using **encryption** and **digital certificates**

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

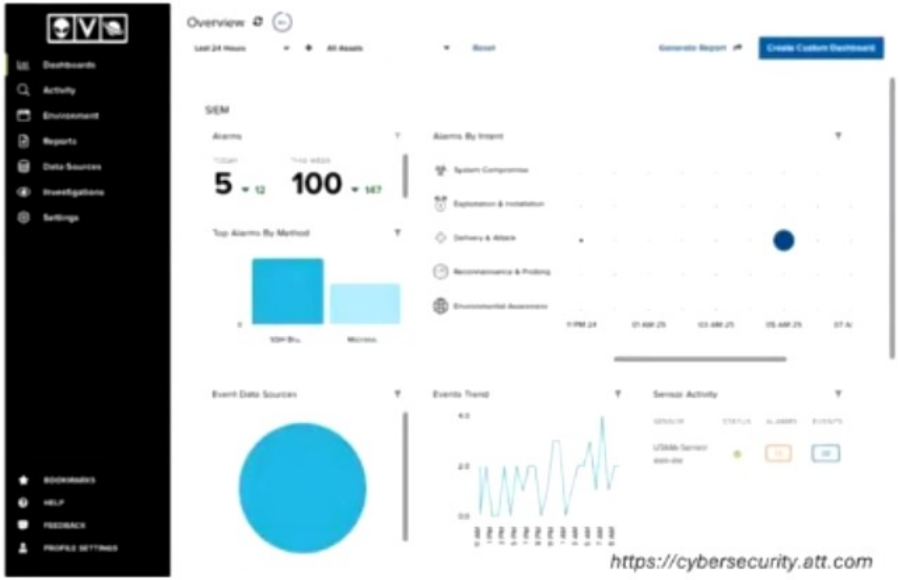
Session Hijacking Detection Tools

USM Anywhere

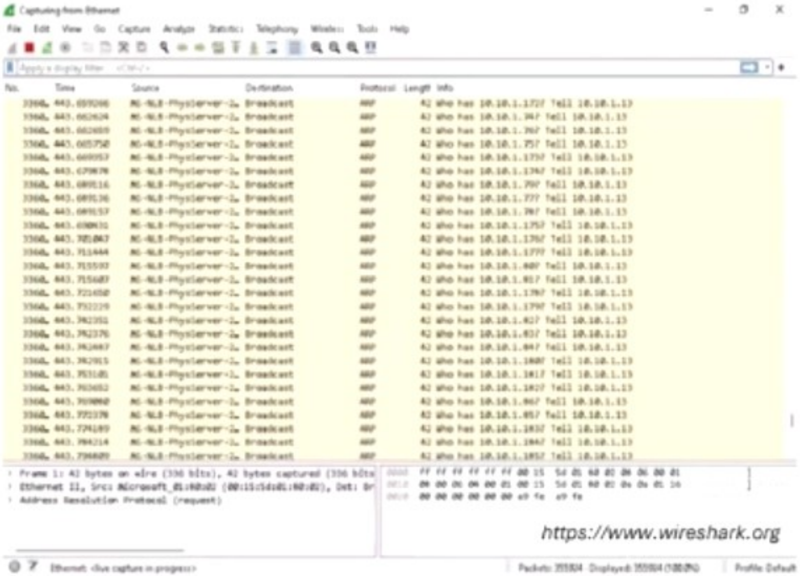
USM Anywhere delivers **threat detection**, incident response, and compliance management across cloud, on-premises, etc.

Wireshark

Wireshark allows you to **capture and interactively browse the traffic** running on a computer network



https://cybersecurity.att.com


Capturing from Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info

https://www.wireshark.org

Session Hijacking Detection Tools:

Quantum Intrusion Prevention System (IPS) (*https://www.checkpoint.com*)

SolarWinds Security Event Manager (https://www.solarwinds.com)

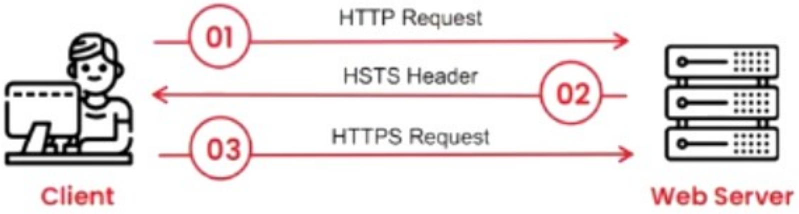
IBM Security Network Intrusion Prevention System (https://www.ibm.com)

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Approaches to Prevent Session Hijacking

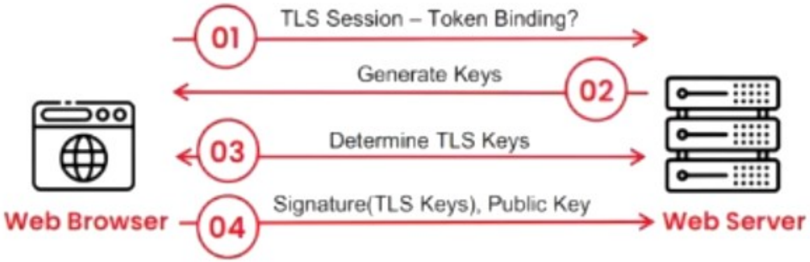
HTTP Strict Transport Security (HSTS)

- HTTP Strict Transport Security (HSTS) is a **web security policy** that protects HTTPS websites against MITM attacks
- It allows web servers to **enforce web browsers** to interact with it using secure HTTPS protocol



Token Binding

- When a user logs on to a web application, it generates a cookie with an **SID**, called a **token**
- Token binding **protects client–server communications** against session hijacking attacks

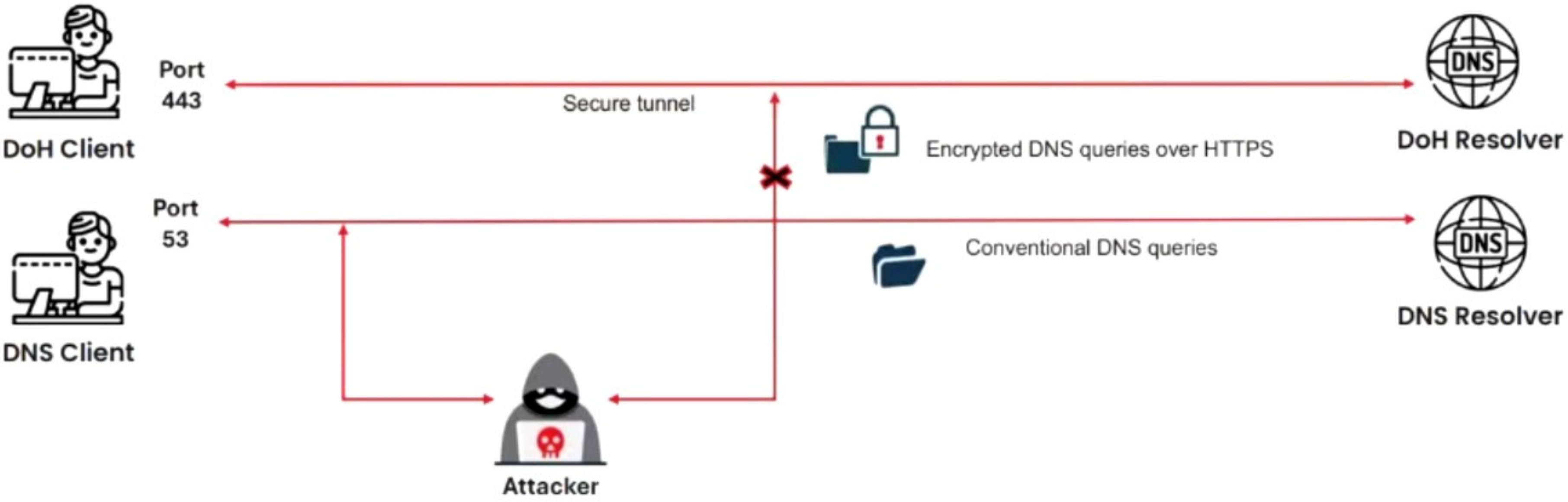


Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Approaches to Prevent MITM Attacks

DNS over HTTPS

- DNS over HTTPS (DoH) is an enhanced version of DNS protocol, which is used to **prevent snooping** of user's web activities or DNS queries during the DNS lookup process
- The web queries and traffic are sent through **encrypted HTTPS** via port 443



Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

IPsec

- IPsec is a protocol suite developed by the IETF for **securing IP communications** by **authenticating** and **encrypting** each IP packet of a communication session
- It is deployed widely to implement **VPNs** and for **remote user access** through dial-up connection to private networks

IPsec Authentication and Confidentiality

- IPsec uses two different security services for authentication and confidentiality
  - **Authentication Header (AH)**: Provides the data authentication of the sender
  - **Encapsulation Security Payload (ESP)**: Provides both the data authentication and encryption (confidentiality) of the sender

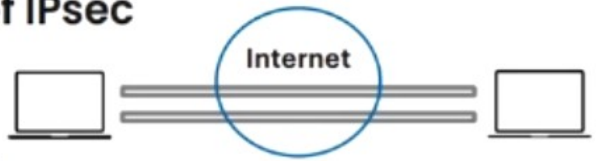
Benefits of IPsec

- Network-level peer authentication
- Data origin authentication
- Data integrity
- Data confidentiality (encryption)
- Replay protection

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

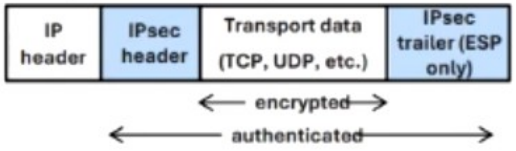
IPsec (Cont'd)

Modes of IPsec

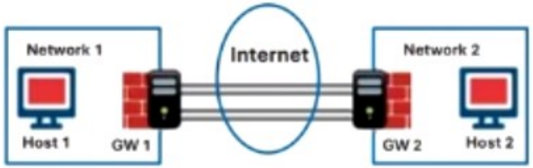


Transport Mode

Transport – mode encapsulation

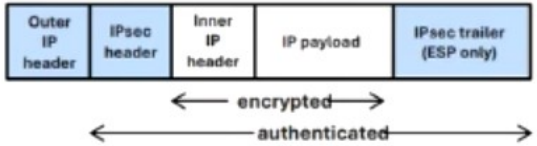


IP header	IPsec header	Transport data (TCP, UDP, etc.)	IPsec trailer (ESP only)



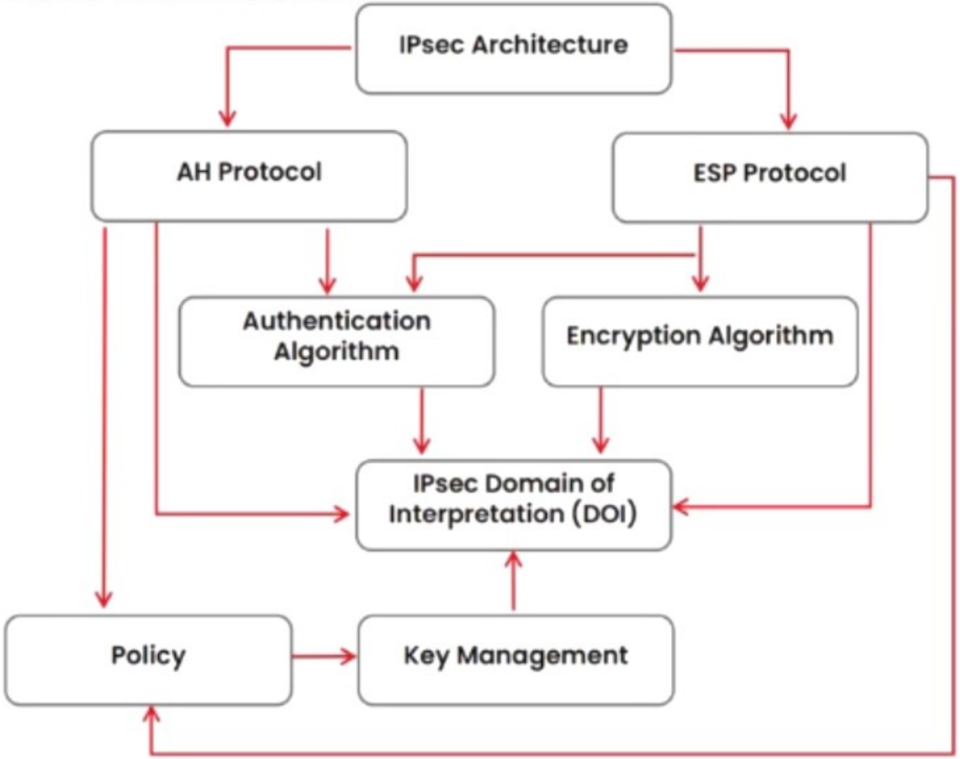
Tunnel Mode

Tunnel – mode encapsulation



Outer IP header	IPsec header	Inner IP header	IP payload	IPsec trailer (ESP only)

IPsec Architecture



Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org

Module **Summary**

- In this module, we have discussed the following:
  - ✓ Session hijacking concepts and different types of session hijacking
  - ✓ Application-level and network-level session hijacking attacks
  - ✓ Various session hijacking tools
  - ✓ How to detect, protect, and defend against session hijacking attacks, as well as various session hijacking detection and prevention tools
  - ✓ We concluded with a detailed discussion on various countermeasures to be employed to prevent session hijacking attempts by threat actors
- In the next module, we will discuss in detail how attackers, as well as ethical hackers and pen-testers, evade network security components such as IDSs and firewalls to compromise the infrastructure

Copyright © EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited. For more information, visit eccouncil.org