ANALYSIS OF SESSION HIJACKING via MITM USING OWASP ZAP

Submitted by

Group Number: 10

Abishek E[CB.EN.U4CSE22001]

Aditya R [CB.EN.U4CSE22004]

Guhanesh T[CB.EN.U4CSE22015]

Prashanna R [CB.EN.U4CSE22036]

COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF COMPUTING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112

April 2025

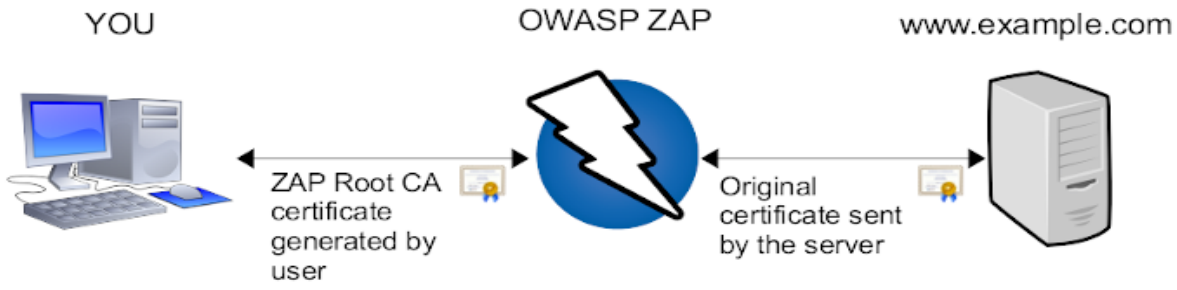
Contents

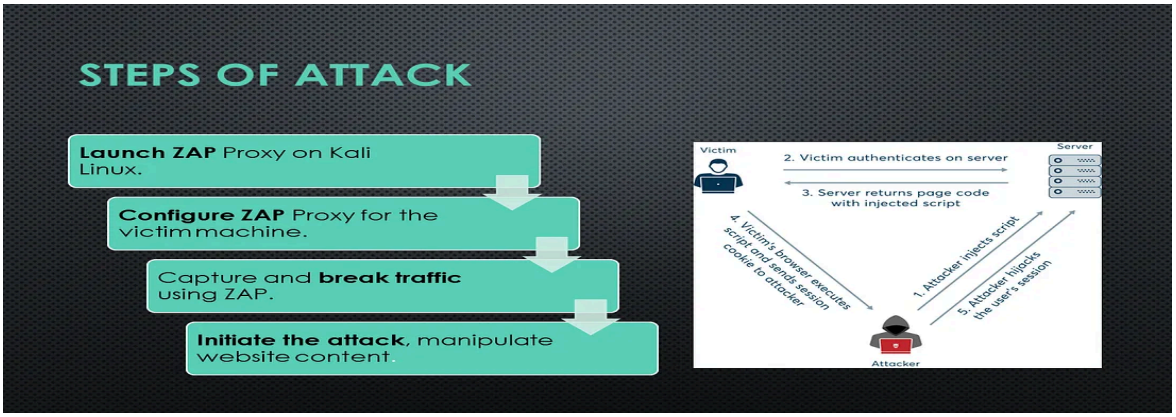
Section 1: Introduction

In today's interconnected world, securing user sessions is a critical aspect of cybersecurity. With countless applications relying on session tokens to maintain user state, any compromise can result in unauthorized access and significant data breaches.

Cybersecurity threats range from injection attacks and cross-site scripting (XSS) to advanced session hijacking. Among these, session hijacking is particularly dangerous as it directly leads to account takeover—granting an attacker the same privileges as a legitimate user.

Tools such as OWASP ZAP are essential in detecting vulnerabilities and simulating attacks. They help organizations identify misconfigurations and weak security practices (like missing Secure, HttpOnly, and SameSite flags) that leave session tokens exposed.This report demonstrates how an attacker can perform a session hijacking attack via MitM using OWASP ZAP. It also discusses how OWASP ZAP not only identifies vulnerabilities but also aids in implementing robust mitigation strategies.





Section 2: Session Hijacking via MitM

Session hijacking is a critical cybersecurity threat that involves intercepting and taking over an active user session. Literature on the subject consistently highlights that once a session token is compromised, the attacker can bypass authentication controls. Research shows that the lack of properly secured cookies (missing Secure, HttpOnly, and SameSite attributes) is a primary contributor to such vulnerabilities.

Key Research Findings and Methodologies

- **Intercepting Traffic:** Using intercepting proxies like OWASP ZAP, attackers can capture session tokens when they are transmitted in plaintext.

- **Token Injection:** Once intercepted, session tokens can be injected into a different browser session to impersonate the legitimate user.

- **Mitigation Techniques:** Research emphasizes the importance of cookie flagging, token regeneration, and multi-factor authentication (MFA) to limit the window of opportunity for an attacker.

Notable Real-World Incidents

Notable incidents, such as the Firesheep attack (which exploited insecure HTTP sessions on public Wi-Fi), illustrate how session hijacking can lead to mass compromise of user accounts. These incidents have spurred enhanced security practices and the adoption of tools like OWASP ZAP in penetration testing environments.

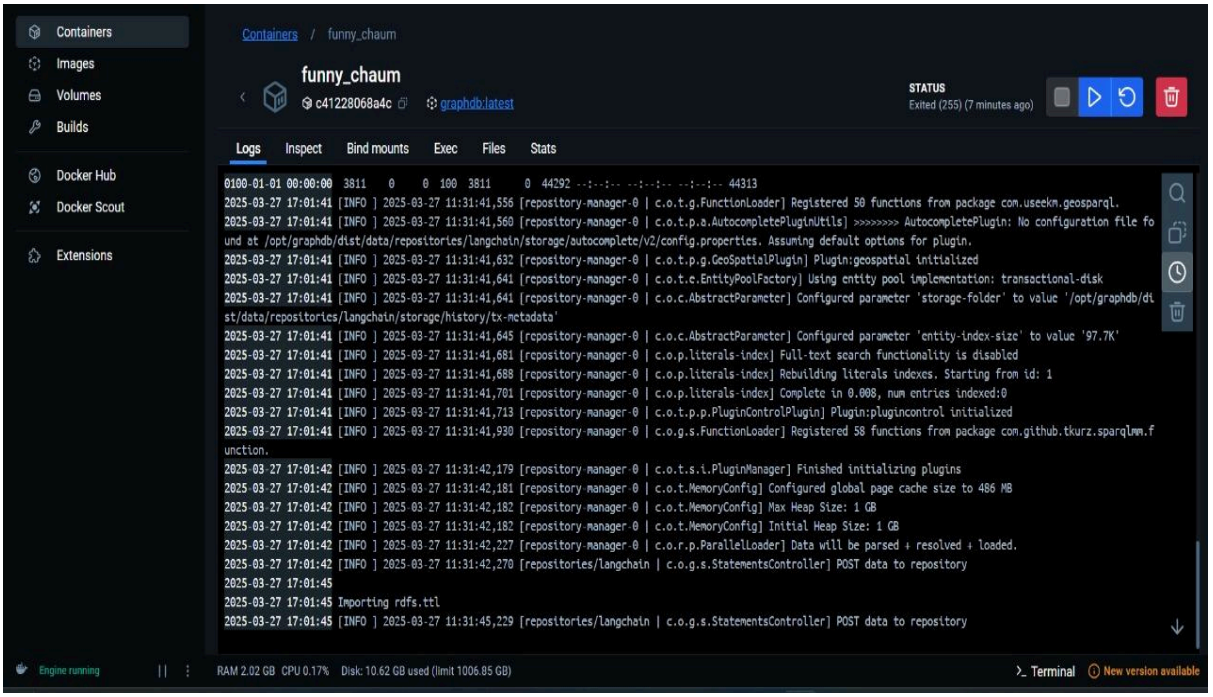
Impact and Security Challenges

The impact of a successful session hijacking attack includes:

- **Full Account Takeover:** The attacker gains complete access to the victim's account.

- **Data Breach:** Sensitive information may be exposed.

- **Loss of Trust:** Compromises can lead to significant reputational damage. The primary challenge is ensuring that session tokens are secured both in transit and at rest.



Section 3: Overview of OWASP ZAP

Description of OWASP ZAP

OWASP ZAP (Zed Attack Proxy) is a free, open-source security testing tool maintained by the Open Web Application Security Project (OWASP). It is designed for finding security vulnerabilities in web applications during the development and testing phases.

Developer and Version Details

Developed by the OWASP community, OWASP ZAP is continuously updated to address the latest threats. It is widely adopted in both academic and professional environments for its effectiveness and community support.

Working Mechanism

OWASP ZAP works as an intercepting proxy that sits between the tester's browser and the target application. By intercepting HTTP/HTTPS traffic, ZAP can capture requests, analyze responses, and detect vulnerabilities such as insecure cookie configurations or missing security headers.

Key Functionalities and Features

- **Intercepting Proxy:** Captures and modifies requests/responses in real time.
- **Active and Passive Scanning:** Automatically scans for common vulnerabilities.
- **Session Analysis:** Identifies insecure session tokens and potential hijacking vectors.
- **Automated Alerts:** Provides recommendations for mitigating detected vulnerabilities.
- **Extensibility:** Supports plugins and scripts for custom testing scenarios.

Section 4: Detailed Analysis of Session Hijacking via MitM Using OWASP ZAP

Nature of the Attack:

Session hijacking via MitM occurs when an attacker intercepts the communication channel between a client and a web server to capture session tokens (e.g., cookies or JWTs). Once the token is intercepted, the attacker can inject it into their own browser session and impersonate the legitimate user. This technique is especially effective in environments where data is transmitted in plaintext or where cookie attributes (Secure, HttpOnly, SameSite) are misconfigured. The MITM method splits the original TCP connection into two separate connections—one between the client and attacker, and the other between the attacker and server—allowing for both passive observation and active modification of data packets.

Existing Mitigation Solutions:

Traditionally, defenses against session hijacking include:

- **Proper Cookie Attributes:** Setting Secure, HttpOnly, and SameSite flags prevents client-side script access and ensures cookies are only transmitted over secure channels.
- **Encryption (HTTPS/TLS):** Encrypting the data in transit helps protect session tokens from interception.
- **Token Regeneration and Expiry:** Regularly rotating session tokens and enforcing strict expiration policies reduce the attack window.
- **Multi-Factor Authentication (MFA):** Adding an extra verification layer helps mitigate the damage even if the session token is compromised.

Many organizations also employ network-level tools such as Intrusion Detection Systems (IDS) to monitor for unusual network traffic patterns that could indicate a MITM attack.

Detection with OWASP ZAP:

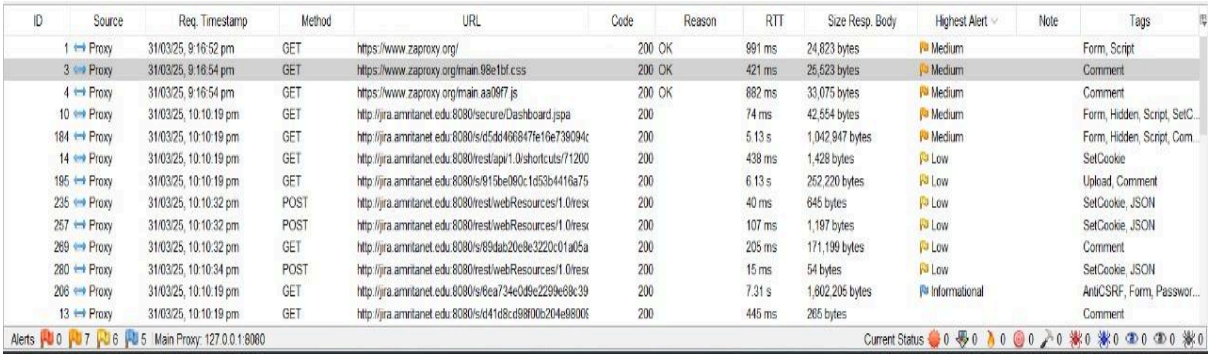
OWASP ZAP is an effective tool for detecting vulnerabilities that can lead to session hijacking. Its key features include:

- **Intercepting Proxy Setup:** By configuring ZAP as a proxy (e.g., 127.0.0.1:8080), all HTTP/HTTPS traffic between the browser and the web application is intercepted.

- **Break Mode and Manual Inspection:** ZAP's break mode allows testers to pause and inspect live traffic. During a session hijacking test, you can observe session cookies in the HTTP headers and verify whether security flags are missing.

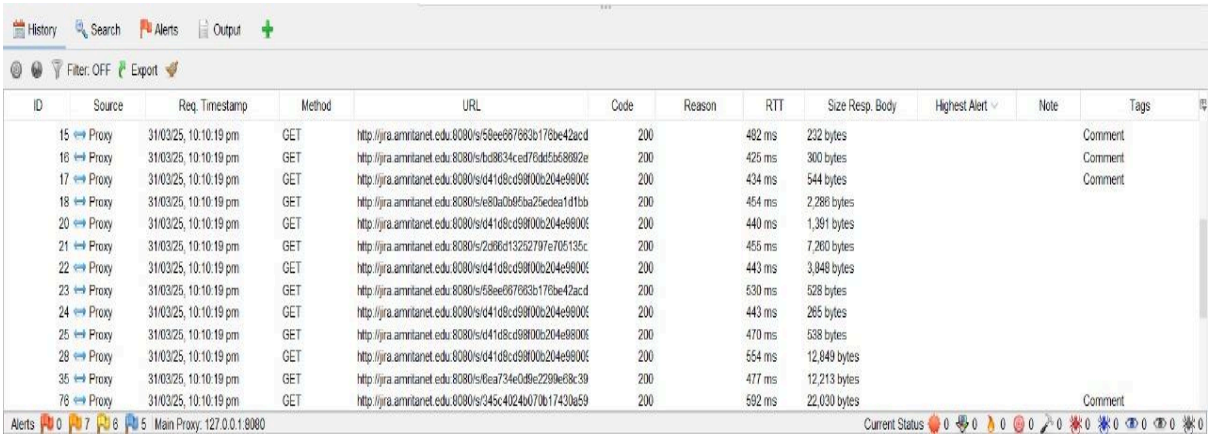
- **Automated Vulnerability Alerts:** ZAP scans for insecure cookie configurations and flags alerts when cookies are transmitted without proper security attributes. This provides actionable feedback for remediation.

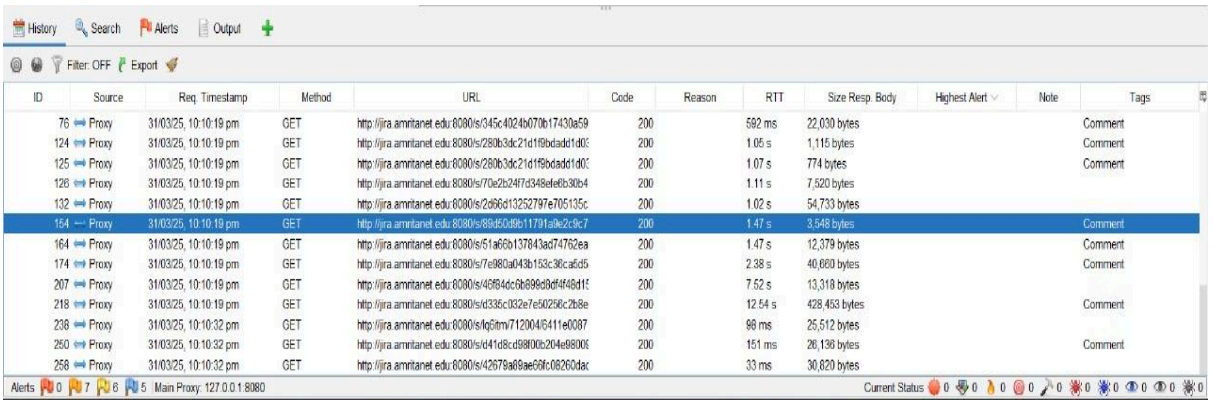
- **Simulation of Token Injection:** Testers can manually inject captured tokens into a secondary session to demonstrate the hijacking risk.



ID	Source	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
1	Proxy	31/03/25, 9:16:52 pm	GET	https://www.zaproxy.org/	200	OK	991 ms	24,823 bytes	Medium		Form, Script
3	Proxy	31/03/25, 9:16:54 pm	GET	https://www.zaproxy.org/main.98e1bf.css	200	OK	421 ms	26,623 bytes	Medium		Comment
4	Proxy	31/03/25, 9:16:54 pm	GET	https://www.zaproxy.org/main.aa0bf7.js	200	OK	882 ms	33,075 bytes	Medium		Comment
10	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/secure/Dashboard.jspa	200		74 ms	42,554 bytes	Medium		Form, Hidden, Script, SetC...
184	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/s/d5dd466847fe16e73809dc	200		5.13 s	1,042,947 bytes	Medium		Form, Hidden, Script, Com...
14	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/rest/api/1.0/shortcuts/71200	200		438 ms	1,428 bytes	Low		SetCookie
195	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/s/915be090c1d53b4416a75	200		6.13 s	252,220 bytes	Low		Upload, Comment
235	Proxy	31/03/25, 10:10:32 pm	POST	http://jira.amritanet.edu:8080/rest/webResources/1.0/res	200		40 ms	645 bytes	Low		SetCookie, JSON
257	Proxy	31/03/25, 10:10:32 pm	POST	http://jira.amritanet.edu:8080/rest/webResources/1.0/res	200		107 ms	1,197 bytes	Low		SetCookie, JSON
269	Proxy	31/03/25, 10:10:32 pm	GET	http://jira.amritanet.edu:8080/s/89dab20e8e3220c01a05a	200		205 ms	171,199 bytes	Low		Comment
280	Proxy	31/03/25, 10:10:34 pm	POST	http://jira.amritanet.edu:8080/rest/webResources/1.0/res	200		15 ms	54 bytes	Low		SetCookie, JSON
206	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/s/6ea734e0d9e2299e68c39	200		7.31 s	1,602,206 bytes	Informational		AntiCSRF, Form, Passwor...
13	Proxy	31/03/25, 10:10:19 pm	GET	http://jira.amritanet.edu:8080/s/d41d8cd98f00b204e98006	200		445 ms	265 bytes			Comment

Alerts 0 7 6 5 Main Proxy: 127.0.0.1:8080 Current Status 0 0 0 0 0 0 0 0 0 0 0 0





Proposed Solution:

The solution we propose enhances existing mitigation strategies by integrating proactive defenses with real-time detection capabilities:

1. **Enhanced Cookie Security:** Beyond enforcing Secure, HttpOnly, and SameSite flags, our approach advocates for dynamic session token regeneration upon significant actions or at periodic intervals.

2. **Real-Time Anomaly Detection:** Combining OWASP ZAP's alert system with behavioral analysis (e.g., monitoring for IP address or user agent changes) can help quickly identify and terminate hijacked sessions.

3. **Automated Response and Token Binding:** Implementing automated responses that immediately invalidate tokens suspected of being hijacked (using techniques such as IP binding or device fingerprinting) further minimizes exposure.

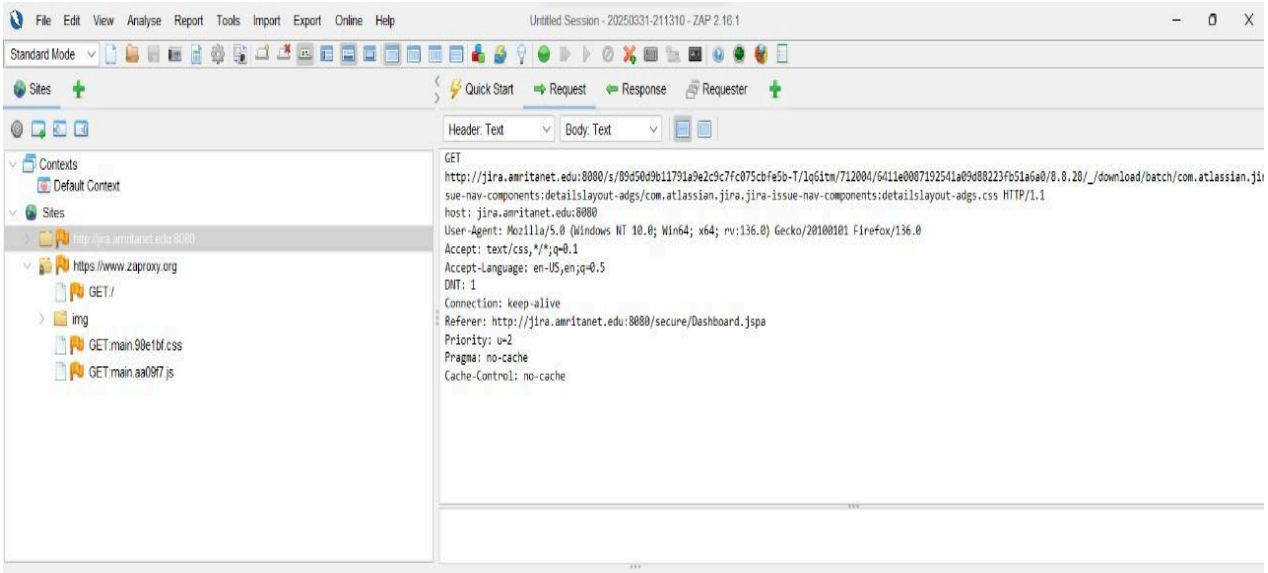
4. **Integration with Multi-Factor Authentication:** By integrating MFA directly with session management, even if a token is intercepted, unauthorized access is prevented.

How the Proposed Solution Differs:

- **Proactive Versus Reactive:** While traditional methods rely on periodic checks and static configurations, the proposed solution employs continuous monitoring and dynamic token management to preemptively cut off hijacked sessions.

- **Automated Response:** Instead of solely alerting developers to vulnerabilities, the solution includes automated remediation (e.g., immediate token invalidation and user re-authentication) when suspicious activity is detected.

- **Enhanced User Session Binding:** By using additional parameters (like IP address or device fingerprinting) for token validation, the system reduces the risk of session token reuse from a different location or device—a gap in many standard implementations.



Section 5: Comparative Analysis

Comparison with Other Cybersecurity Tools

OWASP ZAP is often compared with tools such as Burp Suite and Acunetix. While Burp Suite offers a more polished commercial experience and Acunetix provides automated scanning for enterprise environments, OWASP ZAP stands out for its:

- **Cost-Effectiveness:** Being free and open-source.
- **Community Support:** Frequent updates and contributions from security professionals.
- **Flexibility:** Customizable through add-ons and scripts.

Strengths and Weaknesses

Strengths:

- Extensive plugin ecosystem.
- Ease of integration into CI/CD pipelines.
- Robust scanning capabilities for a wide range of vulnerabilities.

Weaknesses:

- May require manual configuration for advanced scenarios.
- User interface can be less intuitive compared to commercial tools.
- Occasional performance lags with very large applications.

Best Use Cases

OWASP ZAP is best suited for:

- **Development Environments:** Where cost-effective, frequent testing is needed.

- **Small to Medium Enterprises:** That need robust vulnerability assessments without heavy investments.

- **Educational Purposes:** Training ethical hackers and developers in recognizing common vulnerabilities.

Section 6: Challenges and Limitations

Challenges Faced While Using OWASP ZAP

- **Configuration Complexity:** Proper proxy and certificate configuration is required to intercept HTTPS traffic.

- **Learning Curve:** New users may find the array of options and settings overwhelming.

- **Resource Intensive:** Active scanning on large applications can be time-consuming.

Scenarios Where the Tool May Not Be Effective

- **Encrypted or Obfuscated Traffic:** When applications use advanced encryption or token obfuscation, ZAP may struggle to intercept and analyze traffic effectively.

- **Highly Customized Environments:** Applications with nonstandard session management mechanisms might require custom scripts or additional

tools for full analysis.

Possible Improvements Suggested in Research Studies

- **Enhanced Automation:** More streamlined scanning and reporting features.
- **Integration with Machine Learning:** To predict potential vulnerabilities based on historical data.
- **User Interface Enhancements:** Making it more intuitive for both beginners and experts.

Section 7: Conclusion and Future Advancements

This report has detailed the mechanics, impact, and mitigation strategies for a session hijacking attack via MitM using OWASP ZAP. Key takeaways include:

- **High Impact:** Session hijacking poses severe risks, leading to unauthorized access and data breaches.
- **Critical Role of Proper Cookie Management:** Implementing Secure, HttpOnly, and SameSite flags, along with other security measures, is paramount.
- **OWASP ZAP as a Dual-Use Tool:** Not only can it simulate attacks, but it also educates and guides developers in remediating vulnerabilities.

Future Advancements

Looking ahead, integrating more sophisticated automation, machine learning analytics, and better UI/UX designs could further enhance OWASP ZAP's capabilities. These improvements will help security teams quickly adapt to evolving threats and protect increasingly complex web applications.

References

- **OWASP ZAP Official Documentation – Comprehensive resource on setup, configuration, and best practices.**

- **Research papers and case studies on session hijacking and MitM attacks.**

- **Documentation and security advisories for OWASP Juice Shop.**