WEB-APP PENTERATION TESTING

Vulnerability Analyst / Penetration Testing

WEB-APP
Penteration

WEB-APP
Penteration

AGENDA

Use links to go to a different page inside your presentation.

Highlight text, click on the link symbol on the toolbar, and select the page in your presentation you want to connect.

WEB-APP
Penteration

INTRODUCTION

This Penetration Testing engagement was conducted on our E-Commerce Application with the objective of identifying vulnerabilities and assessing the overall security posture of the web application.

The scope of the test included a comprehensive evaluation of the site’s external and internal vulnerabilities, focusing on areas such as authentication, data handling, and overall infrastructure security

TESTING PHASES

Reconnaissance

is the initial phase of penetration testing, where information about the target system is gathered using active and passive methods to identify vulnerabilities and entry points.

Scanning

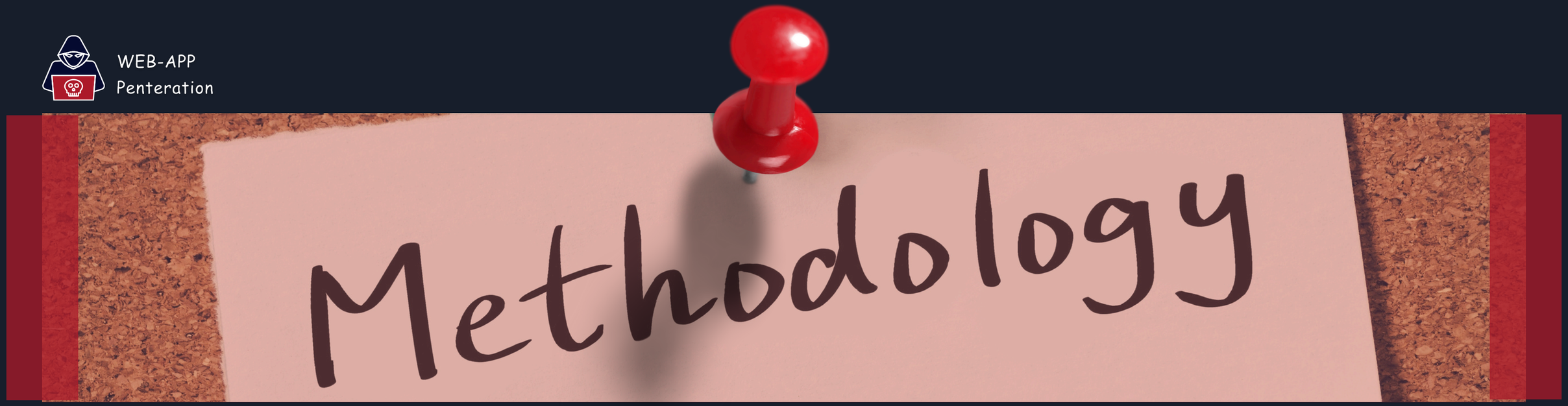
is the second phase of penetration testing, where detailed probing of the target system is done to identify open ports, services, and vulnerabilities using techniques like port scanning, network mapping, and vulnerability scanning.

Exploitation

is the phase in penetration testing where identified vulnerabilities are actively exploited to gain unauthorized access or control over the target system, demonstrating the potential real-world impact of the security flaws.

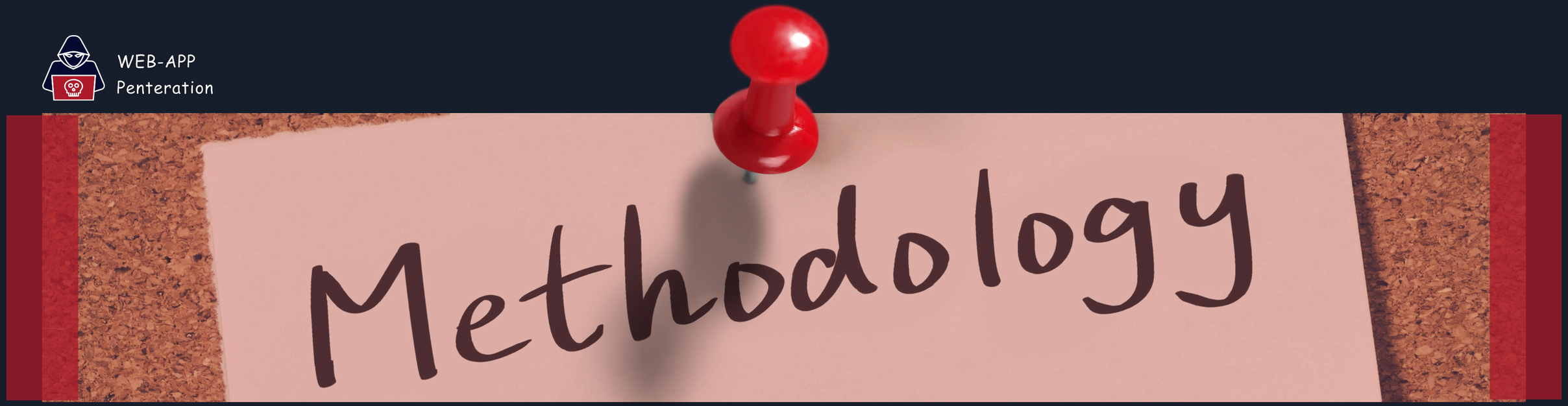
Reporting

is the final phase of penetration testing, where all findings are documented, including vulnerabilities, exploitation methods, and recommendations, to provide a clear overview of the system's security status and steps for remediation.



Our penetration testing followed the OWASP Testing Guide, which provided a comprehensive framework to identify, exploit, and document vulnerabilities. The testing process involved the following key steps:

- Information Gathering: Collect details about the target application, including technologies used, subdomains, and URLs, utilizing tools like Burp Suite or OWASP ZAP.
- Input Validation: Check how the web application handles user inputs, testing for common vulnerabilities such as Cross-Site Scripting (XSS), SQL Injection, and Command Injection by attempting to inject malicious scripts and commands into the input fields.
- Application Mapping: Explore the web app's structure, user roles, input fields, and critical components to identify potential attack vectors.
- Authentication and Session Management: Analyze the security of login mechanisms and session handling. This includes testing password policies, investigating weaknesses in JWT secret keys, and checking vulnerabilities in the password reset mechanism that could allow unauthorized access.

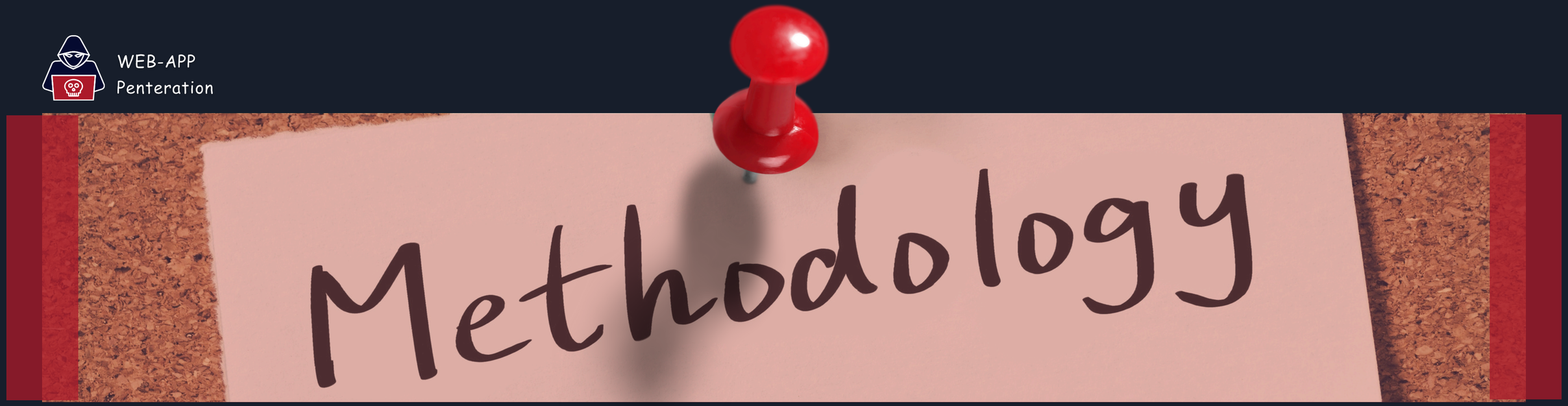


- Access Control: Evaluate the access control mechanisms to ensure that only authorized users can perform specific actions, simulating scenarios where unauthorized users might exploit weaknesses to access restricted areas or functionality.

- Configuration Management: Review the configuration settings of the web application and its infrastructure for security weaknesses, such as local file inclusion and OS command injection, where attackers could potentially access sensitive files or run system commands.

- Error Handling and Logging: Test how the web application handles errors and whether sensitive information is exposed in error messages or logs, including examining server responses for unintended data leaks.

- Vulnerability Identification: Search for common web vulnerabilities, such as SQL Injection, XSS, CSRF, and authentication flaws, using both automated and manual testing.



- Exploitation: Attempt to exploit the identified vulnerabilities to gain unauthorized access or manipulate the application.
- Post-Exploitation: Analyze the impact of successful exploits, including access to sensitive data or control of the system.
- Reporting: Document findings, including exploited vulnerabilities and remediation recommendations, ranked by severity.

This comprehensive methodology ensures a thorough evaluation of the web application’s security, helping to identify and mitigate potential risks effectively.

Why? WE SKIPPED RECON PHASE ?!

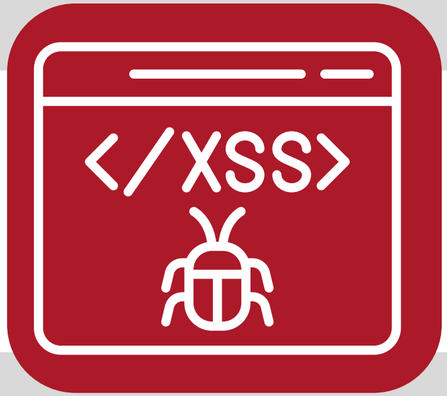
In penetration testing, the Reconnaissance (Recon) phase is typically crucial for gathering initial information about the target. However, in some cases, this phase can be skipped if the necessary information is already available or if the penetration tester has been provided with the essential details. In this scenario, the target was a well-known website, and the required information had already been provided, making the Recon phase unnecessary

OUR FINDINGS

OS Command Injection in API User-Agent

Account Takeover via Reset Password Link

JWT Secret Key Weakness

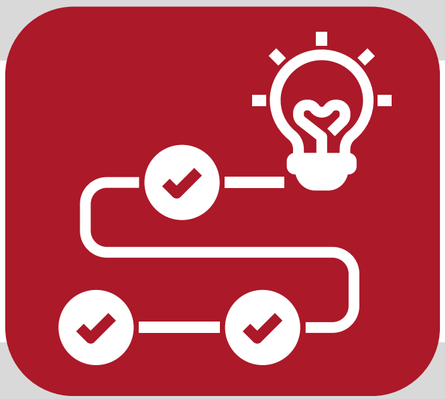


Cross-Site Scripting (XSS)

Open Redirect

OUR FINDINGS

Email Enumeration and Lack of Rate



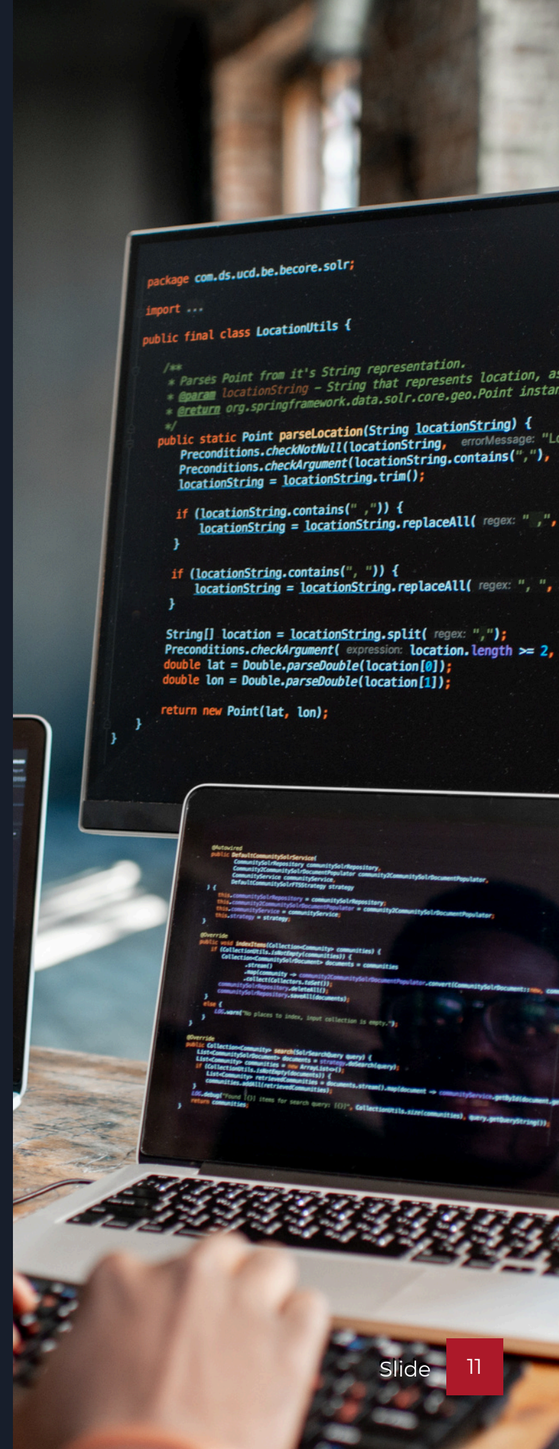
Logic Bug in Payment System

Weak Password Policies in Sign-Up and Reset Mechanism

Path Traversal

ABOUT MITIGATIONS

Mitigations in penetration testing refer to the recommended actions or strategies to reduce or eliminate the risks posed by identified vulnerabilities. These include applying security patches, improving configurations, strengthening access controls, and implementing continuous monitoring.



MITIGATIONS

OS COMMAND INJECTION IN API USER-AGENT

- Avoid the use of direct system commands like exec() or system().

ACCOUNT TAKEOVER VIA RESET PASSWORD LINK

- Ensure that the password reset link is confined to the domain in the host header.

MITIGATIONS

JWT SECRET KEY WEAKNESS

- Use a robust secret key for JWT tokens.
- Validate token expiration.
- Apply strong hashing algorithms.

CROSS-SITE SCRIPTING (XSS)

- Ensure proper sanitization and validation of inputs.
- Apply HTML, JavaScript, and URL encoding practices.
- Implement Content Security Policy (CSP) headers.
- Utilize Web Application Firewall (WAF) solutions such as Cloudflare.

MITIGATIONS

OPEN REDIRECT

- Establish a whitelist for valid URLs.
- Use relative URLs wherever possible instead of absolute URLs.

EMAIL ENUMERATION AND LACK OF RATE

- Display generic error messages.
- Enforce rate-limiting mechanisms, such as a limit of 5 login attempts, followed by a temporary lockout.

MITIGATIONS

LOGIC BUG IN PAYMENT SYSTEM

- Implement backend checks to control the total transaction price.
- Send only the item identifier to the backend, not the total price.

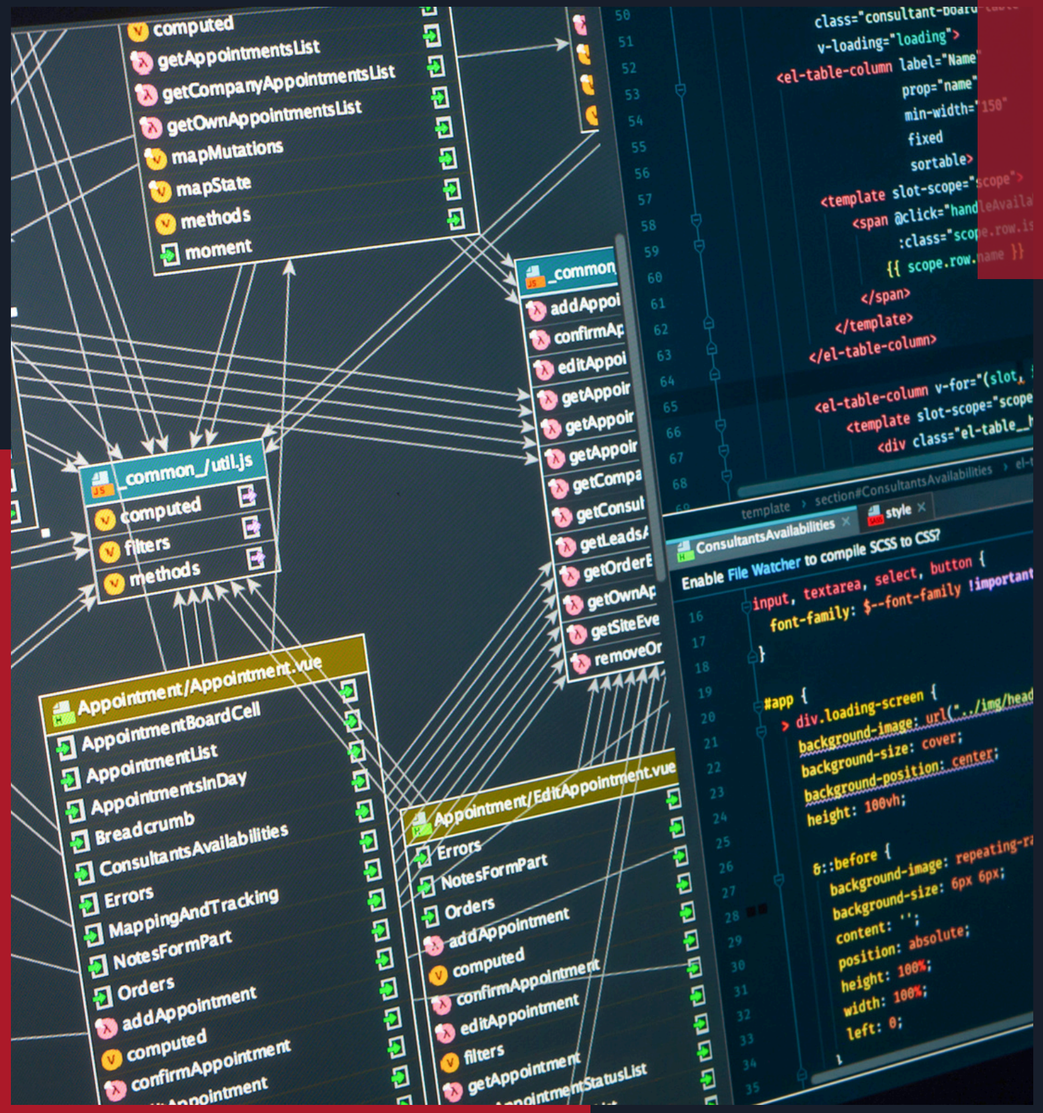
WEAK PASSWORD POLICIES IN SIGN-UP

- Enforce password policies requiring a minimum of 6 characters, including both capital letters and symbols.

MITIGATIONS

PATH TRAVERSAL

- Validate and whitelist inputs.
- Prevent manipulation of file paths on the server.



CONCLUSIONS

This testing was based on the technologies and known threats as of the date of this document. All the security issues discovered during that exercise were analyzed and described in this report. Please note that as technologies and risks change over time, the vulnerabilities associated with the operation of systems described in this report, as well as the actions necessary to reduce the exposure to such vulnerabilities, will also change

OUR TEAM

Hossam Mohamed

Ahmed Elhendawy

Ahmed Mazhar

Kareem Mohamed

Omar Samy

THANK YOU

WEB-APP Penteration Team