AICS-TASK : 4

TITTLE : Explain any 10 Web Server Attacks determine them using images if

available

Name : Rohitha Koganti

1. XML External Entity (XXE) Attack: XXE attacks occur when an application parses XML input from an untrusted source and can lead to information disclosure, server-side request forgery, or denial of service. Attackers can exploit this vulnerability to access internal files or execute arbitrary code.

2. Insecure Deserialization: This vulnerability occurs when an application allows the deserialization of untrusted data. Attackers can manipulate serialized data to execute arbitrary code, leading to remote code execution and potential security breaches.

3. Security Misconfiguration: While OWASP does mention security misconfigurations, it's not as detailed as some specific vulnerabilities. These can include open cloud storage, default credentials, unnecessary services running, and excessive permissions, which can be exploited to gain unauthorized access or expose sensitive data.

4. Cross-Site Request Forgery (CSRF): CSRF attacks trick users into performing unintended actions on a different site where they are authenticated. Attackers can forge requests that perform actions on behalf of the user without their consent, potentially leading to data modification or account takeover.

5. Content Security Policy (CSP) Bypass: CSP is a security feature that helps prevent cross-site scripting (XSS) and other code injection attacks. However, misconfigured CSP policies or bypass techniques can allow attackers to execute malicious scripts, compromising the security of the web application.

6. Server-Side Template Injection (SSTI): SSTI occurs when an application allows users to inject code into templates that are executed on the server-side. Attackers can exploit this vulnerability to execute arbitrary code, potentially compromising the server and data.

7. Insecure File Uploads: When an application allows users to upload files without proper validation and controls, it can lead to remote code execution, denial of service, or the spread of malware. Attackers can upload malicious files or manipulate file extensions to exploit this vulnerability.

8. Insecure Cross-Origin Resource Sharing (CORS): CORS misconfigurations can allow unauthorized websites to access sensitive data or functionality on a different domain, leading to data exposure or unauthorized actions on behalf of the user.

9. HTTP Security Headers Missing: Not setting essential security headers like Strict-Transport-Security (HSTS), X-Content-Type-Options, and X-Frame-Options can leave the web application vulnerable to various attacks, including protocol downgrade attacks and clickjacking.

10. Insufficient Password Policies: Weak password policies, such as not enforcing strong passwords, lack of account lockout mechanisms, or inadequate password storage, can expose user accounts to brute-force attacks and unauthorized access.

Addressing these web application vulnerabilities is crucial for maintaining the security and integrity of web-based systems. Web developers and security professionals should actively monitor and mitigate these risks to protect against potential threats and attacks.