Task 5: Web server attacks

1. SQL Injection:

Description: Attackers inject malicious SQL queries through input fields to manipulate a web application's database.

Method: By entering malicious SQL code into input forms.

Prevention: Use parameterized queries or prepared statements, input validation and sanitization, and employ a web application firewall (WAF) to block suspicious SQL queries.

2. Cross-Site Scripting (XSS):

Description: Attackers inject malicious scripts into web pages viewed by other users, stealing their data or executing actions on their behalf.

Method: By inserting malicious scripts into input fields or URLs.

Prevention: Validate and sanitize user inputs, implement content security policy (CSP), and escape output data in web pages.

3. Cross-Site Request Forgery (CSRF):

Description: Attackers trick authenticated users into performing unintended actions without their consent on a different site.

Method: Crafting a malicious link or form that executes actions on another site without the user's knowledge.

Prevention: Use anti-CSRF tokens, check the "Referer" header, and employ same-site cookie attributes.

4. Distributed Denial of Service (DDoS):

Description: Attackers overwhelm a web server with a flood of traffic, rendering it unavailable.

Method: Using a network of compromised devices to generate massive traffic towards the target server.

Prevention: Implement rate limiting, traffic filtering, utilize a CDN, and set up DDoS protection services.

5. Server-Side Request Forgery (SSRF):

Description: Attackers manipulate a web server to make unauthorized requests to internal resources.

Method: By tricking the server into sending requests to unintended resources.

Prevention: Validate and sanitize user inputs, restrict outgoing requests using firewalls, and use whitelists for allowed resources.

6. Remote File Inclusion (RFI) and Local File Inclusion (LFI):

Description: Attackers exploit poor file inclusion practices to execute malicious code or access sensitive files.

Method: Manipulating URLs to include and execute external or local files.

Prevention: Avoid dynamic file inclusion, validate user inputs, and use secure file access methods.

7. Brute Force Attacks:

Description: Attackers attempt to guess usernames and passwords to gain unauthorized access to a web server.

Method: Repeatedly trying different combinations of credentials until successful.

Prevention: Enforce strong password policies, implement account lockout mechanisms, and use CAPTCHA or rate limiting.

8. Server Misconfiguration:

Description: Attackers exploit improperly configured server settings or services to gain unauthorized access or expose sensitive information.

Method: Exploiting default settings, unnecessary open ports, or weak security configurations.

Prevention: Regularly update and patch server software, disable unnecessary services, and follow security best practices.

9. File Upload Vulnerabilities:

Description: Attackers exploit weak file upload mechanisms to upload and execute malicious files on a server.

Method: Uploading files with malicious content, such as executable scripts.

Prevention: Validate and restrict file types, implement proper file permission settings, and scan uploaded files for malware.

10. Clickjacking:

Description: Attackers overlay deceptive elements on a web page to trick users into clicking on hidden, malicious elements.

Method: Hiding malicious content under legitimate content or transparent layers.

Prevention: Implement X-Frame-Options HTTP header, use frame-busting scripts, and ensure proper CSS styling to prevent element overlay.

Preventing these attacks requires a combination of secure coding practices, regular security audits, patching and updating software, and employing various security mechanisms. Keep in mind that the security landscape is constantly evolving, so staying informed about the latest threats and best practices is crucial.