Command Injection

Command Injection

- Command injection vulnerabilities in the context of web application penetration testing occur when an attacker can manipulate the input fields of a web application in a way that allows them to execute arbitrary operating system commands on the underlying server.

- This type of vulnerability is a serious security risk because it can lead to unauthorized access, data theft, and full compromise of the web server.

Command Injection - Causes

- User Input Handling: Web applications often take user input through forms, query parameters, or other means.
- Lack of Input Sanitization: Insecurely coded applications may fail to properly validate, sanitize, or escape user inputs before using them in system commands.
- Injection Points: Attackers identify injection points, such as input fields or URL query parameters, where they can insert malicious commands.

Command Injection - Exploitation

- Malicious Input: Attackers craft input that includes special characters, like semicolons, pipes, backticks, and other shell metacharacters, to break out of the intended input context and inject their commands.

- Command Execution: When the application processes the attacker's input, it constructs a shell command using the malicious input.

- The server, believing the command to be legitimate, executes it in the underlying operating system.

Command Injection - Impact

- Unauthorized Execution: Attackers can execute arbitrary commands with the privileges of the web server process. This can lead to unauthorized data access, code execution, or system compromise.

- Data Exfiltration: Attackers can exfiltrate sensitive data, such as database content, files, or system configurations.

- System Manipulation: Attackers may manipulate the server, install malware, or create backdoors for future access.

Demo: Command Injection

PHP Code Injection

PHP Code Injection

- PHP code injection vulnerabilities, also known as PHP code execution vulnerabilities, occur when an attacker can inject and execute arbitrary PHP code within a web application.

- These vulnerabilities are a serious security concern because they allow attackers to gain unauthorized access to the server, execute malicious actions, and potentially compromise the entire web application.

PHP Code Injection - Exploitation

- Malicious Input: Attackers craft input that includes PHP code snippets, often enclosed within PHP tags (<?php ... ?>) or backticks (`).

- Code Execution: When the application processes the attacker's input, it includes the injected PHP code as part of a PHP script that is executed on the server.

- This allows the attacker to run arbitrary PHP code in the context of the web application.

Demo: PHP Code Injection