Concept Overview

File Inclusion Attacks

A file inclusion attack is a type of approach which to utilize a vulnerability that is most commonly found to affect web applications that rely on a scripting run time. This issue is caused when an application builds a path to executable code using an attacker-controlled variable in a way that allows the attacker to control which file is executed at run time. A file include vulnerability is distinct from a generic Directory Traversal Attack, in that directory traversal is a way of gaining unauthorized file system access, and a file inclusion vulnerability subverts how an application loads code for execution. Successful exploitation of a file include vulnerability will result in remote code execution on the web server that runs the affected web application.

This can lead following attacks:

- Code execution on the web server
- Cross Site Scripting Attacks (XSS)
- Denial of service (DOS)
- Data Manipulation Attacks

Types of inclusion:

Local File Inclusion

Local File Inclusion (LFI) is also known as Path Traversal. The vulnerability files are only local files i.e. files on the current server can be included for execution. This issue can lead to remote code execution by including a file that contains attacker-controlled data such as the web server's access logs.

Remote File Inclusion

Remote File Inclusion occurs when the URI of a file located on a different server is passed to as a parameter to the PHP function “include”, “include_once” , “require” , or “require_once” . PHP incorporates the content into the pages. If the content happens to be PHP source code, PHP executes tl ⓘ le.

PHP Remote File Inclusion allows and attacker to embed his/her own PHP code inside a vulnerable PHP script, which may lead to disastrous results such as allowing the attacker to execute remote commands on the web server, deface parts of the web or even steal confidential information.

Mitigation

- Strong Input Validation
- A whitelist of acceptable inputs
- Reject any inputs that does not strictly conform to specifications
- For filenames, use stringent whitelists that limits the character set to be used
- Exclude directory separators such as “/”
- Use a whitelist of allowable file extensions
- Environment hardening
- Develop and run your code in the most recent versions of PHP available
- Configure your PHP applications so that it does not use register_globals
- Set allow_url_fopen to false, which limits the ability to include files from remote locations
- Run your code using the lowest privileges
- Use a vetted library or framework that does not allow this weakness.

Reference:

http://www.hackingarticles.in/beginner-guide-file-inclusion-attack-lfirfi/

https://resources.infosecinstitute.com/php-lab-file-inclusion-attacks/

https://www.w3schools.com/

https://www.owasp.org/index.php/Testing_for_Local_File_Inclusion

https://www.acunetix.com