International Journal of Scientific Research in Science, Engineering and Technology
Print ISSN: 2395-1990 | Online ISSN : 2394-4099 (www.ijsrset.com)
doi : https://doi.org/10.32628/IJSRSET218384

Deep Dive into Directory Traversal and File Inclusion Attacks leads to Privilege Escalation

Mrunalsinh Chawda*, Dr. Priyanka Sharma
School of Information Technology, Artificial Intelligence, and Cyber Security, Rashtriya Raksha University, Gandhinagar, Gujarat, India

Article Info
Volume 8, Issue 3
Page Number : 115-120

Publication Issue :
May-June-2021

Article History
Accepted : 10 May 2021
Published: 16 May 2021

ABSTRACT

In Modern Web application directory traversal vulnerability that can potentially allow an attacker to view arbitrary files and some sensitive files. They can exploit identified vulnerabilities or misconfigurations to obtain root privileges. When building the web application, ensure that some arbitrary file is not publicly available via the production server. when an attacker can include. Traversal vulnerabilities this vulnerability exploits the dynamic file include a mechanism that exists in programming frameworks a local file inclusion happens when uncontrolled user input such as form values or headers for example are used to construct a file include paths. By exploiting directory traversal attacks in web servers, they can do anything and with chaining with code injection they can upload a shell into a web server and perform a website defacement attack. Path-traversal attacks take advantage of vulnerable Website parameters by including a URL reference to remotely hosted malicious code, allowing remote code execution and leads to privilege escalation attack.

Keywords - Cyber Security, Information Security, Web Application Security, Privilege Escalation, Remote Code Execution, Directory Traversal, Vulnerability Analysis, LFI, RFI.

I. INTRODUCTION

A remote attacker tries to execute some of the malicious payload to server and try to access unauthorizedly. these kinds of attacks Some web applications need to obtain resources.

on the file system to implement the web application (such as images, static text and so on). They sometimes use parameters to define the resources. When these parameters are user-controlled, not properly sanitized and are used to build the resource path on the file system, security issues may arise. If the web application does not sanitize the parameter properly, an attacker could manipulate it to obtain the contents of any arbitrary file. File inclusion vulnerabilities are divided into Remote and Local, depending on where the file to include is located. This type of vulnerability was very common at that time, as a result of security awareness not being very widespread among developers. However, it is still found in custom scripts where path characters are not stripped from the input the vulnerability is easier to understand if we look at a simple section of PHP code.

Let's assume that the target application changes its context based on attacker attack vector. [1].

Directory traversal is the methodology of containing files, that are already present on the webserver, through the exploitation of vulnerable composition procedures executed in the application. This vulnerability occurs, for example, when a page gets, as input, the path to the file that needs to be included. When this input is not properly sanitized, directory traversal characters can be injected. Although most cases point to vulnerable PHP scripts, we should keep in mind that it is also prevalent in other technologies such as JSP, ASP and others.

The File Inclusion vulnerability authorises the attacker to inject a file, normally exploiting a dynamic file inclusion mechanism performed in the target application. The vulnerability appears due to the use of user-supplied information without proper validation.[2]

The application can maintain a white list of files, that can be included on the page, and then use an identifier to gain access to the aspired file. Any request including an illogical identifier needs to be discarded as this contains an attack surface for malicious users to manipulate the path.

II. RELATED WORK

Security data breaches are a major concern for companies and the government sector. there are some unidentified threats are available in the real world it might be the most critical scenario for business.
The problem occurs because of a lack of proper validation. A number of researches have been conducted on different web application vulnerabilities such as SQL injection, XSS, CSRF, Buffer overflow, broken authentication. Based on CVES or different kind of case studies directory traversal attack and Remote code execution both are the most critical vulnerability in real-time web applications is a programming language designed for Web development and is in use for more than 75 percent of applications on the Internet. Our recent research into directory traversal reveals how hackers use directory traversal "In the wild one. directory traversal vulnerability, for example, compromised 1.2M WordPress blogs via a Tim Thumb, a PHP application.

Our overview provides security teams with the context needed for the importance of awareness of the technique. Analysis of a Directory traversal attack –A visual step-by-step technical analysis of an RFI-infected file shows how shellcode obfuscates the attack vector, highlighting how it can bypass conventional detection and mitigation methods.

The number of vulnerabilities published to the National Vulnerability Database (NVD) increased by 127%with web application vulnerabilities delivering up 51% of all disclosed vulnerabilities for 2017, based on Michael Flanders Research directory traversal or path traversal vulnerabilities which made up more than 22% of vulnerability disclosures and based on is Research many vulnerability scanner default configurations some automation approach not able to detect local file inclusion vulnerability [6].

If used without a qualifier or prefixed with local the term is largely synonymous with read-related directory traversal. Remote file inclusion, on the other hand, is an alternative way to exploit file inclusion vulnerabilities by specifying a URL rather than a valid file path. In some scripting languages, a single, common API opens local files and fetches remote URLs. In these cases, the ability to retrieve the file from an attacker-controlled server may offer substantial benefits, depending on how the data is subsequently processed.

III. DIRECTORY TRAVERSAL EXPLOITATION

Local File Inclusion (LFI) allows an attacker to inject files on a server within the web browser. This vulnerability exists when a web application includes a file without correctly sanitising the input, allowing an attacker to manipulate the input and inject directory traversal payloads and include other files from the webserver. Example of Vulnerable code: The following is a case of vulnerable PHP code to :

```
<?php
$file = $_GET['file'];
if(isset($file))
{
        include("pages/$file");
}
else
{
        include("index.php");
}
```

Figure 1 : vulnerable code

A. ANALYSIS

LFI is the capability to use and execute local files on the server-side. The vulnerability allows a remote user to gain access with a specially crafted request to arbitrary files on the server, including those including confidential data. Simply put, this is a vulnerability of opening files from the server + inadequate filtering, which allows you to open an arbitrary file [2].

1. Searching for File Inclusion:

Searching for parameters There are two options to search for parameters: manual or automatic search.

2. Manual search:

Now I will talk about manual search. Let's assume we have found a GET parameter:

http://domain/folder/index.php?file=gallery

Substitute the string 'index' under parameter:

http://domain/folder/index.php?file=index

If you have opened any of the index. Files (any extension) located on the site, then here we can definitely say that the file is responsible for swapping the file.

3. Automatic Search

In automatic approach you need to perform parameter mining and identify based pattern.

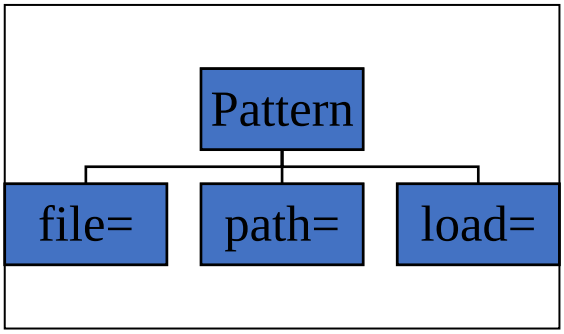

Figure 2: pattern identification

if it's convenient for you. Directives like error reporting, debug (Django) should be configured accordingly. Hacking is simplified when we are no longer dealing blindly, when a hacker can reveal the full path to the web directory, so that in the future, for example, **use** it with sql inj (in this, I'm talking about file_priv). Therefore, it is not necessary to show any errors of the server software or interpreters to the end user. And in the code, the best way would be constructions**:** There are also 404 and similar errors. You should use your own view of these pages, not from the server. And this is a plus not only in terms of security, but general use. And in some case this vulnerability leads to RCE [5].

Vulnerabilities related to the wrapper mechanism implemented in PHP have been discussed for a long time. They are referenced in OWASP TOP 10 and WASC TCv2. However, a number of features of the data encoding implementation led to the fact that even applications designed with security requirements in mind may contain vulnerabilities (including critical ones). In this article, we'll first take a quick look at what PHP wrappers are and how they can be useful for programmers. Then we will analyse

their features, which allow you to bypass the security filters built into the application and implement attacks related to unauthorized access to the file system and the execution of arbitrary code.

B. EXPLOITATION

Directory traversal permits an attacker to inject files on a server within the web browser. This vulnerability exists when a web application includes a file without precisely sanitising the input, passing an attacker to manipulate the input and inject path traversal characters and include additional files from the webserver.

That vulnerabilities allow attackers to connect files on the server through a browser. This flaw is present where there is no correct processing of incoming data, and an attacker can manipulate the incoming information, inject characters like path traversal, and include other files from the web server. If attacker able to perform successfully then he might be read and modify the arbitrary data .it includes all the sensitive information.

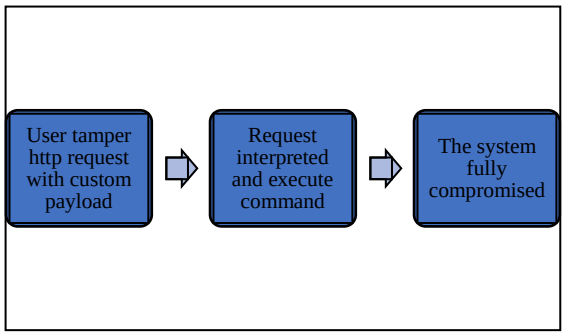

Figure 3: Workflow of attack

These are the parameters we need.

Defining Filters

After we get the list of parameters, we need to check if they have filtering.

1. Zero filtering

Let's try to upload files that we didn't expect to be shown ,The analogue of such a file in link is the file / etc / passwd(In this case, we took http://domain/folder/index.php?file=index.html for the line with the parameter)Moving to the /../ folder means climbing up the hierarchy (more precisely, this is the path traversal vulnerability). Because the etc folder is in the root folder, then we have to reach it by guessing: that is, the more often we go up, the higher the chance that we will find ourselves in the root folder.

If the file showed up. Consider that you have found LFI. In this case, there is no filter at all.

2. Zero byte

The string with the parameter, that is, with the missing ending. But even in the absence of a filter, there can be problems. For example, an ending can be added at the end of a parameter.

But this time, there is an option to fix the line. In older versions of PHP, there was such a flaw as Null Byte Injection. One of them is a null byte assignment. Parameters, when transmitted over http, are encrypted in URL encryption. And in this encoding, the zero byte looks exactly in% 00.

And since in PHP and many other languages, reading the string goes exactly to the zero byte, then we can assign it to the middle of the string so that the next part of the string is discarded. In this case, if we write in the parameter /../../../../../../etc/passwd.php%00,

3. String limit

Another option for dropping the end is possible with String Limit - shortening the string. And what is the use of this? And what if we discard the part of the line with the end, then we get the line we need, but without the end. Already this time, the line can help us /./. I explain what is happening:

More specifically, the two lines are identical in bash. Here's an example of how this can help.

4. php filter

For me, the most interesting option for lfi is lfi with php filter. I give an example right away http://domain/folder/index.php?file=php://filter/convert.base64-encode/resource=index

As a result, the php file will not start in our browser, but its base64 sources will be displayed. This has recently appeared in competitions more and more often.

5. Remote vector

RFI is server-side execution of remote files. Simply put, we have a server that returns the program code with some request, which will be opened and run on the victim server requires several parameters:

1. allow_url_fopen = On
2. allow_url_include = On

6. PHP wrappers

The PHP interpreter has several wrappers that can be used to bypass input filters. The php: // filter wrapper allows the penetration tester to include local files and base64-encode the output stream. Thus, decryption is needed to get readable content. Here base 64 encoding bypass some restriction. We can parse payload and bypass some kind of server security mechanism.

test.php? page = php: //filter/convert.base64-encoded/resources=/etc/passwd

7. Expect: // wrapper

The expect: // wrapper allows system commands to be executed. Unfortunately, this module is not allowed by default.

php? page = expect: // ls

In the example below, the payload is sent to the server via a POST request (using the php: // input construct, execute the ls command).

8. Zip: // wrapper

The zip: // wrapper handles uploaded .zip files on the server side. With this function, a pentester can download a .zip file using the vulnerable download function, and then execute it through the zip unpacker and include it locally. A typical attack looks like this:

If / proc / self / environ gets connected with a flaw in including local files, an attack through the User Agent header is possible. Once the code is injected into the User Agent header, the LFI vulnerability is then exploited to execute / proc / self / environ and restart environment variables, which in turn allows the reverse shell to run.

Blank Byte Injection helps bypass filters by adding an encoded blank byte (for example,% 00) to the URL. Typically, this trick allows you to bypass basic filters by adding blank characters that are valid and are not handled by the back end of the web application.

Practical examples related to injecting an empty byte to include a local file:

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

test.php? page = / etc / passwd% 2500

9. Shortening techniques

Shortening is another technique for bypassing filters. When injecting a long parameter into a mechanism with an LFI vulnerability, the web application can truncate the input parameter, which will help bypass the input filter. You have to bypass such as a mechanism to execute malicious command to the server.

10. Log file injection

The method is based on injecting the source code through other external services into the target system's log file [4]. For example, injecting PHP reverse shell code into a URL will prompt the syslog service to create a 404 (page not found) entry in the Apache web server access log. Then the Apache log can be parsed using the previously discovered LFI vulnerability and run the injected PHP code.

After adding the source code to the log (s) of the target system, the next step is to find the location of these files. After finding the type of target system and web server, you should search the logs using the standard paths used in this particular system and web server. Payloads tailored for the LFI vulnerability in combination with the Burp Intruder application can be used to find the paths where the logs are on the target system.

C. PRIVILEGE ESCALATION

In real based scenario you can chain with other vulnerabilities different attack scenarios lead to privilege escalation issues and increase your attack surface.

Example if an attacker able to gain access by exploiting a directory traversal attack in a web server they can do anything and with chaining with code injection they can upload a shell into a web server and perform a website defacement attack. Nowadays lots of government websites suffered from it.

IV. CONCLUSION

In Morden web application era maintaining security is major concern. Day by day new technologies come and security practices are necessary. It's all about understand the logic and workflow of the loopholes of the particular system and exploit in such a manner. Directory traversal are connected to such a code injection like remote code injection and it will be very convenient to chaining so it could be very serious trouble for the server.

V. REFERENCES

[1]. Michael Flanders. A Simple and Intuitive Algorithm for Preventing Directory Traversal Attacks (August 2019)

[2]. Afsana Begum,Md. Maruf Hassan,Touhid Bhuiyan,Md. Hasan Sharif. RFI and SQLi based local file inclusion vulnerabilities in web applications of Bangladesh (Dec. 2016)

[3]. Douglas Rocha; Diego Kreutz; Rogério Turchetti.A free and extensible tool to detect vulnerabilities in Web systems (June 2012)

[4]. Yunhui Zheng; Xiangyu Zhang. Path sensitive static analysis of web applications for remote code execution vulnerability detection.

[5]. Hannes Holm,Teodor Sommestad,Ulrik Franke,Mathias Ekstedt. "Success Rate of Remote Code Execution Attacks Expert Assessments and Observations."

[6]. Noertjahyana, Agustinus and Gunawan, Ibnu and Tjahjono, Deddie (2012) Website Application Security Scanner Using Local File Inclusion and Remote File Inclusion

[7]. Michal HubczykAdam DomanskiJoanna Domanska . Local and Remote File Inclusion Part of the Advances in Intelligent and Soft Computing book series (AINSC, volume 118)

[8]. Md. Maruf Hassan, Touhid Bhuiyan, Saikat Biswas.An Investigation of Educational Web Applications in Bangladesh: A Case Study on Local File Disclosure Vulnerability. (November 2016)

Cite this article as :

Mrunalsinh Chawda, Dr. Priyanka Sharma "Deep Dive into Directory Traversal and File Inclusion Attacks leads to Privilege Escalation", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 8 Issue 3, pp. 115-120, May-June 2021. Available at doi : https://doi.org/10.32628/IJSRSET218384
Journal URL : https://ijsrset.com/IJSRSET218384