Directory Traversal Vulnerability

https://campus.barracuda.com/doc/42049342/

Description

The Directory Traversal attack (also known as *path traversal* attack or a *dot dot slash* attack) is an HTTP exploit that allows an attacker to access restricted files, directories, and commands that reside outside of the web server's root directory. Directory traversal attacks are executed through web browsers. An attacker might manipulate a URL in such way that the website will reveal the confined files on the web server.

Typically, web servers provide two security mechanisms to restrict user access:

- Access Control Lists (ACLs)
- Web Document Root Directory

The access control list determines which users or groups are privileged to access, modify, or execute files on the web server. Users are restricted from accessing the specific part of the file system on the server, which is known as *root*, *web document root*, or *CGI root* directory. The attacker uses the special character dot-dot-slash (`../`) sequence to escape the web document root, or alternate encodings of the `../` sequence to bypass security filters and access files or directories that reside outside of the root directory.

Some directory traversal attack variations include:

	forward slash character (../)	**backslash character (..\ )**
URL encoded characters	`..%2e%2e%2f`	`..%2e%2e%5c`
Unicode encoding	`..%u2216`	`..%c0%af`
Double encoding	`..%252F`	`..%255C`

These techniques employ special characters such as the dot (`.`) or NULL (`%00`) character to obfuscate directory traversal exploits.

A directory traversal vulnerability can exist either in web servers or web applications. Web applications that fail to validate input parameters (like form parameters and cookie values) are vulnerable to directory traversal attacks.

Effects

If a web server or web application is vulnerable to directory traversal attack, the attacker can exploit the vulnerability to reach the root directory and access restricted files and directories. Attackers can modify critical files such as programs or libraries, download password files, expose source code of the web application, or execute powerful commands on the web server, which can lead to complete compromise of the web server.

Methods

A Directory Traversal Attack can occur when user-supplied input is not properly filtered or sanitized.

The following data must be sanitized properly before being processed:

- URL Parameters
- FORM Parameters (GET and POST parameters)
- Cookies
- HTTP Request Headers

Examples

If the *server* is vulnerable to directory traversal attack

When average website visitors access `http://www.vulnerable.com/index.html` , they are able to view the content in this page. The `index.html` page resides in a web directory in the server where all web pages are stored. Users can navigate and access the pages to which user access privileges have been defined by the administrator. In this example, web users are allowed to access other files that are not in the *web root* directory of the server. Thus, the server is vulnerable to directory traversal attack. If the attacker intends to access the password file on the server, they would send the following request. This would give them access to sensitive information from the server.

```
http://www.vulnerable.com/../../../etc/passwd
```

If the *application* is vulnerable to directory traversal attack

Consider a daily online news website (`www.example.news.com`) that contains different sections and news articles. The user clicks on the first news article, the request goes to `http://www.example.news.com/news=new_page1.html`, where `new_page1.html` content is displayed to the user. In the same way, when the user clicks on the second article, the URL browser displays content from `http://www.example.news.com/news=new_page2.html`.

An attacker might replace `new_page1.html` or `new_page2.html` with `/etc/shadow` (resulting in `http://www.vulnerable.com/news=/etc/shadow`). This might not be successful, because `/etc/shadow` might not reside in the same directory as `new_page1.html`. The attacker might execute different trial and error methods to find the exact way to access `/etc/shadow`. The attacker sends the following request and retrieves the file in the server:

```
http://www.vulnerable.com/news=../../../../etc/shadow
```

With this action, the attacker can steal sensitive user accounts information in the server.

Preventing Attacks

To prevent directory traversal attacks, you must perform proper input validation on all of the entities mentioned in the **Methods** section above. This includes normalizing the input data to account for obfuscations and encodings.

For applications being actively developed, this filtering and validation should be part of the SDLC. Developers and testing teams should be trained to identify and prevent such vulnerabilities.

For legacy and third party code, this might not be an easy thing to enforce. The Barracuda Web Application Firewall dynamically remediates this vector by identifying and blocking directory traversal patterns across all HTTP entities where they can occur.

Tags

OWASP Top 10, PCI-DSS

See Also

CWE 22, OWASP, WASC