DETECTING WEB ATTACKS - 2 BY P3NGU1N:

Detecting Open Redirection Attacks

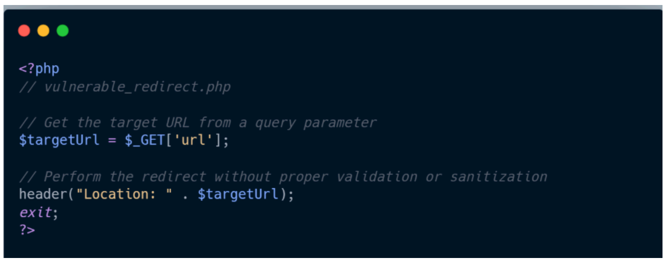
What is open redirection?

- web security vulnerability
- when website redirect user to a different url without proper sanitization of url
- attacker use this to trick user to visit malicious website
- attacker crafts a legitimate url hosted on a vulnerable site but includes a malicious url as parameter or query string
- when user clicks on url, it redirects user to malicious website
- they ocuur when websites use user input such as url as part of their redirection mechanism without proper sanitization

Open Redirection Types / Possible Vectors

1. URL Based Open Redirection
2. Java-Script Based Open Redirection
3. Meta refresh based redirection: when website uses the HTML ″meta refresh″ tag to redirect user to another url automatically
4. Header based open redirection
5. parameter based open redirection

How Open Redirection Works?



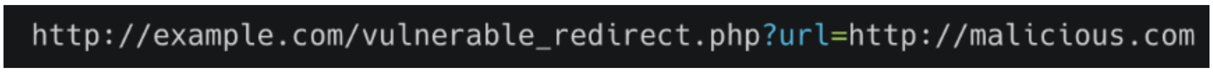
```php
<?php
// vulnerable_redirect.php

// Get the target URL from a query parameter
$targetUrl = $_GET['url'];

// Perform the redirect without proper validation or sanitization
header("Location: " . $targetUrl);
exit;
?>
```

- it is taking a url from user and using it to redirect without proper sanitization

- it can create a url having malicious site to which it is redirecting



```
http://example.com/vulnerable_redirect.php?url=http://malicious.com
```

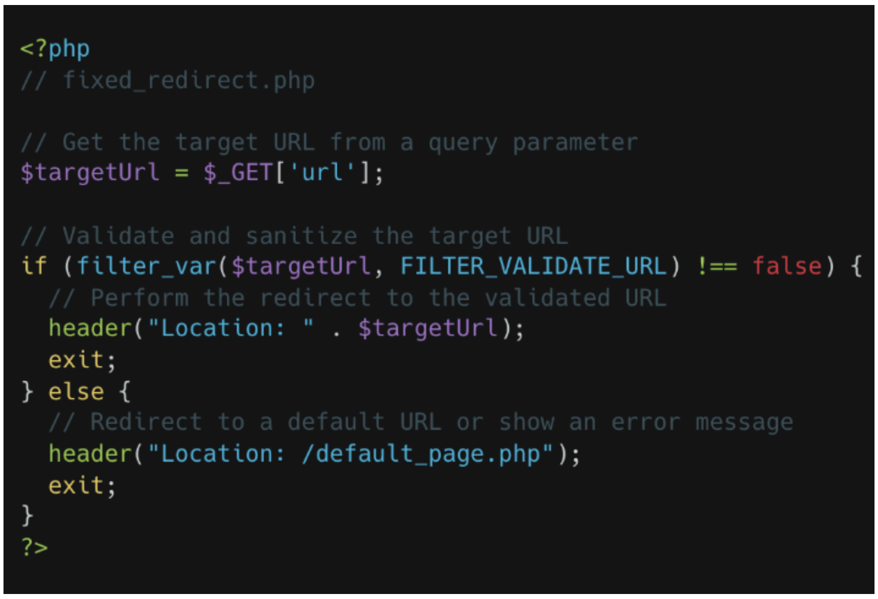
Impact of Open Redirection:

- phishing
- malware distribution
- social engineering attacks
- Reputation Damage
- legal consequences

Prevention Methods for Open Redirection

- Use whitelist approach
- validate and sanitize inputs
- avoid user controlled data in redirects
- proper authentication and authorization
- Implement secure coding practices
- educate users

Fixed Code:



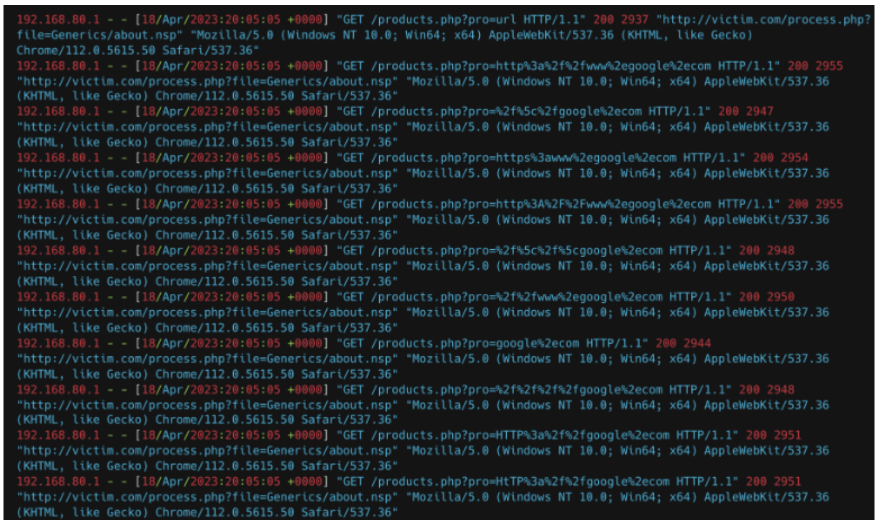
```php
<?php
// fixed_redirect.php

// Get the target URL from a query parameter
$targetUrl = $_GET['url'];

// Validate and sanitize the target URL
if (filter_var($targetUrl, FILTER_VALIDATE_URL) !== false) {
  // Perform the redirect to the validated URL
  header("Location: " . $targetUrl);
  exit;
} else {
  // Redirect to a default URL or show an error message
  header("Location: /default_page.php");
  exit;
}
?>
```

- In the fixed version, the `filter_var` function with `FILTER_VALIDATE_URL` filter is used to validate the user-supplied `url` parameter.
- This filter checks if the value is a valid URL according to the PHP filter extension, and if it returns `true`, the redirect is performed to the validated URL.
- If the `url` parameter does not pass the validation, a default URL or an error message can be shown, and no redirection is performed.

Example:



```
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=url HTTP/1.1" 200 2937 "http://victim.com/process.php?
file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=http%3a%2f%2fwww%2egoogle%2ecom HTTP/1.1" 200 2955
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=%2f%5c%2fgoogle%2ecom HTTP/1.1" 200 2947
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=https%3awww%2egoogle%2ecom HTTP/1.1" 200 2954
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=http%3A%2F%2Fwww%2egoogle%2ecom HTTP/1.1" 200 2955
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=%2f%5c%2f%5cgoogle%2ecom HTTP/1.1" 200 2948
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=%2f%2fwww%2egoogle%2ecom HTTP/1.1" 200 2950
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=google%2ecom HTTP/1.1" 200 2944
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=%2f%2f%2f%2fgoogle%2ecom HTTP/1.1" 200 2948
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=HTTP%3a%2f%2fgoogle%2ecom HTTP/1.1" 200 2951
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
192.168.80.1 - - [18/Apr/2023:20:05:05 +0000] "GET /products.php?pro=HtTP%3a%2f%2fgoogle%2ecom HTTP/1.1" 200 2951
"http://victim.com/process.php?file=Generics/about.nsp" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
```

Here attacker want to redirect to google.com with ?pro parameter

requests are sent by using a tool because multiple requests in a minute

Log location: /root/Desktop/QuestionFiles/Open-Redirection/access.log

What date did the exploitation phase of Open Redirection start? Format: dd/MMM/yyyy HH:mm:ss

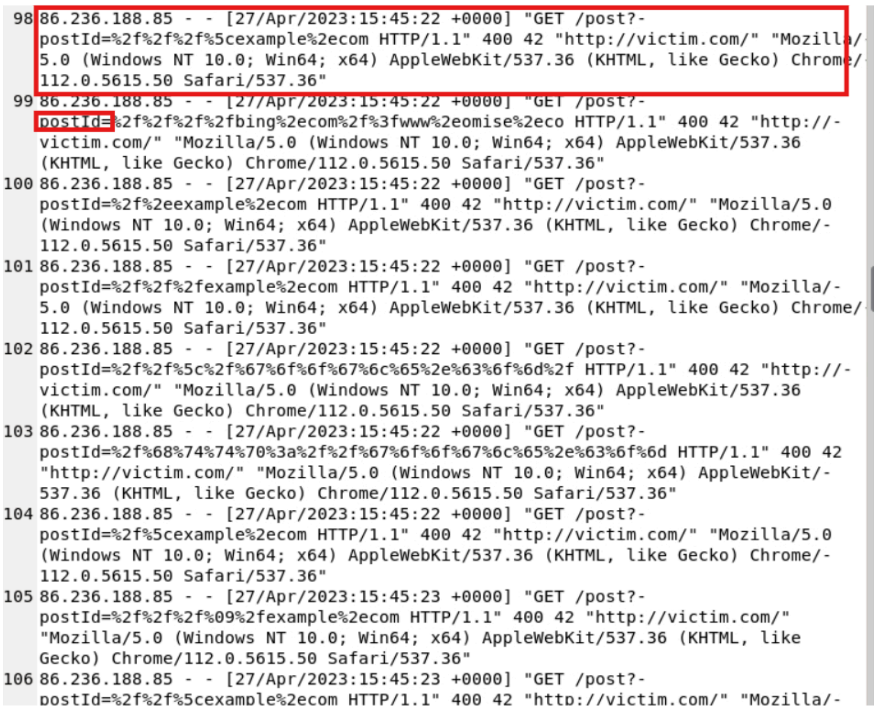
27/Apr/2023 15:45:22

What is the IP address of the attacker who performed the Open Redirect attack?

86.236.188.85

What was the parameter that attacked?

postId



```
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%2f%2f%5cexample%2ecom HTTP/1.1" 400 42 "http://victim.com/" "Mozilla/-
5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/-
112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%2f%2f%2fbing%2ecom%2f%3fwww%2eomise%2eco HTTP/1.1" 400 42 "http://-
victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%2eexample%2ecom HTTP/1.1" 400 42 "http://victim.com/" "Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/-
112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%2f%2fexample%2ecom HTTP/1.1" 400 42 "http://victim.com/" "Mozilla/-
5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/-
112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%2f%5c%2f%67%6f%6f%67%6c%65%2e%63%6f%6d%2f HTTP/1.1" 400 42 "http://-
victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%68%74%74%70%3a%2f%2f%67%6f%6f%67%6c%65%2e%63%6f%6d HTTP/1.1" 400 42
"http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/-
537.36 (KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:22 +0000] "GET /post?-
postId=%2f%5cexample%2ecom HTTP/1.1" 400 42 "http://victim.com/" "Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/-
112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:23 +0000] "GET /post?-
postId=%2f%2f%2f%09%2fexample%2ecom HTTP/1.1" 400 42 "http://victim.com/"
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/112.0.5615.50 Safari/537.36"
86.236.188.85 - - [27/Apr/2023:15:45:23 +0000] "GET /post?-
postId=%2f%2f%5cexample%2ecom HTTP/1.1" 400 42 "http://victim.com/" "Mozilla/-
```

Detecting Directory Traversal Attacks

What is Directory Traversal?

Directory traversal is an attack type that the attackers leverage often to access files and directories that are stored outside the web server's root directory.

also known as the "dot-dot-slash" attack

For example, let's say a web application uses the following URL to display user profile pictures:

http://example.com/profiles/picture.php?name=user1.jpg

could use the following URL to access a file outside of the profiles

directory: http://example.com/profiles/picture.php?name=../../etc/passwd

Similar to LFI?

Directory traversal involves in manipulating the input that is used to access files on a web server

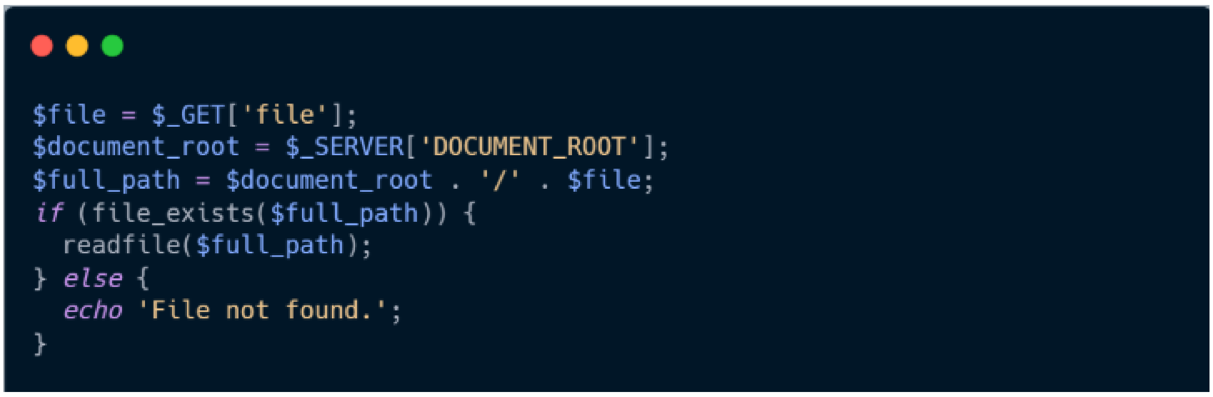
whereas LFI involves in manipulating input that is used to include local files within a web application

Directory Traversal Possible Vectors

Directory traversal attacks can occur through various attack vectors, including:

1. user input
2. cookies
3. HTTP headers
4. File upload
5. Direct requests
6. URL Manipulation
7. malicious links

How Directory Traversal Works?



```php
$file = $_GET['file'];
$document_root = $_SERVER['DOCUMENT_ROOT'];
$full_path = $document_root . '/' . $file;
if (file_exists($full_path)) {
  readfile($full_path);
} else {
  echo 'File not found.';
}
```

code asks user for file name

then connects filename with root directory to form a path

this is a vulnerable code attacker can manipulate the file parameter

```
http://example.com/vulnerable-script.php?file=../../../../etc/passwd
```

this is a example of directory traversal attack, attacker is trying to get the passwd file through traversal

Impact of directory traversal:

1. system compromise
2. denial of service
3. disclosure of sensitive data
4. execution of arbitrary code

Prevention Methods for Directory Traversal Attacks

Input validation and sanitization

ACLs

Relative File Paths

Whitelisting

Secure Coding Practices

WAF Web Application Firewall

Updated code:

```
<?php
$file = $_GET['file'];

// Validate input: only allow alphanumeric characters, underscores, and
hyphens in the file name.
if (!preg_match('/^[a-zA-Z0-9_-]+$/i', $file)) {
  die('Invalid file name.');
}

$document_root = $_SERVER['DOCUMENT_ROOT'];
$full_path = realpath($document_root . '/' . $file);

// Check that the resulting path is within the document root directory.
if (strpos($full_path, $document_root) !== 0) {
  die('Invalid file path.');
}

if (file_exists($full_path)) {
  readfile($full_path);
} else {
  echo 'File not found.';
}
?>
```

- we first validate the input using a regular expression to ensure that the file name only contains alphanumeric characters, underscores, and hyphens.
- We then use the realpath() function to get the absolute path of the file
- check that the resulting path is within the document root directory.
- This prevents the use of directory traversal sequences like ../ to access files outside of the intended directory.
- If the file exists, we read and output its contents; otherwise, we output an error message.

Detecting Directory Traversal Attacks

example payloads for the directory traversal vulnerability

```
http://victim.com/page?parameter=../
http://victim.com/page?parameter=..\/
http://victim.com/page?parameter=%2e%2e%2f
http://victim.com/page?parameter=%252e%252e%252f
```

As a bypass technique, attackers may also use unicode encode characters to bypass WAF or any other product.

```
. = %c0%2e, %e0%40%ae, %c0ae
/ = %c0%af, %e0%80%af, %c0%2f
\ = %c0%5c, %c0%80%5c
```

In that case log will look like

```
64.156.17.84 - - [23/Apr/2023:00:38:55 +0000] "GET /example.php?
file=%c0%ae%c0%ae%c0%af HTTP/1.1" 200 1380 "http://victim.com/" "Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/112.0.5615.50 Safari/537.36"
```

Log location: /root/Desktop/QuestionFiles/Directory-Traversal/access.log

What date did the exploitation phase of Directory Traversal start? Format: dd/MMM/yyyy HH:mm:ss

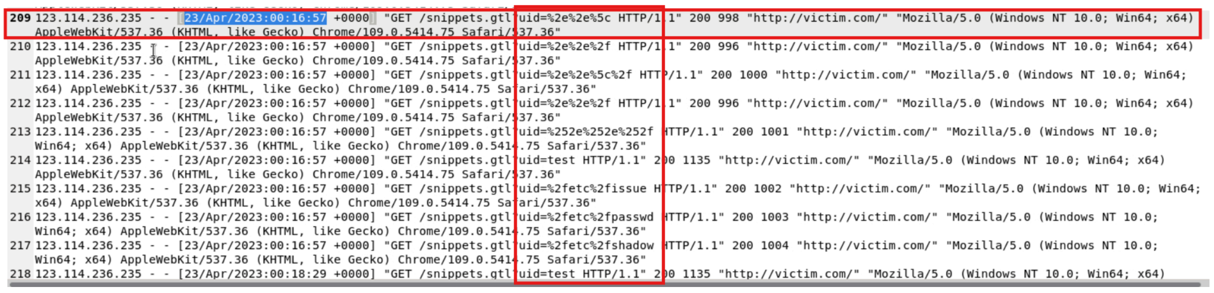
23/Apr/2023 00:16:57

What is the IP address of the attacker who performed the Directory Traversal attack?

123.114.236.235

What was the parameter that attacked?

uid



```
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2e%2e%5c HTTP/1.1" 200 998 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2e%2e%2f HTTP/1.1" 200 996 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2e%2e%5c%2f HTTP/1.1" 200 1000 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2e%2e%2f HTTP/1.1" 200 996 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%252e%252e%252f HTTP/1.1" 200 1001 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=test HTTP/1.1" 200 1135 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2fetc%2fissue HTTP/1.1" 200 1002 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2fetc%2fpasswd HTTP/1.1" 200 1003 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:16:57 +0000] "GET /snippets.gtl?uid=%2fetc%2fshadow HTTP/1.1" 200 1004 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36"
123.114.236.235 - - [23/Apr/2023:00:18:29 +0000] "GET /snippets.gtl?uid=test HTTP/1.1" 200 1135 "http://victim.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
```