New issue

Jump to bottom

Directory traversal vulnerability (CVE-2022-40734) #1150

Open **silicahd** opened this issue on Jun 22 · 5 comments

silicahd commented on Jun 22

I do not use this package, but I seen this in our firewall blocked list. My guess is that there is some sort of vulnerability.

Time	User IP	Type	URL address	Status	Filter	Operating
2022-06-14 23:41:04	18.220.145.184	GET	/laravel-filemanager/download?working_dir=%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2Fetc%2F&type=&file=passwd	Blocked	args	False positive | Details

4

johseg commented on Sep 6

yes that is a security vulnerability. I found that during a pentest for a client and wanted to report this privately, but since the picture shows the attack the cat is out of the back anyway

POC with curl:
curl "https://$DOMAIN/laravel-filemanager/download?working_dir=%2F../../../../../../../../../../../../../../../../etc&type=Files&file=passwd"

This will give you the file /etc/passwd on the server. All files which are readable by the user used to run the application can be read. This usually includes configuration files with passwords etc.

I want to request a CVE for this for proper tracking. Any objections to this by the maintainers? I'll wait a week until I request one.

duyld2108 commented on Sep 10

Just protected lfm routes with middleware [auth] until this issue fixed.

johseg commented on Sep 15

This was assigned CVE-2022-40734.

@silicahd: Can you please change the title to something like "Directory traversal vulnerability (CVE-2022-40734)"?

👍 1

silicahd changed the title ~~**Possible Attack and vulnerability**~~ **Directory traversal vulnerability (CVE-2022-40734)** on Sep 15

xabbuh referenced this issue in Roave/SecurityAdvisories on Sep 19

`Committing generated "composer.json" file as per "2022-09-16T17:14:39…` ··· `b644f94`

attritionorg mentioned this issue on Sep 26

Composer detected a directory traversal vulnerability #1165

Closed

jelmersdp commented on Oct 4 • edited

@duyld2108 is it possible that this is already the case. If I look at the package code I see that the auth middleware is already called.
Are there any other specific lfm route that need to be include in the auth middleware?

👀 1

Jacotheron added a commit to Jacotheron/laravel-filemanager that referenced this issue 8 days ago

`attempt to fix the Issue: Directory traversal vulnerability (CVE-2022…` ··· `3248ab4`

Jacotheron commented 8 days ago • edited

Having spent quite a bunch of time to get familiar with this issue (after I noticed no further releases to fix this issue), I am quite confident that this issue can be marked as resolved, and the releases no longer being marked as insucure.

When using any of the local filesystem drivers (managed by League\Flysystem) will be inspected and get the path normalized (luague/flysystem/src/WhitespacePathNormalizer.php - WhitespacePathNormalizer.php). Inside this class, the method "normalizeRelativePath" will split the path up into its various sections (relative to the root for the laravel disk). This will then rebuild the path to never allow access to anything outside of this root. Should it find that the relative path wants to escape the root, it will throw a PathTraversalDetected exception. This existed for at least 3 years already

From my stacktrace, this happens when LFM determines whether the file exist, prior to fetching it to serve.

The way this is written is quite bulletproof, and other errors will be thrown if it detects possible ways one might try to fool it. I attempted this traversal on my own local server, throwing a bunch of the common directory traversal options (from Directory Traversal Cheat Sheet) at it, and it blocked most, and the others failed as they were so wacky, PHP could not decode them into a path (the file does not exist, thus error 404).

I actually tried to make a pull-request with a bit of basic changes to fix this issue, but was unable to test it to confirm that it works as expected. My strategy was to simply use the PHP "realpath" function, where a relative path can be entered and it normalizes it into an absolute path; then ensure that the laravel disk root is above the normalized real path. For anyone wanting the code:

DownloadController -> getDownload method contents

```
        $file = $this->lfm->setName(request('file'));
         $file_absolute = $file->path('absolute');

         if (!Storage::disk($this->helper->config('disk'))->exists($file->path('storage'))) {
             abort(404);
         }


         if(config('filesystems.disks.'.$this->helper->config('disk').'.driver') === 'local'){
             $disk_root = realpath(config('filesystems.disks.'.$this->helper-
>config('disk').'.root'));
             $file_real_path = realpath($file_absolute);
             if(!Str::startsWith($file_real_path, $disk_root)){
                 abort(404);
             }
         }

         return response()->download($file_absolute);
```

I was unable to get any request to pass the first "abort(404)", while attempting to go outside of the disk root.

As a conclusion, while this might have been an issue in the past, it has been solved (yes, it might not stop bots attempting to break in). While I would like these types of issues to be solved within the package that exposes it, the fact that is fixed by the underlying abstraction layer is good enough for me.

Just an edit: my tests is run with Laravel v9.40.1 (latest at time of writing), and PHP 8.1

💔 2

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

5 participants