Cyber Report

2023

Unmasking the Digital Menace: The CVE-2023-32315 Incident Decoded

The Point of Failure

Date unknown; a vulnerability was first uncovered in the weeks prior to May 23rd, 2023 by user Siebene, a security researcher, who responsibly reported this crucial weak point in Openfire's security to the Openfire community. This point meant users were able to bypass authorisation to the Openfire admin portal using a mechanism known as path traversal, which was a vulnerability in the site at the time. As a mechanism, path traversal works by adding special characters to the URL to bypass security and navigation controls. This can be done by trial and improvement, access to source code which allows discovery of bugs, or fuzzing, which is a large-scale path traversal attempt achieved by using the same command on many websites, and continuing to exploit only those websites that work.This may allow access to protected parts of the system. Although path traversal protections were already in place to guard against this kind of attack, they didn't defend against particular non-standard URL encoding for UTF-16 characters (Unicode) as these characters were not supported by the embedded web server used at the time. This ultimately permitted an unauthenticated user to use the unauthenticated Openfire Admin Console reserved for administrative users.

Here is an example of standard vs non-standard URL encoding:

Standard URL encoding:

space %20

The format is %xx, where xx is a UTF-8 code unit represented as two hexadecimal digits.

Non-standard URL encoding:

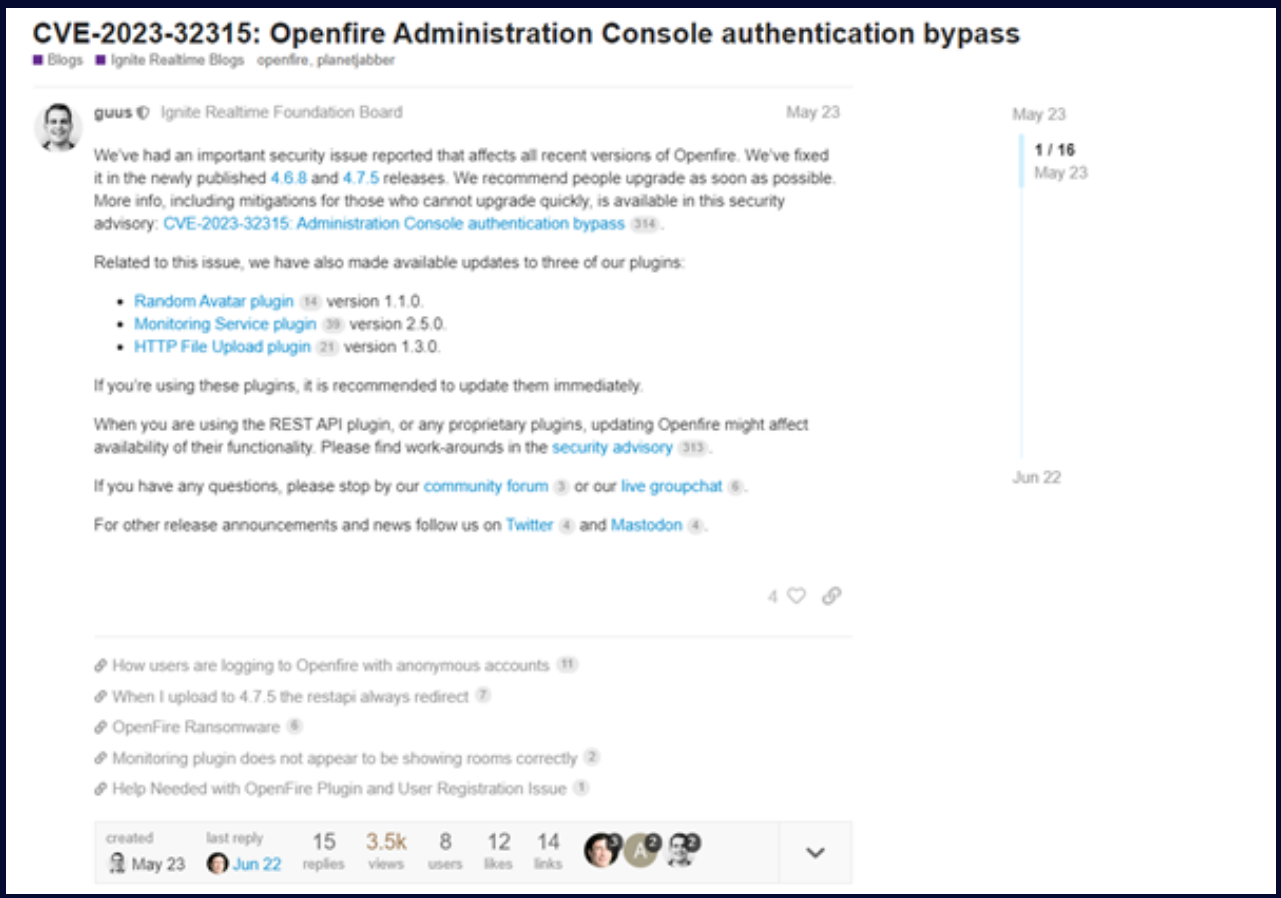
. %u002e

The format is %uxxxx, where xxxx is a UTF-16 code unit representing four hexadecimal digits.

While looking at github user tangxiaofeng7's code for the exploit, which will be discussed more thoroughly later in this report, it is clear that he is using path traversal to perform the exploit as the command '/%%u002e%%u002e/' can be decoded using non-standard URL decoding to form '/../'. This causes the web application to navigate backwards in its internal routing, a vital role in bypassing internal checks.

PATCH N RELEASE N SEC ADV

Shortly after discovering a possible future exploit, a member of the Ignite Realtime Foundation Board, Guus, with the help of cyber solutions company Surevine, created two patched versions of the open-source software Openfire. On May 23rd, Guus made an Igniterealtime forum post going live with the patches (below), summarising the issue and providing a more detailed description of the exploit and patch on a GitHub security tab, encouraging those with the open-source software to update to secure the vulnerability. He also addressed the severity of the issue (8.6/10.0). He stated which versions were vulnerable, as seen in the image below, in addition to a well-assessed evaluation of the exploit and patch.



CVE-2023-32315: Openfire Administration Console authentication bypass

Blogs · Ignite Realtime Blogs · openfire, planetjabber

guus · Ignite Realtime Foundation Board — May 23

We've had an important security issue reported that affects all recent versions of Openfire. We've fixed it in the newly published 4.6.8 and 4.7.5 releases. We recommend people upgrade as soon as possible. More info, including mitigations for those who cannot upgrade quickly, is available in this security advisory: CVE-2023-32315: Administration Console authentication bypass 314.

Related to this issue, we have also made available updates to three of our plugins:

- Random Avatar plugin 14 version 1.1.0.
- Monitoring Service plugin 39 version 2.5.0.
- HTTP File Upload plugin 21 version 1.3.0.

If you're using these plugins, it is recommended to update them immediately.

When you are using the REST API plugin, or any proprietary plugins, updating Openfire might affect availability of their functionality. Please find work-arounds in the security advisory 313.

If you have any questions, please stop by our community forum 3 or our live groupchat 6.

For other release announcements and news follow us on Twitter 4 and Mastodon 4.

4

May 23

1 / 16

May 23

Jun 22

How users are logging to Openfire with anonymous accounts 11

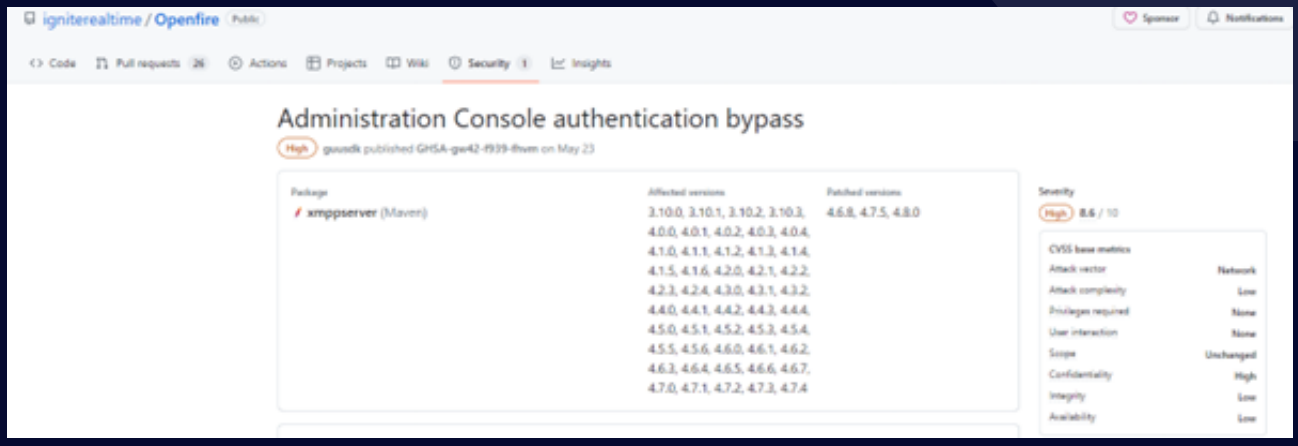
When I upload to 4.7.5 the restapi always redirect 7

OpenFire Ransomware 6

Monitoring plugin does not appear to be showing rooms correctly 2

Help Needed with OpenFire Plugin and User Registration Issue 1

created	last reply	15	3.5k	8	12	14
May 23	Jun 22	replies	views	users	likes	links



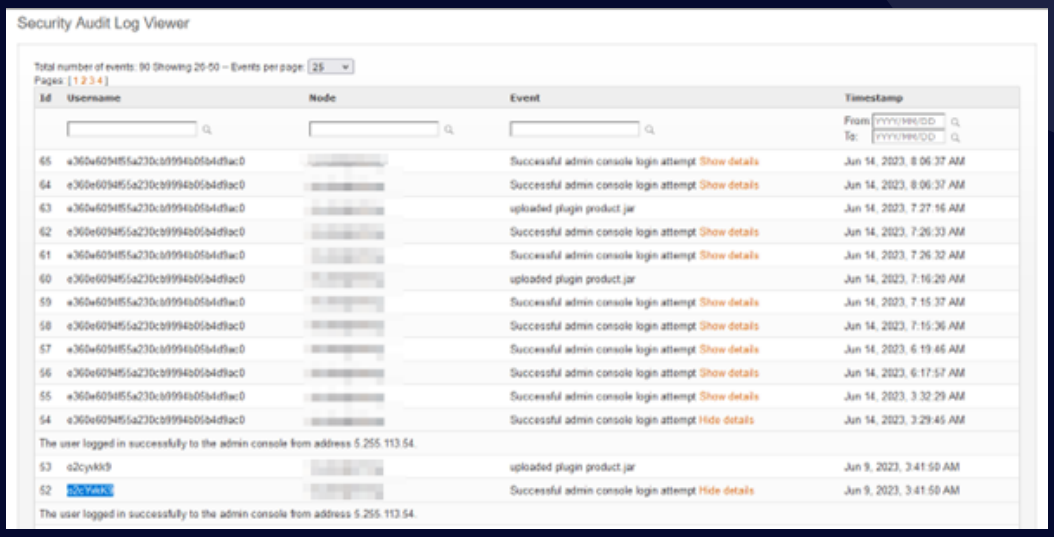
As seen above and simplified below, the patches are on versions 4.6.8 and 4.7.5 and these patches apply to the affected versions below, respectively.

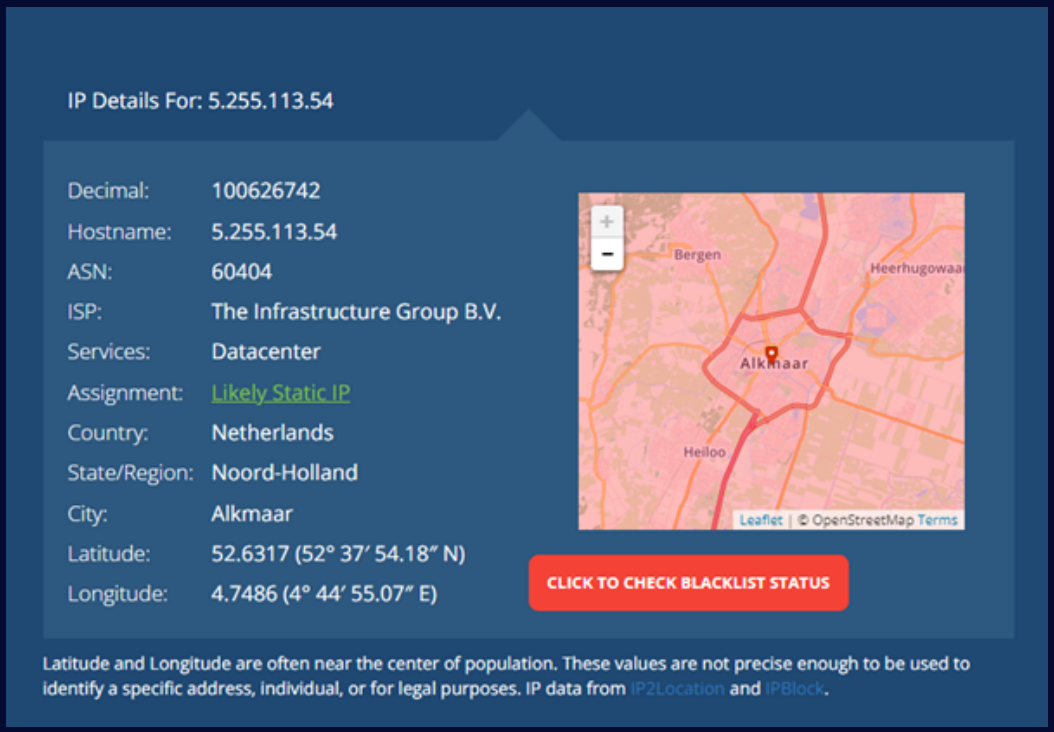
Affected versions	Patched versions
>= 3.10.0, < 4.6.8	4.6.8
>= 4.7.0, < 4.7.5	4.7.5

EXPLOIT IN THE WILD

In the three weeks following Guus' forum post, unbeknownst to the Openfire, Igniterealtime, and Surevine teams, hackers began using Guus' patch notes to their advantage, sparking the development of programs for path traversal into the Openfire admin web portal. This also kicked off the modification of pre-existing example templates for plugins (ironically created by Openfire to add variety in the capabilities of the open source code) into malicious plugins, what we now know are malware loaders, crypto-mining software, and ransomware, as well as website embedded data searchers, able to access companies' SQL databases.

On the 16th of June, the first record of the exploit was reported by user mattv on the Igniterealtime forum, who shared the first screenshot (below) of the exploit being used in the wild a week prior, on the 9th of June; this seemed to be a privately discovered attack as presumably, the method was not publicised, reflected by the fact that this did not lead to a wave of attacks and that there has been no trace of an attack file in the WWW. As expected, when reverse tracing the attacker's IP, it led to a server in a data centre, in a Netherlands location (also below), likely hiding their actual IP.





On top of this, public access websites such as hunter.how can be used by attackers to locate the IP addresses of servers on vulnerable versions, rapidly increasing the attack speed and attack success rate.

From mattv's Openfire logs above, I could observe and deduce the method by which the hackers could attack an Openfire server, which was soon confirmed when viewing the Metasploit and performing the exploit using GCLOUD CLI.

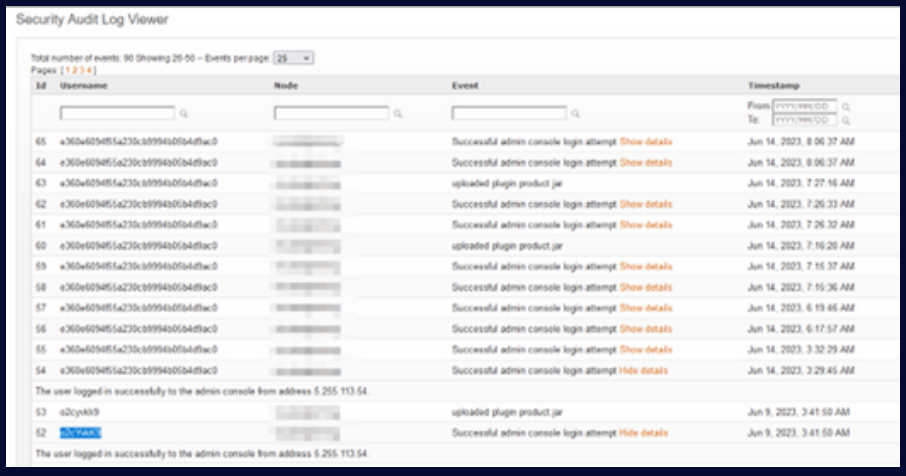
METHOD

The initialisation stage of performing the path traversal using the UTF-16 characters with the non-standard URL encoding allows the hackers to gain access to what seems to be a database/supra-admin facility, where they then perform the following five steps:

1. Add a new admin user
2. Login with this new account
3. Prepare the malicious Openfire plugin with a payload
4. Upload and execute the malicious Openfire plugin with the payload/ use it to gain unauthorised access to databases using an SQL database plugin
5. Finally, remove the plugin and newly added admin user once the destruction has been dealt with to reduce traceability.

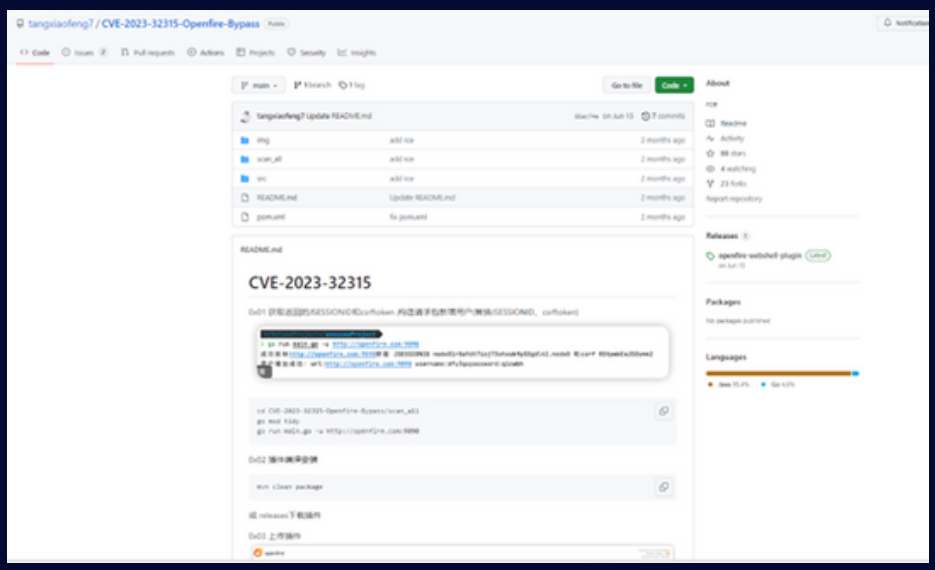
The Metasploit POC (proof of concept), created on the 19th of July as a summary of the attack, confirmed this order below:

```
  def exploit
    # gain access exploiting path traversal vulnerability
    print_status('Grabbing the cookies.')
    fail_with(Failure::NoAccess, 'Authentication bypass is not successful.') unless auth_bypass

    # add a new admin user
    print_status('Adding a new admin user.')
    fail_with(Failure::NoAccess, 'Adding a new admin user is not successful.') if add_admin_user.empty?

    # login with new admin account
    print_status("Logging in with admin user \"#{@admin_login['username']}\" and password
\"#{@admin_login['password']}\".")
    fail_with(Failure::NoAccess, 'Login is not successful.') unless login_admin_user

    # prepare Openfire plugin with payload
    plugin = prepare_plugin_jar
    plugin.add_file("lib/#{rand_text_alphanumeric(8)}.jar", payload.encoded_jar.pack)
    plugin.build_manifest

    # upload and execute Openfire plugin with payload
    print_status("Upload and execute plugin \"#{@plugin_name}\" with payload \"#{datastore['PAYLOAD']}\".")
    fail_with(Failure::PayloadFailed, 'Upload and/or execution of the plugin is not successful.') unless
upload_and_execute_plugin(plugin.pack)

    # cover our tracks!!!
    # remove plugin and newly added admin user
    # Automatic removal of plugin and admin user might cause instability in the application,
    # so remove it manually in Openfire Management console after the exploit is completed.
    print_warning("Plugin \"#{@plugin_name}\" need manually clean-up via Openfire Admin console.")
    print_warning("Admin user \"#{@admin_login['username']}\" need manually clean-up via Openfire Admin
console.")
  end
end
```



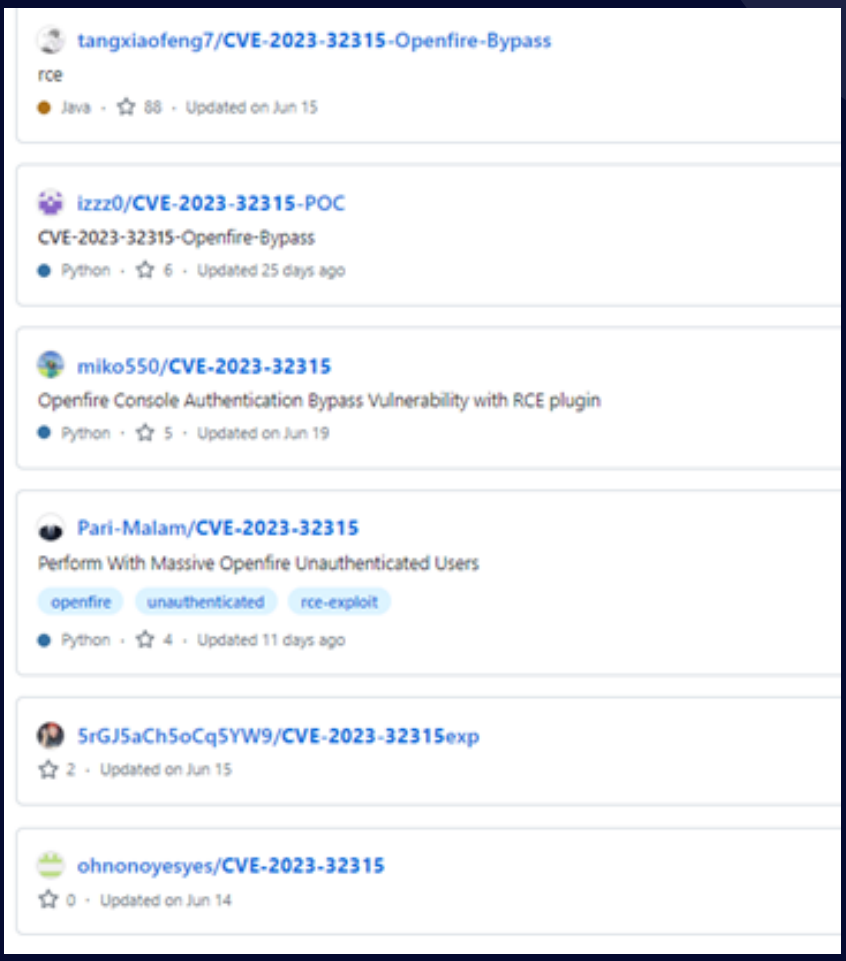
14TH OF JUNE D-DAY

What I found interesting was the date on which the second attack happened, the 14th of June, and the magnitude and frequency of the attack on this Openfire server because, on the 14th of June, a wave of attacks began to hit many Openfire servers at high frequencies and magnitudes for 1-2 weeks, as reflected in the surge of forum posts on these days. This was the same day that GitHub user tangxiaofeng7 uploaded the first fully published version of the exploit into a repository named CVE-2023-32315, containing a complete version of the attack, including a how-to-run instruction description in Chinese (below).



GREY - THE SHADE OF THE UNKNOWN

Tangxiaofeng7 is a pivotal figure in this CVE. They serve a vital role as they are credited in 3/6 of the current repositories, the three most starred repositories of the six related to this CVE, with their original being the highest rated. With a fully-functional exploit code uploaded on the 14th of June and the complete malicious plugin uploaded the next day as well as a comprehensive RCE backdoor allowing complete server access, I think it is safe to say that tangxiaofeng7 has not acted responsibly by releasing this code so soon after the patch, not giving enough time for people to upgrade.

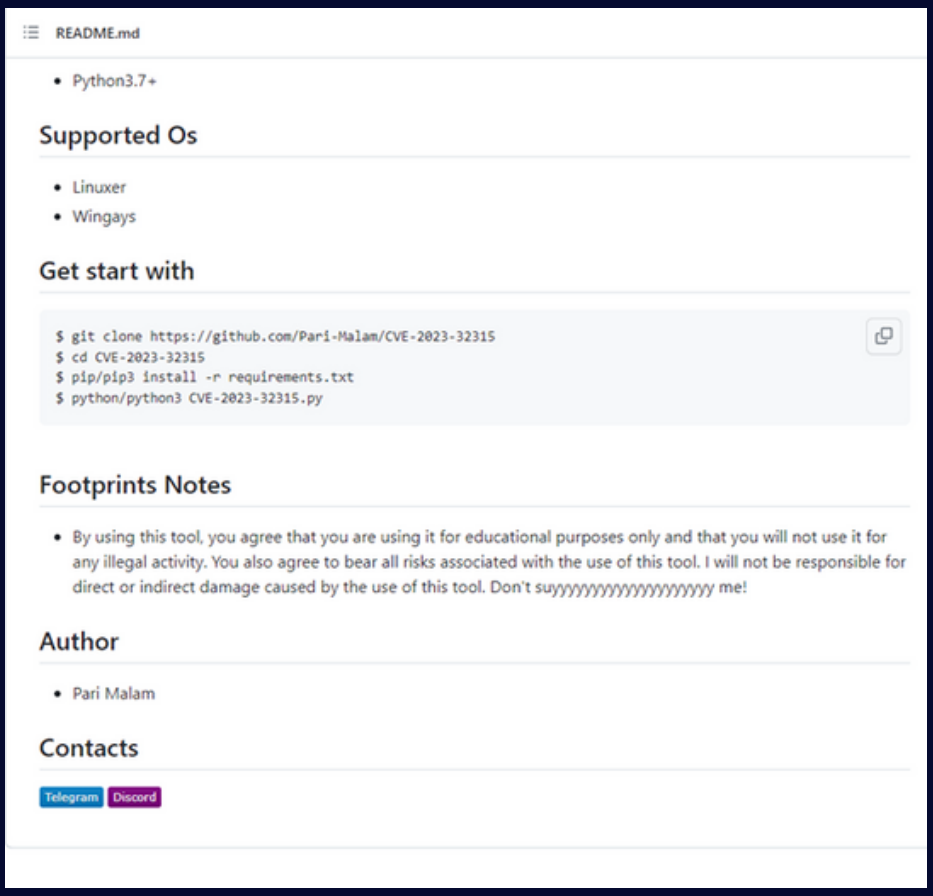


LOCATION

This turned out to be quite an important point as when searching for tangxiaofeng7, I came across a backdoor that implements a Chinese language user interface. It appears to offer the backdoor plugin as a general download on a site which publishes many other malicious payloads. It's hosted in Singapore, on d.serctl.com on a VPS. Only Simplified Chinese is used, and there are no other language options (picture translated below).

When translated, I uncovered the fact that the attacks appeared to come from Chinese language users. Although it could be argued that it is a masking language, the likelihood is minimal as it is common for IPs to be masked but not the language itself. Also, it could be said that this may be a promotion page to display the GitHub link to the Chinese and that this is a whitehat website, just promoting the 'POC'/educational GitHub project; however, the phrase, 'no need to buy' arises suspicion because:

- Usually exploits such as this are commercialised for blackhat usage, hinting towards the fact that the exploit is not valuable so they are giving it for free for black hat usage
- If this POC/Educational use was the case, then a simple 'you agree that you are using it for educational purposes only' would be needed as demonstrated in the GitHub repository for CVE-2023-32315 below by Pari Malam


README.md

- Python3.7+

Supported Os

- Linuxer
- Wingays

Get start with

```
$ git clone https://github.com/Pari-Malam/CVE-2023-32315
$ cd CVE-2023-32315
$ pip/pip3 install -r requirements.txt
$ python/python3 CVE-2023-32315.py
```

Footprints Notes

- By using this tool, you agree that you are using it for educational purposes only and that you will not use it for any illegal activity. You also agree to bear all risks associated with the use of this tool. I will not be responsible for direct or indirect damage caused by the use of this tool. Don't suyyyyyyyyyyyyyyyyyyy me!

Author

- Pari Malam

Contacts

Telegram Discord

- If this was for promotion/exposure, why was it on a non-secure domain?
- Finally, it is clear to see that when downloading one of the files named windows.exe from this website and running it through virus total, it flags it for a virus. This functionality is shown as it appears to open, write, delete and drop files.

These are essential concepts to consider as these are valuable pieces of evidence that allow me to know that this is primarily an attack for Chinese language users to use.

LOOKING AT THE EXPLOIT

After viewing the actual exploit code published on GitHub by tangxiaofeng7, I was able to see the steps of the attack actually in the code itself, e.g. the random username generator in a Java file. I also came to the realisation that the hackers appeared to be rushing to make the exploit (created on the 14th of June) just as fast as the Igniterealtime and Surevine teams rushed to patch the exploit when the vulnerability was discovered prior to the 23rd of May because they had still left in example code from templates instead of refining the folders and files (example shown below).

```
user@burner1:~/Downloads/of2/fproduct$ more readme.html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <title>Readme</title>
</head>
<body>
<h1></h1>
</body>
</html>
```

The HTML file above is a redundant file which seems to have no value nor any attempt to make the plugin look legitimate, therefore emphasising the rush that the attacks were in.

PRIVATE MALWARE SAMPLE

Also, I was provided with a private malware sample by an anonymous forum user. After reverse engineering it using command line functions, I was able to uncover concrete evidence for the date and version that this part of the specific malware sample was created on and targeted, respectfully.

This data confirms the day that the wave of attacks commenced but also shows the carelessness in the hacker's code as they left in a file dedicated to explaining the properties of the files e.g. the date of the download of the template plugin and the version of the target server associated with the plugins.

CLOUD EXPLOIT

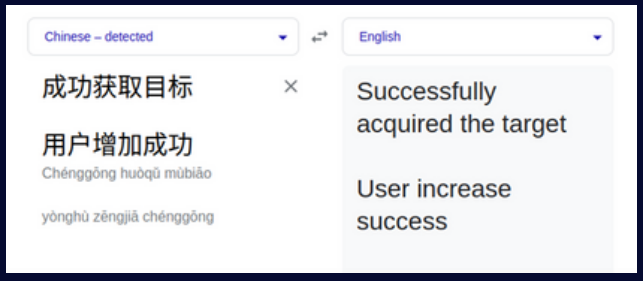
I then performed the exploit provided by tangxiaofeng7 using my own GCLOUD CLI virtual machine as the attacking machine. The victim was a v. 4.7.4 Openfire docker, which I installed from Docker Hub, which I deployed using GCLOUD CLI. I observed that both step 0 and step 1 appeared to be occurring in the background, with no tabs or new windows opening. The only noticeable change was in the users/members area where, after refreshing the page, an admin user had been added (see below for complete execution of the exploit). What was surprising, however, was the rate at which this occurred; when performing the exploit via a Linux terminal, an instantaneous response was generated, detailing the username and password of the new admin account and that the process was successful (an interesting detail explored in the section below). Without any Captcha or 2FA on their website's admin portal, it was clear to me that this, too, was exploitable in terms of a scripting feature, which could mean that the login and actions required to upload a plugin could be automated and, therefore could be used on a much larger scale to install a range of malware, crypto mining software and ransomware or even access company databases using the plugin and many virtual machines.

HACKERS OUTWITTED?

When observing the main.go file from tangxiaofeng7's code, it appeared that he had put in place a mechanism (probably for the fuzzing) in order to authenticate whether the exploit had worked. However, when running Openfire docker v. 4.7.5 and running the terminal, even though the exploit was unsuccessful, due to the new path traversal patch in the update, the terminal returned this:

```
user@hacking-around:~/CVE-2023-32315-Openfire-Bypass/scan_all$ ./exploit -u http://10.128.0.4:9090
成功获取目标http://10.128.0.4:9090 JSESSIONID: node01lg40senhnuta154kjdnzxy8l04.node0 +csrf: fiAZGAfCcBlU7BU
用户增加成功：url:http://10.128.0.4:9090 username:0n6388 password:ij0lef
```

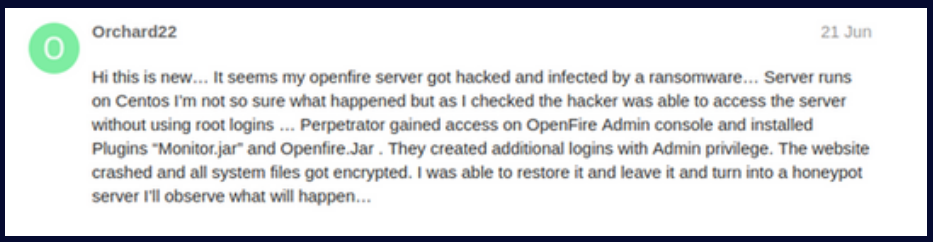
After translation of the two Chinese phrases on the left hand side, it left me with the following messages:



From this I understood that Openfire had presumably put up a defence mechanism with their new update that pretends as if the hack has worked when it hasn't, vastly increasing the number of wasted attempts at logins after unsuccessful 'successful' exploits; this decreases the rate of successful attacks by hackers, and therefore slows them down.

MALWARE

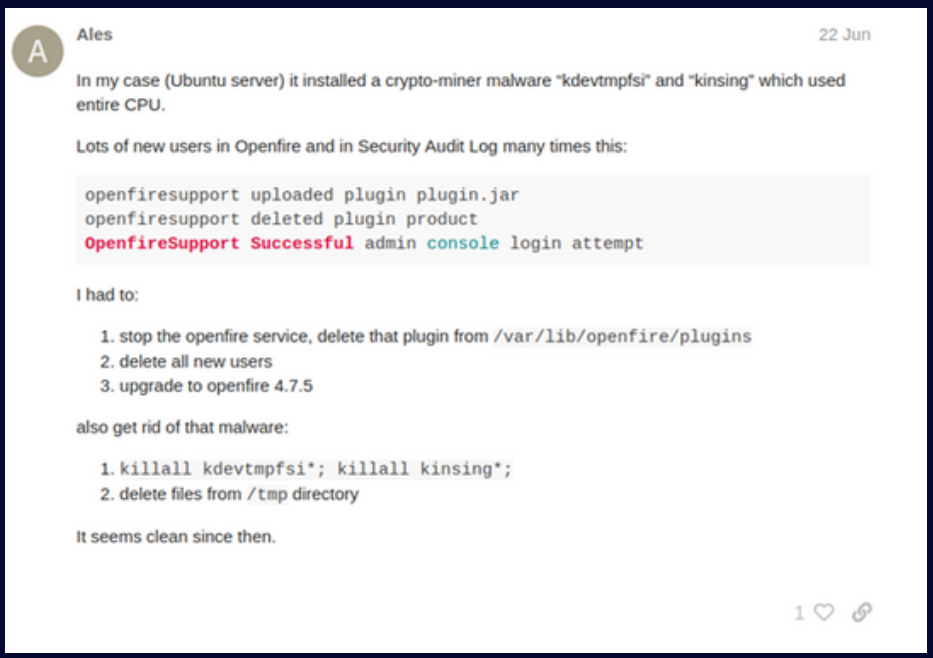
A malicious plugin could steal data and use it for commercial purposes by selling it or holding it and making a company pay a ransom for lucrative income as shown below in an igniterealtime forum post by Orchard22



Orchard22 21 Jun

Hi this is new... It seems my openfire server got hacked and infected by a ransomware... Server runs on Centos I'm not so sure what happened but as I checked the hacker was able to access the server without using root logins ... Perpetrator gained access on OpenFire Admin console and installed Plugins "Monitor.jar" and Openfire.Jar . They created additional logins with Admin privilege. The website crashed and all system files got encrypted. I was able to restore it and leave it and turn into a honeypot server I'll observe what will happen...

A malicious plugin could also allow an attacker to host software for malicious purposes such as a bot net or install crypto miners in order to extract cryptocurrency (and damaging hardware as a by-product) as shown below in an igniterealtime forum post by Ales:



Ales 22 Jun

In my case (Ubuntu server) it installed a crypto-miner malware "kdevtmpfsi" and "kinsing" which used entire CPU.

Lots of new users in Openfire and in Security Audit Log many times this:

```
openfiresupport uploaded plugin plugin.jar
openfiresupport deleted plugin product
OpenfireSupport Successful admin console login attempt
```

I had to:

1. stop the openfire service, delete that plugin from `/var/lib/openfire/plugins`
2. delete all new users
3. upgrade to openfire 4.7.5

also get rid of that malware:

1. `killall kdevtmpfsi*; killall kinsing*;`
2. delete files from `/tmp` directory

It seems clean since then.

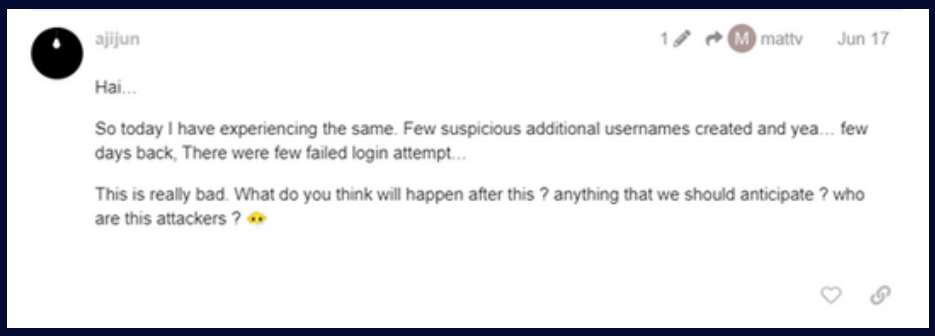
1

SQL

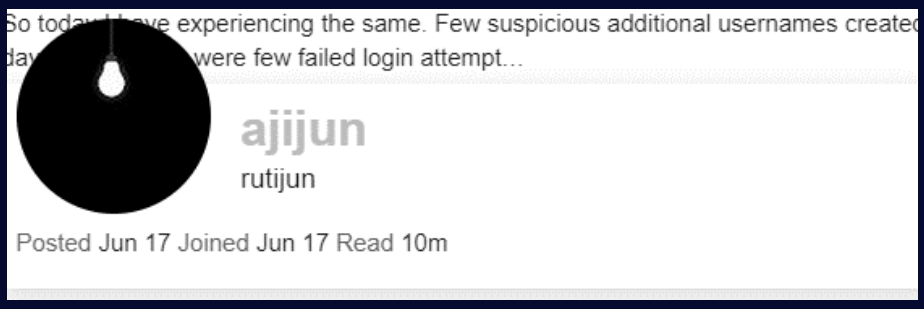
As seen in the video above the plugin allows the option to use SQL commands and access an organisation's database. The fact that this is achievable without authorisation means that it breaks the GDPR as it allows this data to be accessed, modified, deleted, and transferred without permission, therefore leading to commercialised data.

IMPOSTERS

You might think that posts asking for help should always be responded to with the utmost importance and immediate remedy, but this cannot be further from the truth. Whilst reading the forum posts, there appeared to be a suspicious post from the user ajijun asking for help whilst not giving many details, putting ellipses frequently, as well as filler phrases such as 'and yea', creating a vague idea of the issue as shown here:



Feeling that something was not quite right, I viewed his profile, and low and behold:



The fact that the forum had only been read for 10 minutes and that the post was created on the same day as the account gives the impression that, although I cannot say it with certainty, ajijun is one of the attackers/someone considering deploying the exploit and they are probing to see what the Openfire, Igniterealtime, Surevine teams and the Openfire community know. This creates a dilemma for the support team whether to give information to help one in need or to delay this process by asking for unique proof of an issue before engaging in solving the problem itself to prevent fueling the attackers with information on the attack and how much the support teams know.