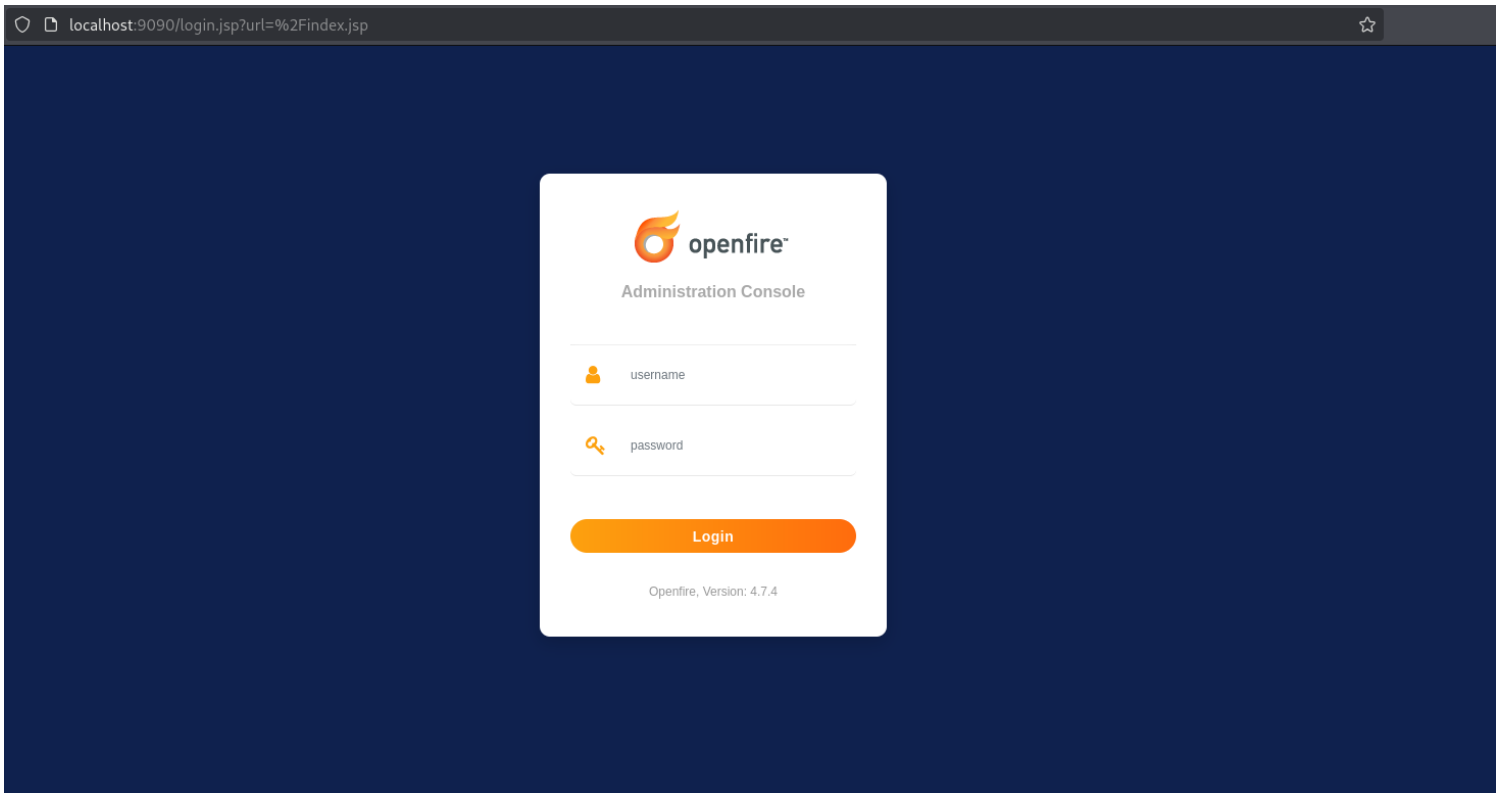
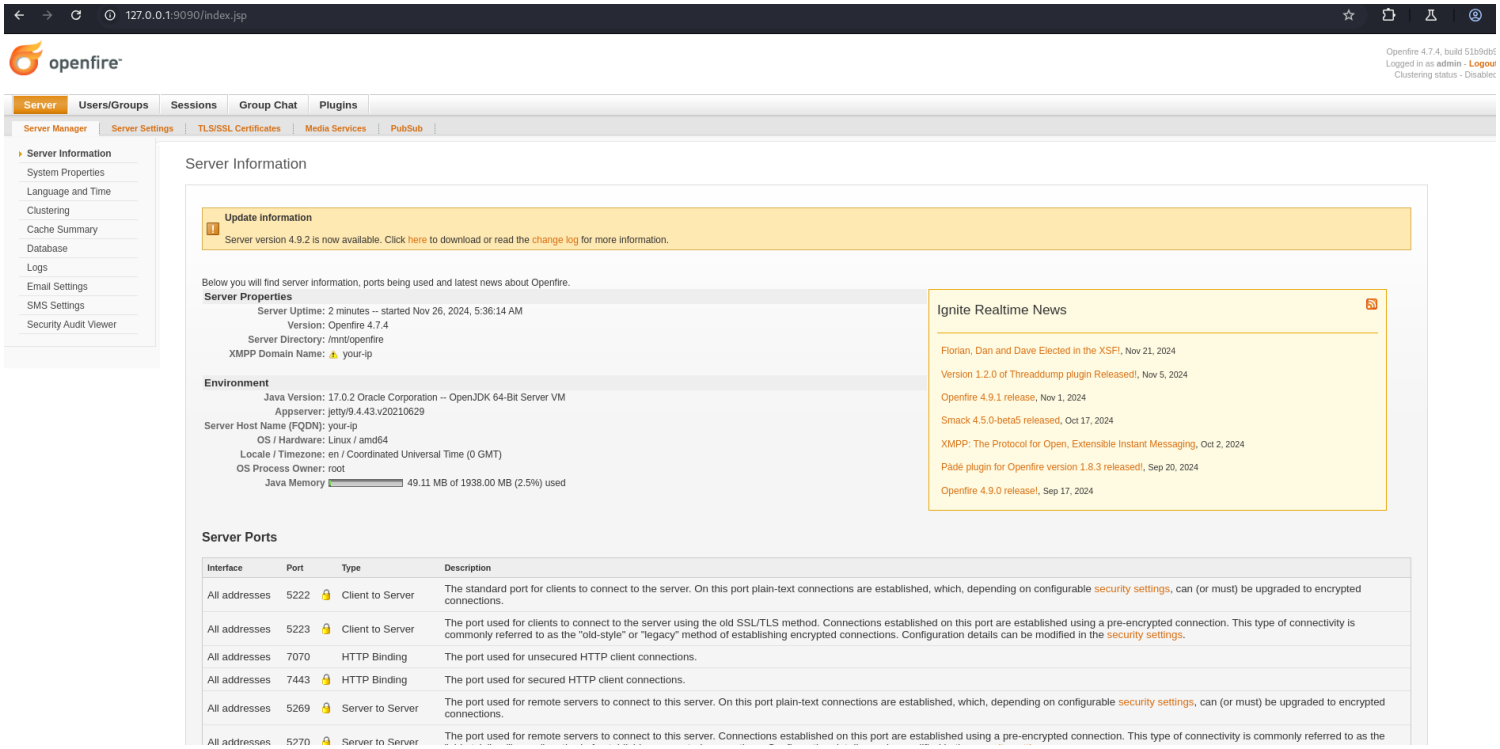
OPENFIRE_EXPLOITATION

#INTRODUCTION

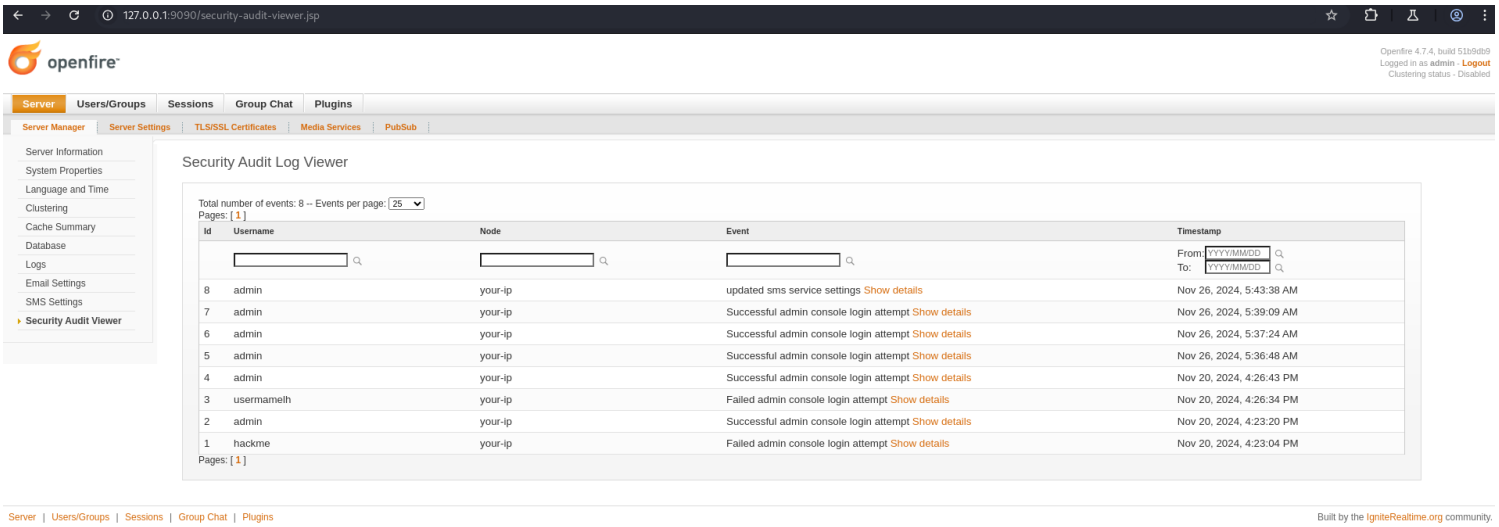
CVE-2023-32315 is a path traversal vulnerability affecting the Openfire admin console. Openfire is a well-known open-source chat server, and according to the current maintainers, Ignite Realtime, the server software has been downloaded almost 9 million times.



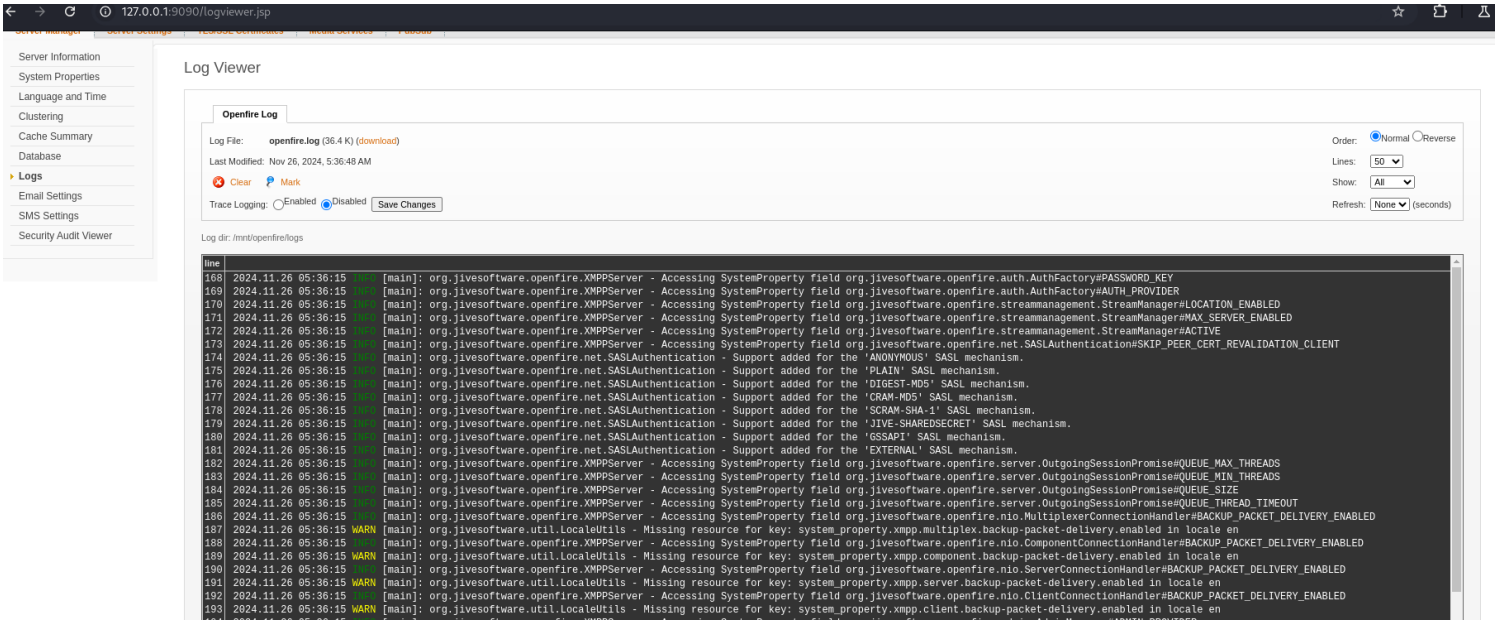
If its the first time to run this software, the default creds is;
admin:admin



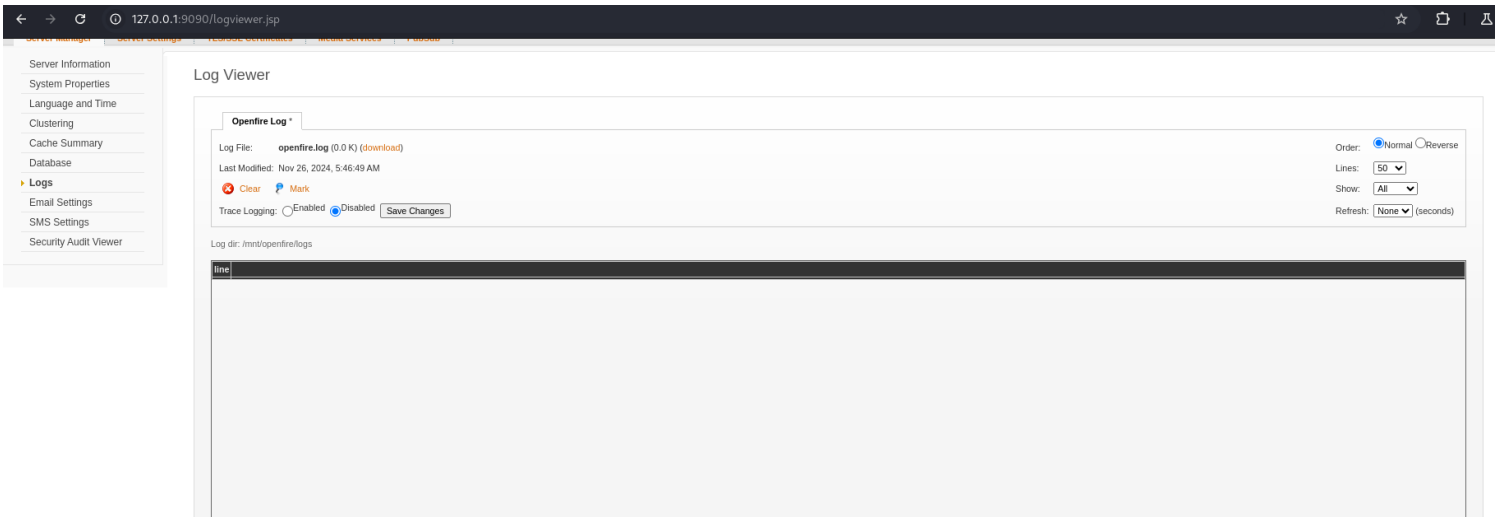
Any activity especially logins, updates and clearance of logs, and logged under security audit viewer which by no means can be cleared, hence leaving an attackers track on the system.



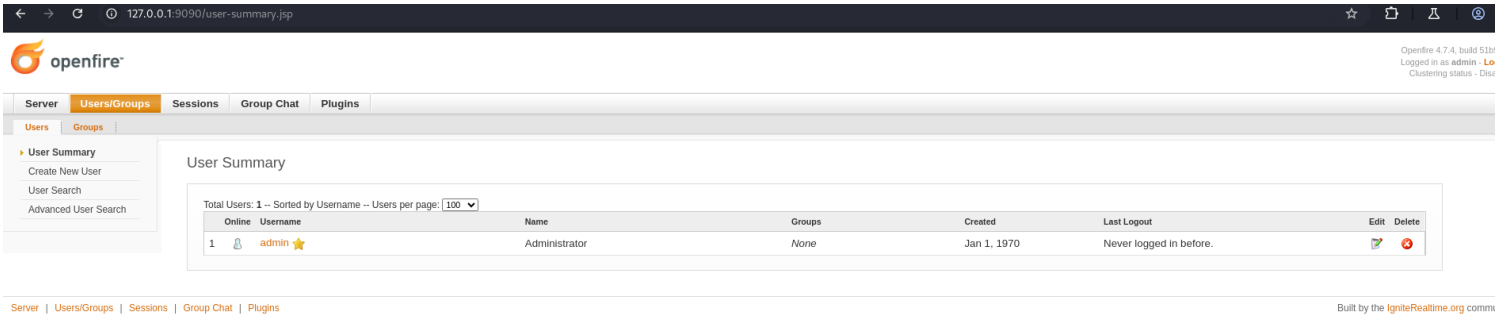
Under the logs, it entails every activity done including upload of plugins and so forth.

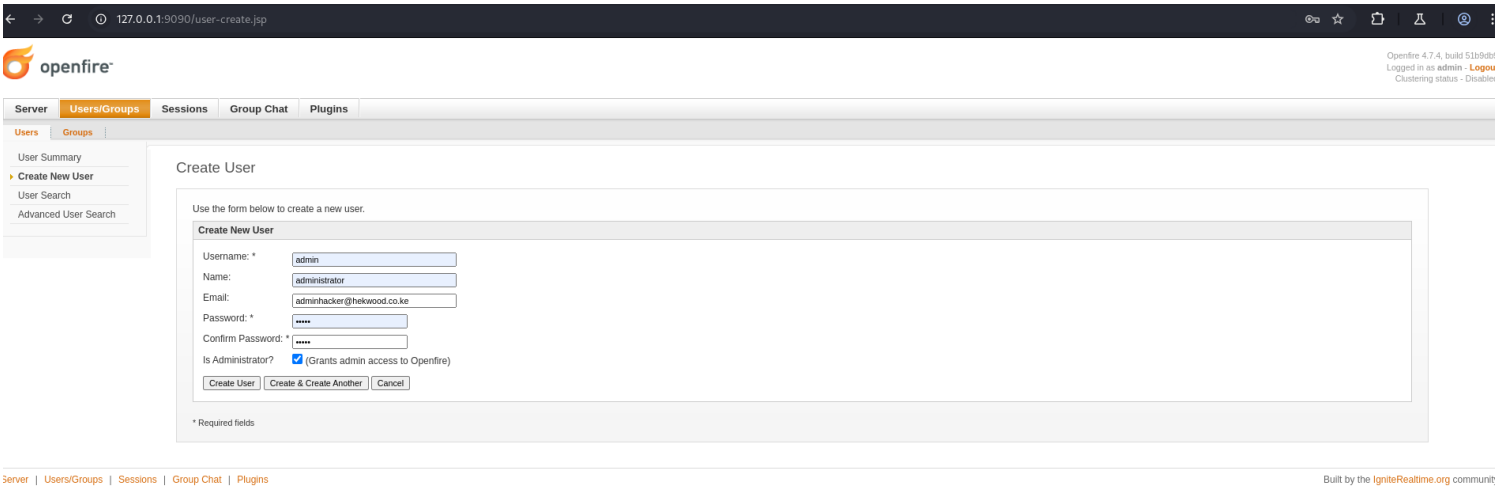


Unfortunately for blue-teamers, the logs, can be cleaned making it difficult to track the breach, though do not forget that the Security Audit Viewer cannot be deleted or cleared.

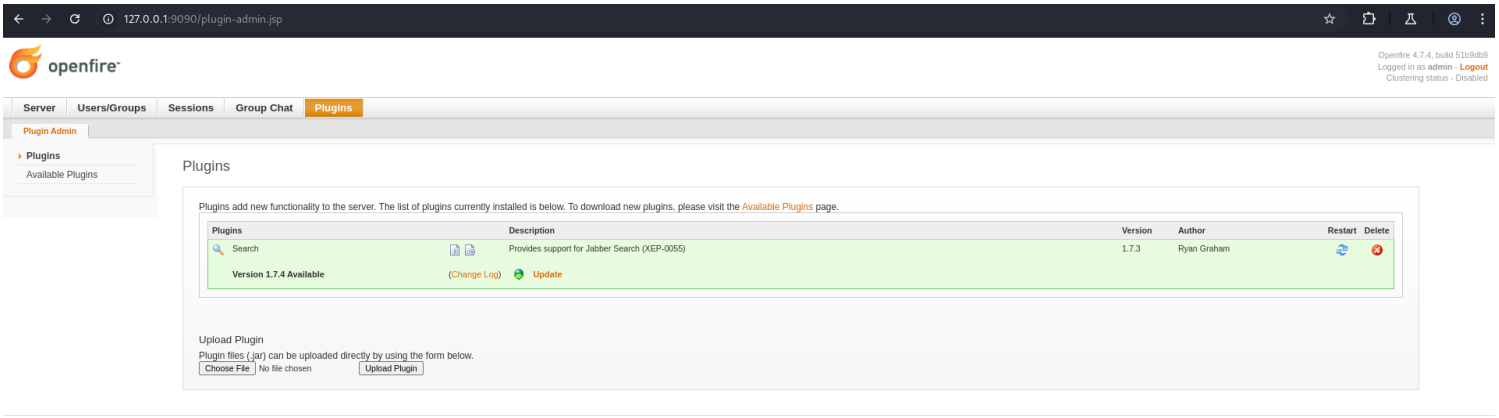


As you can see, There is only one user; admin with administrative privileges.





Here, this software allows upload of plugins specifically .jar plugins with .yml file to define the metadata.

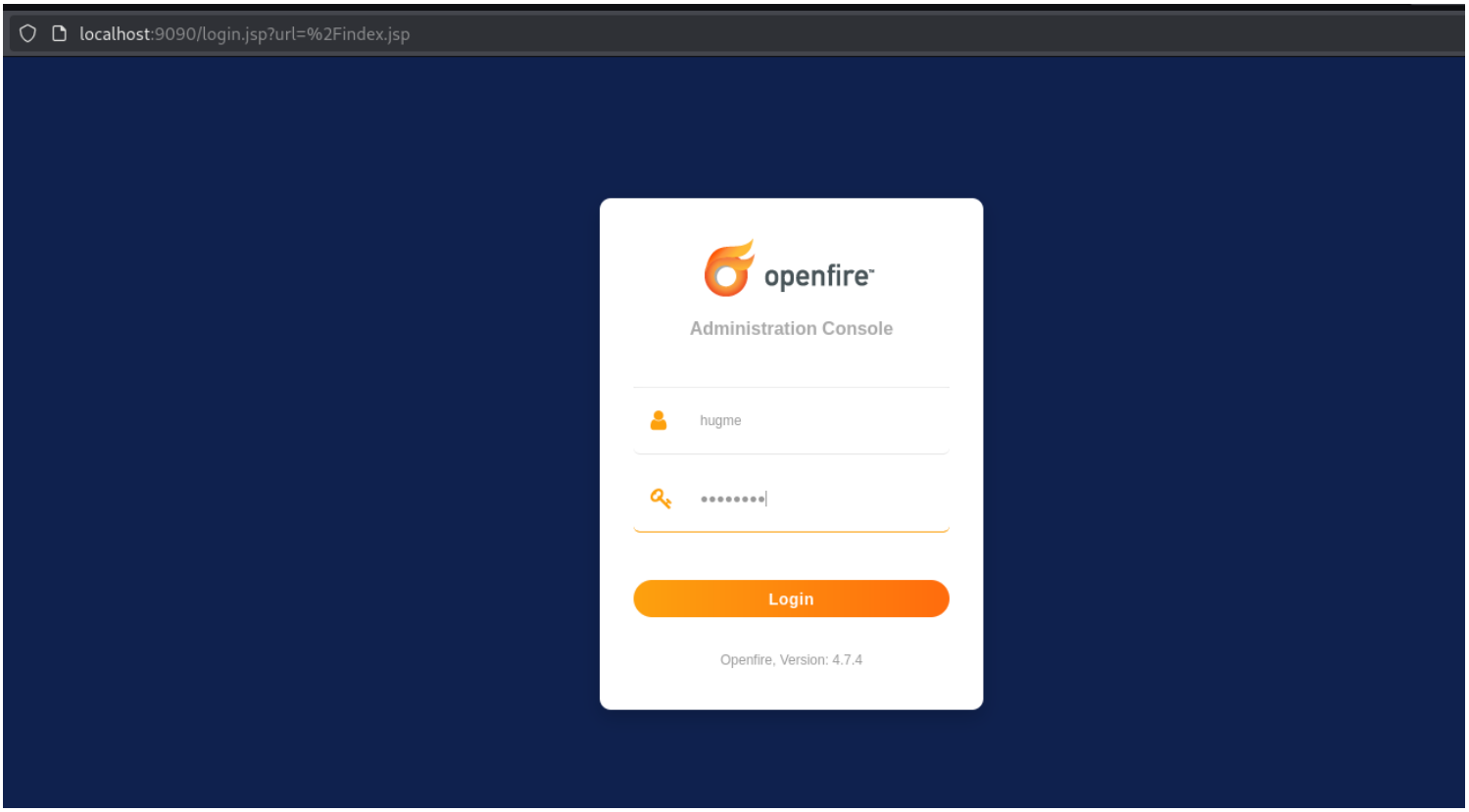


FIRST AUTH_BYPASS

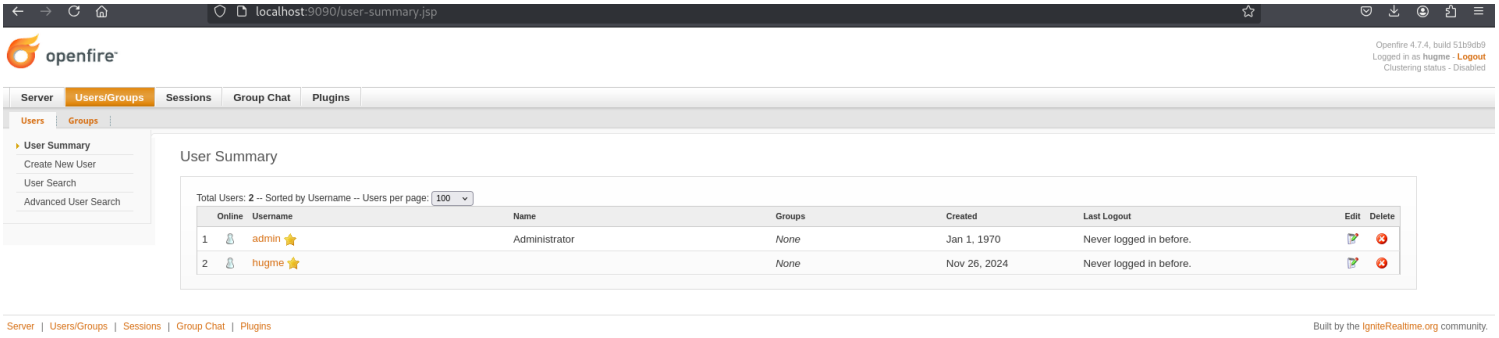
These exploit is creating an admin user to gain access to the Openfire Plugins interface. The plugin system allows administrators to add, more or less, arbitrary functionality to Openfire via uploaded Java JARs, which when accessed by unauthorized malicious user, can lead to RCE and even creation of backdoor in the system.

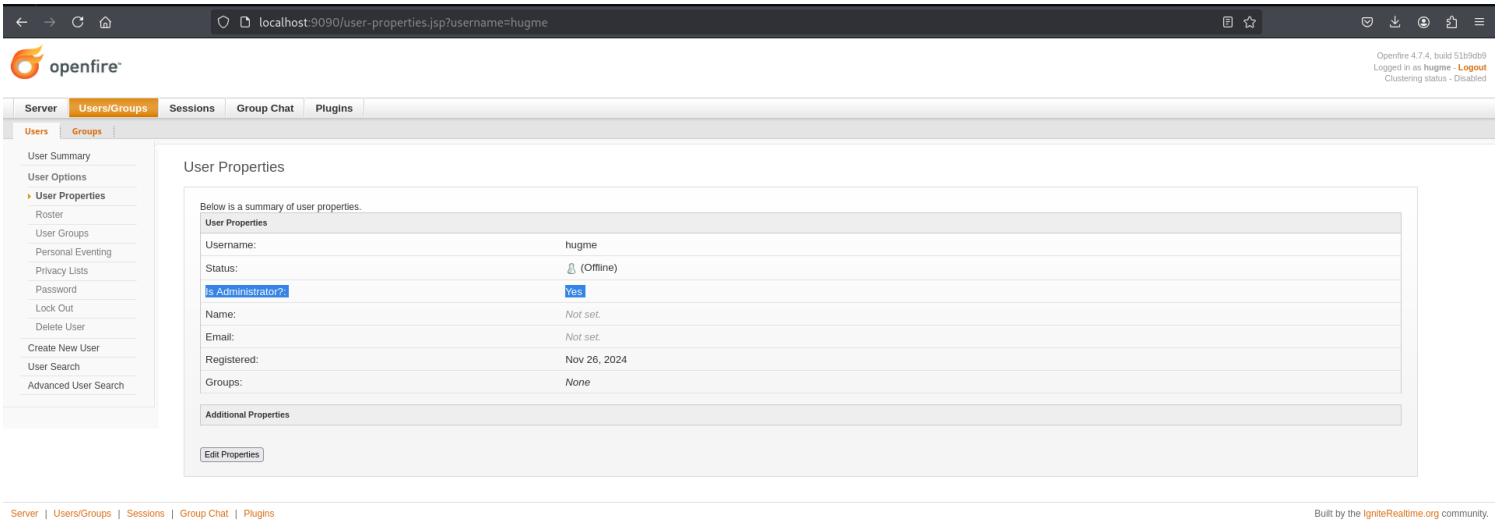
So here this python script exploited this vulnerability and created me a user called hugme:HugmeNow.

Using this creds, I was able to access the web console.



As seen below, the newly created user has admin privileges, which can lead to further damage and breach if he/she has malicious intent.

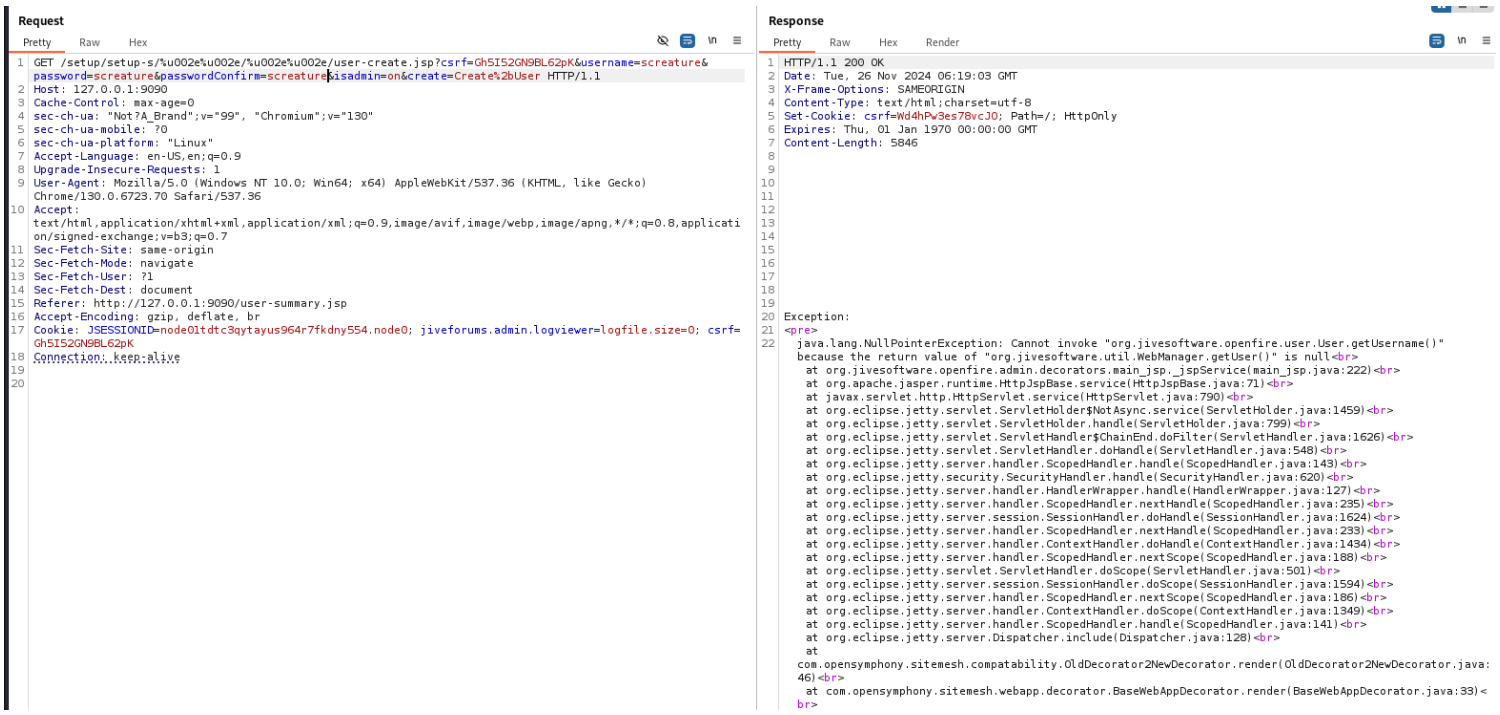


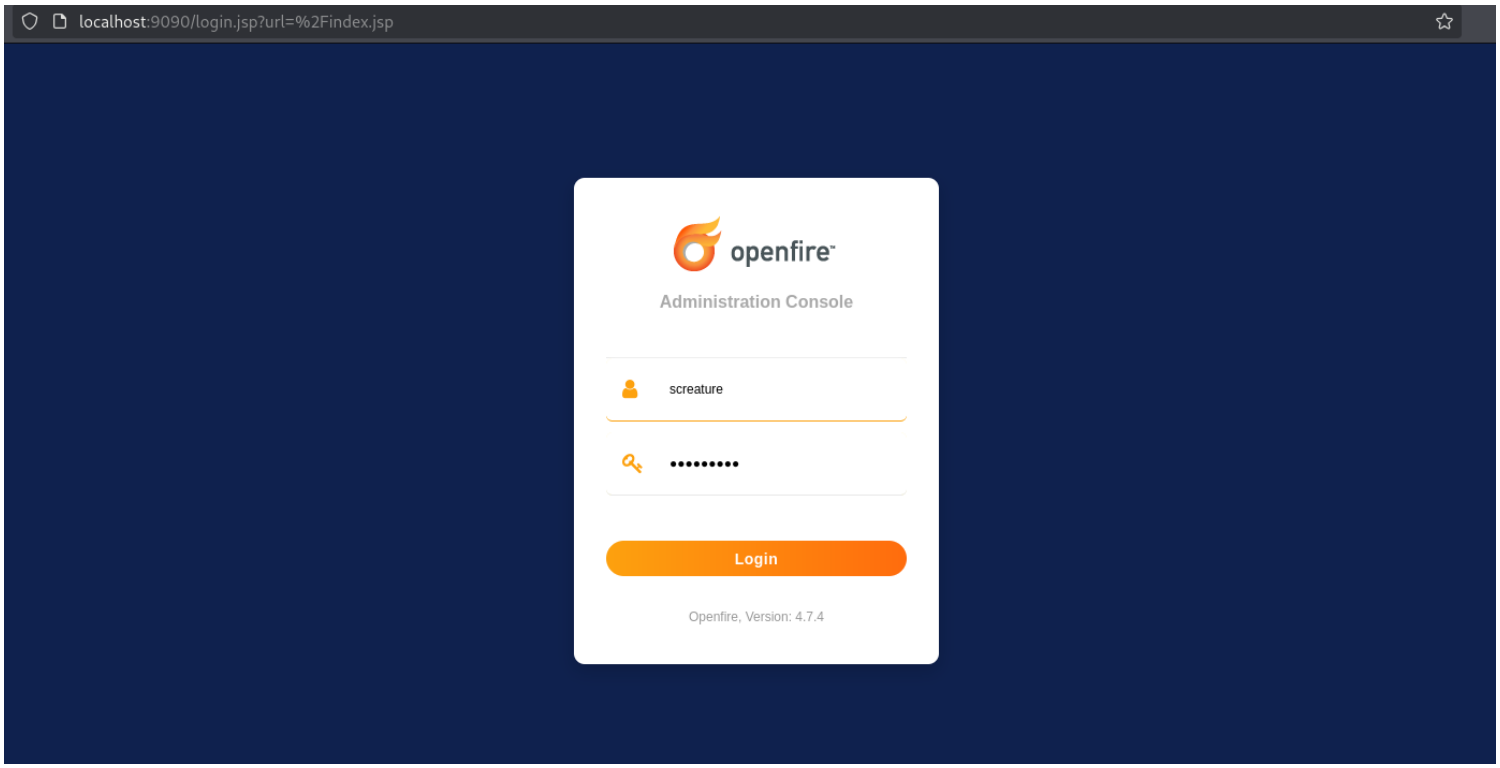


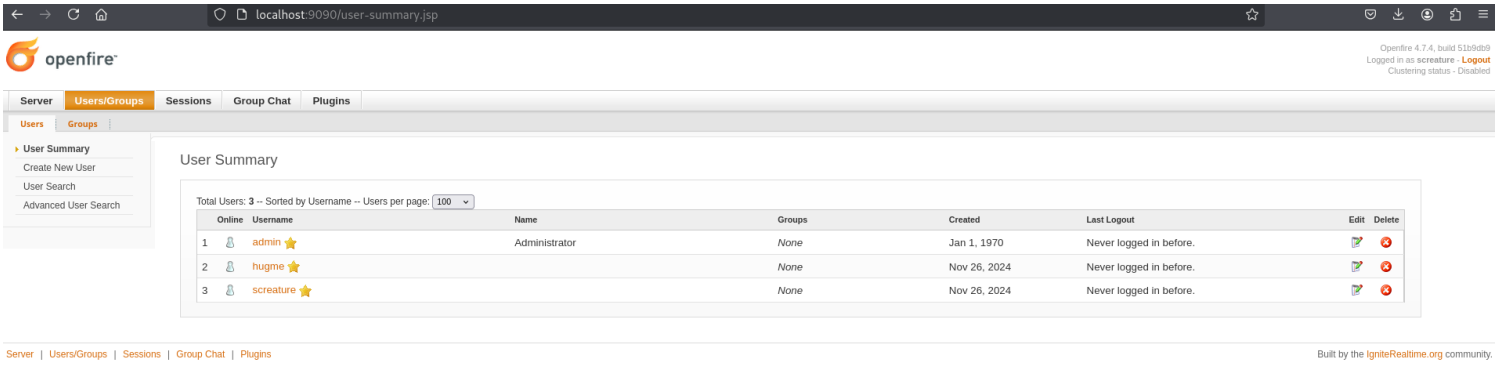
SECOND AUTH_BYPASS

The /setup/setup-s dir environment is used by attackers to create a valid system user without even authenticating to the machine.

As seen below, I created a valid user called "screature:screature" and granted him administrative privileges.

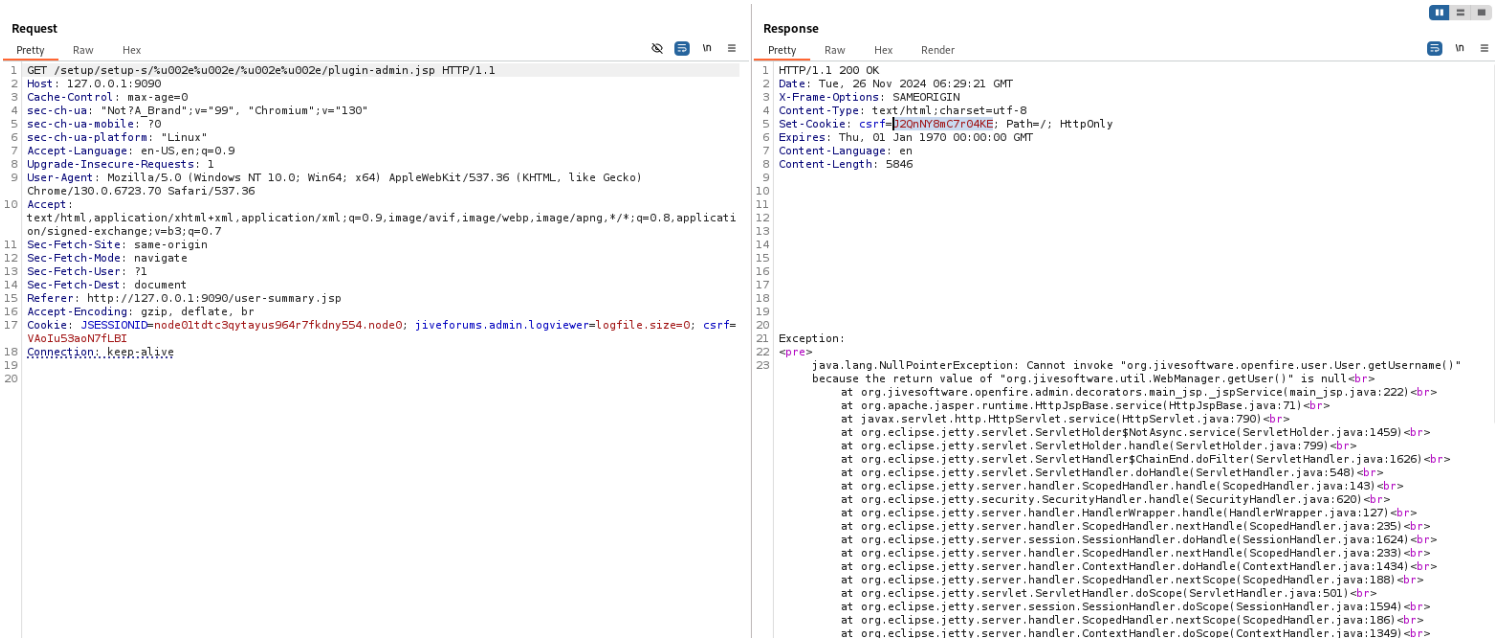




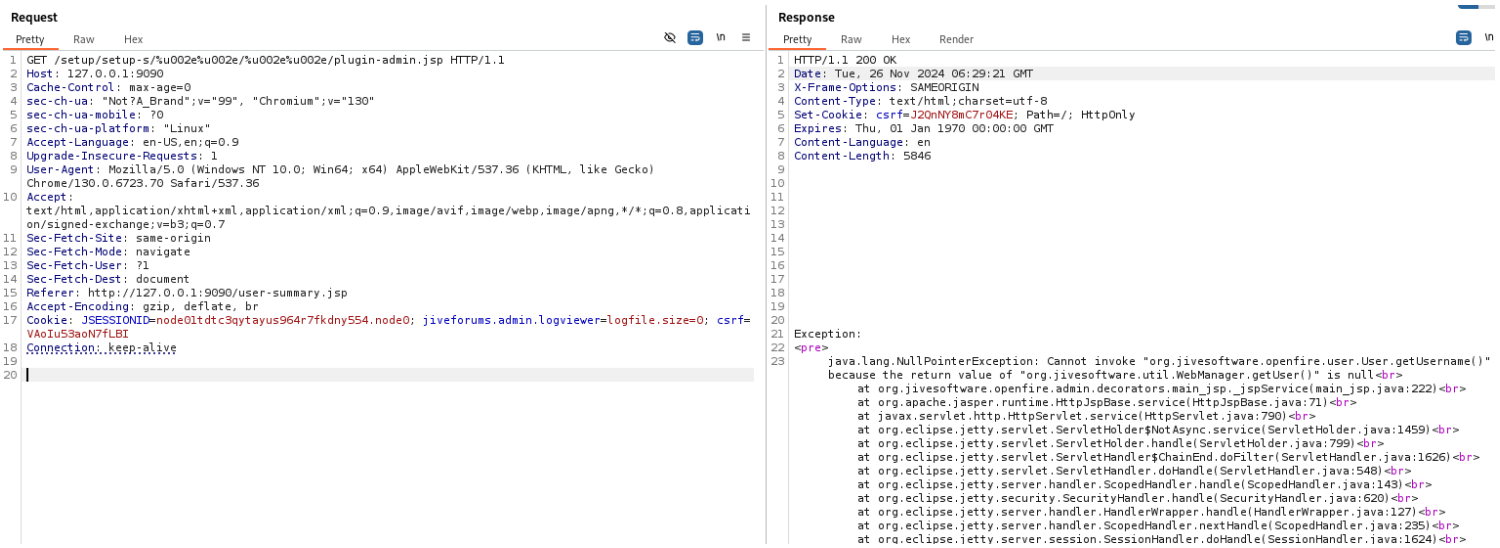


USER-LESS_AUTH_BYPASS TO FILEUPLOAD

What's particularly interesting about this is that creating the administrative user isn't necessary. Unfortunately for defenders, attackers don't need to create a user or authenticate to upload a plugin. CVE-2023-32315 gives the attacker access to `plugin-admin.jsp`, just as it gives the attacker access to `user-create.jsp`.



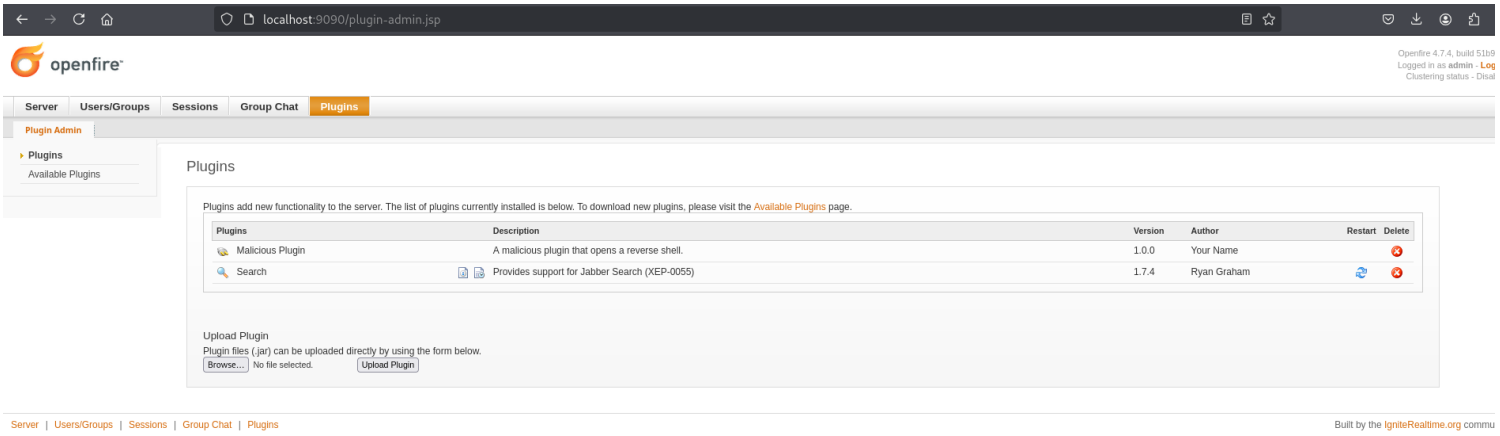
I extracted a `JSESSIONID` and CSRF token from `/setup/setup-s/%u002e%u002e/%u002e%u002e/plugin-admin.jsp` and then executed



by uploading my malicious plugin without even authenticating.

```
┌──(scr34tur3㉿Kali)-[~/Documents/programming/JAVA/malicious_plugin]
└─$ curl -X POST \
  -F "uploadfile=@malicious_plugin.jar" \
  -H "Cookie: JSESSIONID=node0a1q2292y93dnqravl9jyaic24.node0; csrf=DcmTW8t1iZHcrLj" \
  -H "Content-Type: multipart/form-data" \
  "http://localhost:9090/setup/setup-s/%u002e%u002e/%u002e%u002e/plugin-admin.jsp?uploadplugin&csrf=DcmTW8t1iZHcrLj"
```

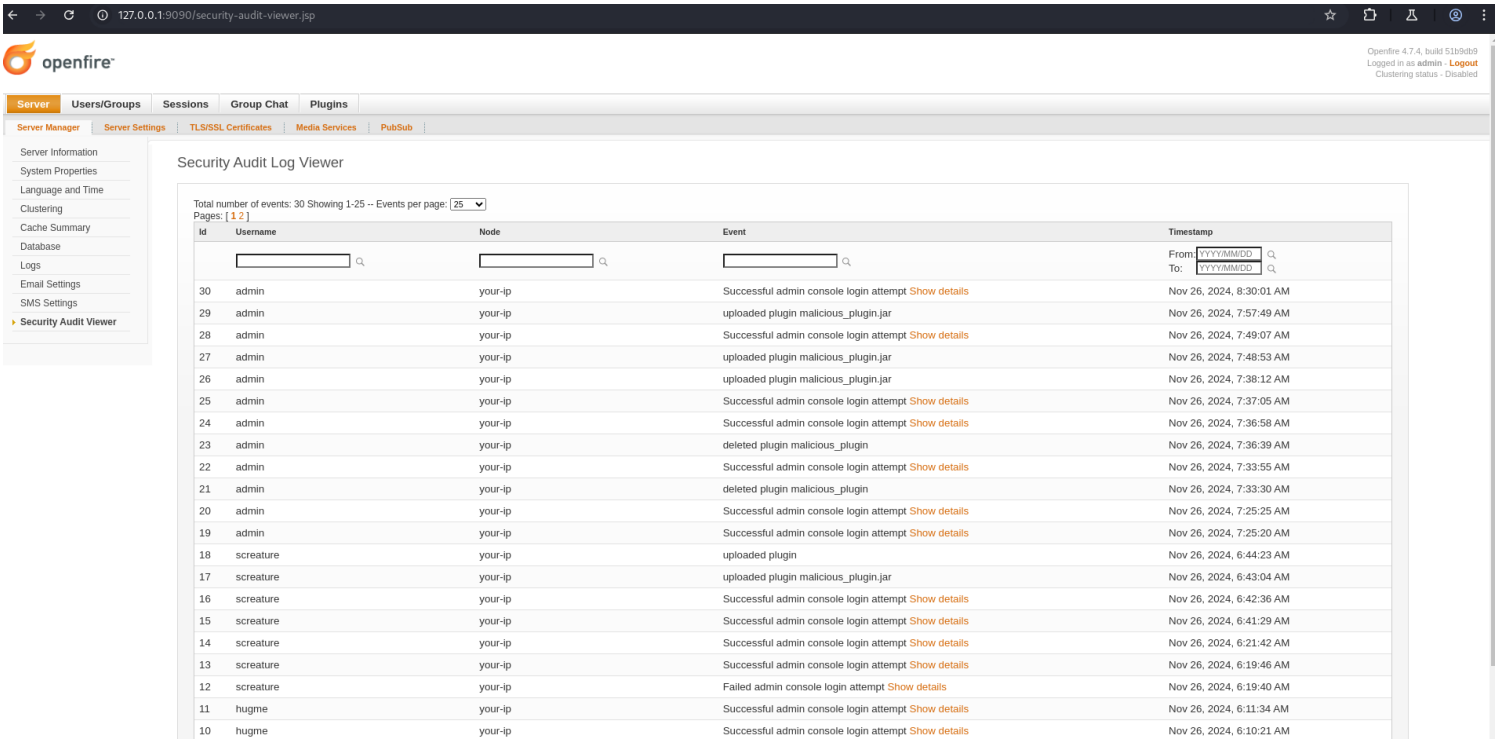
This can be seen below.
Upon a successful execution of this plugin by the server, if its a revshell file, then it will call back to the attackers machine which can lead to comlete breach and takeover of the system.
Without authentication, the plugin is accepted and installed.



NOTE: From there you can trivially pivot inward, remove the webshell, and hide within the system. All without creating the administrative user and making a mess in the log files.

CONCLUSION

Fortunately for defenders, the admin user creation is noisy. This is because the Openfire Security Audit Log can not be cleared.



Unfortunately for defenders, attackers don't need to create a user or authenticate to upload a plugin which can be used to gain RCE:)

Unfortunately, an attacker could use the path traversal to delete the log file. Depending on the permissions of the Openfire user, the attacker might be able to delete the log file via the webshell/reverse shell,which leaves the plugin itself as the only artifact that indicates exploitation.