

## Projet Réseaux : Portail captif

Adham El karn  
Mehmet Ozdemir

### Cross site request forgery

C'est une attaque qui consiste à utiliser l'authentification d'un utilisateur pour lui faire exécuter des action à son insu.

Le token peut être un moyen pour s'assurer qu'une requête vient effectivement du bon utilisateur et non pas d'un pirate.

Lorsque on fait la requête vers le serveur CAS, il nous renvoie un token qu'on renvoie ensuite au serveur ce qui permet au serveur CAS de s'assurer que la deuxième requête provient bien de l'utilisateur qui a fait la première requête.

Bien sûr le token doit être aléatoire et avoir une entropie élevée.

### Règles iptables

On bloque tout par défaut puis on autorise ce qui est nécessaire.

avant l'exécution du build architecture :

```
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$ sudo iptables -S
-P INPUT ACCEPT
-P FORWARD DROP
-P OUTPUT ACCEPT
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$ sudo iptables -t nat -S
-P PREROUTING ACCEPT
-P INPUT ACCEPT
-P OUTPUT ACCEPT
-P POSTROUTING ACCEPT
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$
```

Après l'exécution du build architecture et avant l'authentification du client :

on autorise les requête dns et on met en place la redirection vers le proxy.

```
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$ sudo iptables -S
-P INPUT ACCEPT
-P FORWARD DROP
-P OUTPUT ACCEPT
-A FORWARD -i openvswitch -p udp -m udp --dport 53 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$ sudo iptables -t nat -S
-P PREROUTING ACCEPT
-P INPUT ACCEPT
-P OUTPUT ACCEPT
-P POSTROUTING ACCEPT
-A PREROUTING -s 10.10.10.0/24 -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 3128
-A POSTROUTING -s 10.10.10.0/24 -p udp -m udp --dport 53 -j MASQUERADE
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal/netlab/web_proxytransparent$
```

### Après l'authentification du client :

```
aelk@archlinux:~$ sudo iptables -S
-P INPUT ACCEPT
-P FORWARD DROP
-P OUTPUT ACCEPT
-A FORWARD -s 10.10.10.10/32 -i openvswitch -p tcp -m multiport --dports 80,443 -j ACCEPT
-A FORWARD -i openvswitch -p udp -m udp --dport 53 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
aelk@archlinux:~$ sudo iptables -S -t nat
-P PREROUTING ACCEPT
-P INPUT ACCEPT
-P OUTPUT ACCEPT
-P POSTROUTING ACCEPT
-A PREROUTING -s 10.10.10.10/32 -i openvswitch -p tcp -m tcp --dport 80 -j ACCEPT
-A PREROUTING -s 10.10.10.0/24 -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 3128
-A POSTROUTING -s 10.10.10.10/32 -j MASQUERADE
-A POSTROUTING -s 10.10.10.0/24 -p udp -m udp --dport 53 -j MASQUERADE
aelk@archlinux:~$
```

On voit bien que les règles du firewall ont été ajoutées, après le succès de l'authentification.

```
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal$ sudo iptables -nvL
Chain INPUT (policy ACCEPT 5 packets, 269 bytes)
 pkts bytes target    prot opt in     out     source                 destination
Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                 destination
 18 1230 ACCEPT    tcp  --  openvswitch *      10.10.10.10          0.0.0.0/0             multiport dports 80,443
   6  344 ACCEPT    udp  --  openvswitch *          0.0.0.0/0            0.0.0.0/0             udp dpt:53
 20 17076 ACCEPT    all  --  *        *        0.0.0.0/0            0.0.0.0/0             state RELATED,ESTABLISHED
Chain OUTPUT (policy ACCEPT 7 packets, 414 bytes)
 pkts bytes target    prot opt in     out     source                 destination
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal$ sudo iptables -t nat -nvL
Chain PREROUTING (policy ACCEPT 2 packets, 116 bytes)
 pkts bytes target    prot opt in     out     source                 destination
   2  120 ACCEPT    tcp  --  openvswitch *      10.10.10.10          0.0.0.0/0             tcp dpt:80
   1   60 REDIRECT  tcp  --  *        *      10.10.10.0/24          0.0.0.0/0             tcp dpt:80 redir ports 3128
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                 destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                 destination
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                 destination
   4  236 MASQUERADE all  --  *        *      10.10.10.10          0.0.0.0/0
   2  112 MASQUERADE udp  --  *        *      10.10.10.0/24         0.0.0.0/0             udp dpt:53
aelk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal$
```

On voit que le redirect est fait une fois et que ensuite une fois authentifié c'est la cible accept qui est prise. Et qu'une règle de la table filter permet de router les packets venant du client (10.10.10.10).

Après la navigation du client à l'aide du navigateur :

```
aelk@archlinux:~$ sudo iptables -nvL
Chain INPUT (policy ACCEPT 183 packets, 23882 bytes)
 pkts bytes target    prot opt in     out     source            destination
Chain FORWARD (policy DROP 68 packets, 81164 bytes)
 pkts bytes target    prot opt in     out     source            destination
1312 143K ACCEPT    tcp  --  openvswitch *      10.10.10.10      0.0.0.0/0          multiport dports 80,443
 33 2089 ACCEPT    udp  --  openvswitch *      0.0.0.0/0        0.0.0.0/0          udp dpt:53
1440 1413K ACCEPT    all  --  *      *      0.0.0.0/0        0.0.0.0/0          state RELATED,ESTABLISHED
Chain OUTPUT (policy ACCEPT 190 packets, 20170 bytes)
 pkts bytes target    prot opt in     out     source            destination
aelk@archlinux:~$ sudo iptables -t nat -nvL
Chain PREROUTING (policy ACCEPT 151 packets, 87275 bytes)
 pkts bytes target    prot opt in     out     source            destination
 5 300 ACCEPT    tcp  --  openvswitch *      10.10.10.10      0.0.0.0/0          tcp dpt:80
 1 60 REDIRECT  tcp  --  *      *      10.10.10.0/24    0.0.0.0/0          tcp dpt:80 redir ports 3128
Chain INPUT (policy ACCEPT 3 packets, 180 bytes)
 pkts bytes target    prot opt in     out     source            destination
Chain OUTPUT (policy ACCEPT 9 packets, 553 bytes)
 pkts bytes target    prot opt in     out     source            destination
Chain POSTROUTING (policy ACCEPT 9 packets, 553 bytes)
 pkts bytes target    prot opt in     out     source            destination
 70 4315 MASQUERADE all  --  *      *      10.10.10.10      0.0.0.0/0
 2 112 MASQUERADE udp  --  *      *      10.10.10.0/24    0.0.0.0/0          udp dpt:53
aelk@archlinux:~$
```

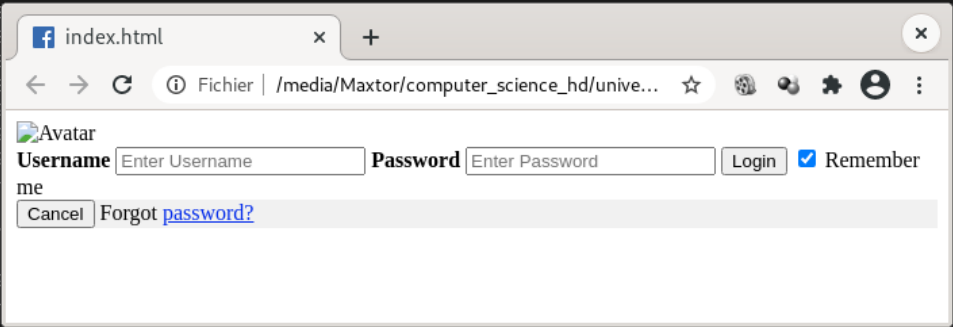
Formulaire d'authentification renvoyé au client lors de la redirection :

```
aelk@archlinux:~/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ wget google.com
--2020-12-23 16:41:20-- http://google.com/
Résolution de google.com (google.com)... 216.58.214.78, 2a00:1450:4007:807::200e
Connexion à google.com (google.com)|216.58.214.78|:80... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : non indiqué [text/html]
Sauvegarde en : « index.html »

index.html [ <=>

2020-12-23 16:41:20 (52,5 MB/s) - « index.html » sauvegardé [792]

aelk@archlinux:~/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ chromium index.html
[4558] parse server address: Unkn
[4558] parse server address: Unkn
[4558] parse server address: Unkn
[4558] libva error
[4558] h multiple threads in proc
[4558] error
Errors:
[4559] 17:455917:1223/164123_742257:ERR08:shared_context_state.cc(74): Skia shader compilation error
```



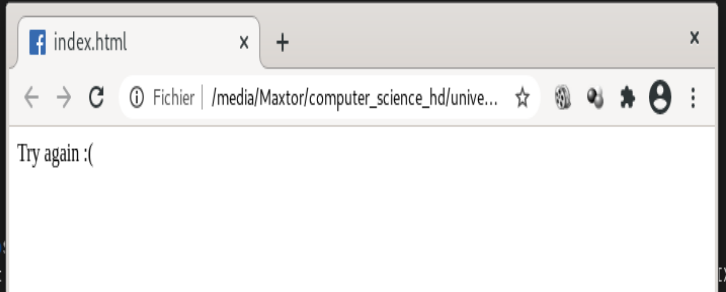
## Réponse au cas ou d'échec :

```
aelk@archlinux:/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ wget --post-data 'uname=falseusername&psw=falsepassword&remember=on' 10.10.10.1:3128
--2020-12-23 16:38:46-- http://10.10.10.1:3128/
Connexion à 10.10.10.1:3128... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : non indiqué [text/html]
Sauvegarde en : « index.html »

index.html [ <=>

2020-12-23 16:38:47 (25,7 B/s) - « index.html » sauvegardé [12]

aelk@archlinux:/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ chromium index.html
[453941:453976:1223/163857.823465:ERROR:bus.cc(393)] Failed to connect to the bus: Could not connect: Connection refused (X "unix")
[453941:453976:1223/163857.823490:ERROR:bus.cc(393)] Failed to connect to the bus: Could not parse server address: unknown address type (examples of valid types are "tcp" and on UNIX "unix")
[453941:453976:1223/163857.823495:ERROR:bus.cc(393)] Failed to connect to the bus: Could not parse server address: unknown address type (examples of valid types are "tcp" and on UNIX "unix")
```



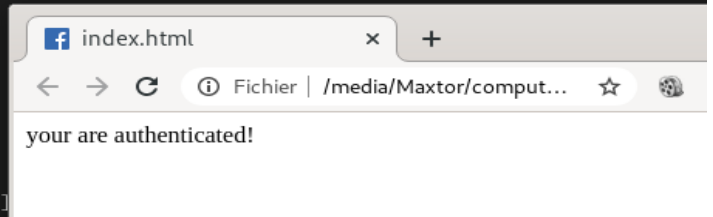
## En cas de réussite :

```
aelk@archlinux:/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ wget --post-data 'uname=falseusername&psw=falsepassword&remember=on' 10.10.10.1:3128
--2020-12-23 16:45:57-- http://10.10.10.1:3128/
Connexion à 10.10.10.1:3128... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : non indiqué [text/html]
Sauvegarde en : « index.html »

index.html

2020-12-23 16:45:59 (0,00 B/s) - « index.html » sauvegardé [0]

aelk@archlinux:/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ chromium index.html
[459817:459842:1223/164605.015770:58808:bus.cc(393)] Failed to connect to the bus: Could not parse server address: Unknown address type (examples of valid types are "tcp" and on UNIX "unix")
```



## Trace Tcpdump avant le Nat lors de la navigation du client vers google.com:

```
aelk@archlinux:/media/Maxtor/computer_science_hd/university/m1_cryptis/captive_portal/netlab$ sudo tcpdump -lnvv -i openvswitch 'tcp[13] & 2 != 0'
tcpdump: listening on openvswitch, link-type EN10MB (Ethernet), capture size 262144 bytes
16:57:54.367052 IP (tos 0x0, ttl 64, id 37151, offset 0, flags [DF], proto TCP (6), length 60)
    10.10.10.10.35974 > 216.58.201.238.80: Flags [S], cksum 0xb66b (incorrect -> 0xa513), seq 1063397169, win 64240, options [mss 1460,sackOK,TS val 3498257873 ecr 0,nop,wscale 7], length 0
16:57:54.413773 IP (tos 0x0, ttl 123, id 63226, offset 0, flags [none], proto TCP (6), length 60)
    216.58.201.238.80 > 10.10.10.10.35974: Flags [S.], cksum 0x8dc7 (correct), seq 3475595574, ack 1063397170, win 65535, options [mss 1430,sackOK,TS val 1981639627 ecr 3498257873,nop,wscale 8], length 0
16:57:54.519381 IP (tos 0x0, ttl 64, id 52079, offset 0, flags [DF], proto TCP (6), length 60)
```

## Trace Tcpdump après le Nat lors de la navigation du client vers google.com:

```
aelk@archlinux:/$ sudo tcpdump -lnvv -i wlp2s0 'tcp[13] & 2 != 0'
tcpdump: listening on wlp2s0, link-type EN10MB (Ethernet), capture size 262144 bytes
16:57:54.367120 IP (tos 0x0, ttl 63, id 37151, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.0.50.35974 > 216.58.201.238.80: Flags [S], cksum 0xf84c (correct), seq 1063397169, win 64240, options [mss 1460,sackOK,TS val 3498257873 ecr 0,nop,wscale 7], length 0
16:57:54.413724 IP (tos 0x0, ttl 124, id 63226, offset 0, flags [none], proto TCP (6), length 60)
    216.58.201.238.80 > 192.168.0.50.35974: Flags [S.], cksum 0xe100 (correct), seq 3475595574, ack 1063397170, win 65535, options [mss 1430,sackOK,TS val 1981639627 ecr 3498257873,nop,wscale 8], length 0
```

## Trace conntrack du passage de la connexion (initée par le client) par le routeur

```
elk@archlinux:~/hd/computer_science_hd/university/ml_cryptis/captive_portal$ sudo conntrack -E
[NEW] udp      17 30 src=10.10.10.10 dst=8.8.8.8 sport=41980 dport=53 [UNREPLIED] src=8.8.8.8 dst=192.168.0.50 sport=53 dport=41980
[UPDATE] udp   17 30 src=10.10.10.10 dst=8.8.8.8 sport=41980 dport=53 src=8.8.8.8 dst=192.168.0.50 sport=53 dport=41980
[UPDATE] udp   17 30 src=10.10.10.10 dst=8.8.8.8 sport=41980 dport=53 src=8.8.8.8 dst=192.168.0.50 sport=53 dport=41980 [ASSURED]
[NEW] tcp      6 120 SYN_SENT src=10.10.10.10 dst=216.58.213.174 sport=41446 dport=80 [UNREPLIED] src=216.58.213.174 dst=192.168.0.50 sport=80 dport=41446
[UPDATE] tcp    6 60 SYN_RECV src=10.10.10.10 dst=216.58.213.174 sport=41446 dport=80 src=216.58.213.174 dst=192.168.0.50 sport=80 dport=41446
[UPDATE] tcp    6 432000 ESTABLISHED src=10.10.10.10 dst=216.58.213.174 sport=41446 dport=80 src=216.58.213.174 dst=192.168.0.50 sport=80 dport=41446 [ASSURED]
```

Chaque client authentifié à l'aide du protocole lemonLdap est autorisé à faire des requête vers des serveurs web.

Un des problèmes rencontrés est les priorité des règles du firewall pour cela nous avons utilisé pour certaines règles '-I' au lieu de '-A' pour insérer des règles avant toutes les autres règles.

Pour faire une démo :

dans le netns root:

```
sudo ./clean; sudo ./build_architecture
```

lancer deux netns, dans deux terminaux différents:

```
sudo ip netns exec h1 sudo -u user bash
```

```
sudo ip netns exec h2 sudo -u user bash
```

attributions des ip au netns du réseaux privé(10.10.10.0/24)

dans le netns root

```
sudo dnsmasq -d -z -i openvswitch -F10.10.10.10,10.10.10.20,255.255.255.0
```

dans les netns du réseau privé:

```
sudo dhclient
```

ensuite lancer dans le netns root le proxy transparent :

```
cd web_proxytransparent
```

```
sudo ./http_server.py 3128
```

et ensuite dans les netns du réseau privé :

```
wget google.com → redirection ver le proxy
```

```
wget --post-data 'uname=(username ent) &psw=(password ent)&remember=on' 10.10.10.1:3128
```

et ensuite les clients du réseaux privé peuvent naviguer que ce soit sur des sites http ou https