Reconocimiento y recolección de evidencias

Al abrir el sistema de 4Geeks Academy atacada, se propone corregir y obtener evidencias suficiente para encontrar archivos sospechosos, por ende, se inicia la máquina hackeada, y tras un intento de entrada, el inicio de sesión es “debian” y su respectiva contraseña “123456”. Entrando en la máquina iniciamos el reconocimiento.

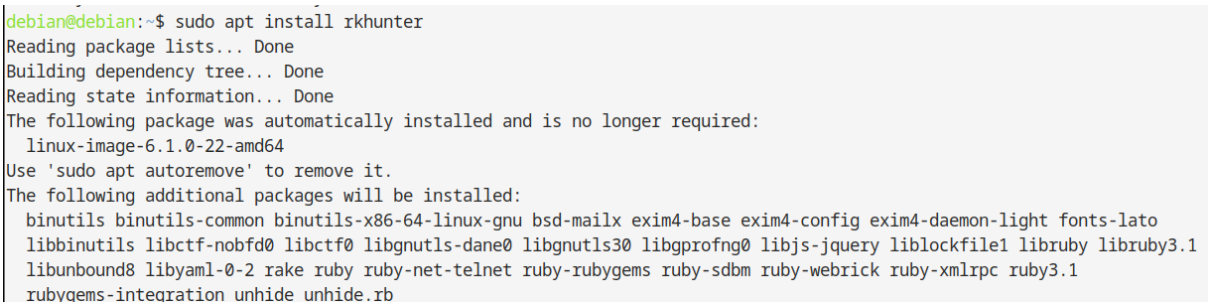
1. Reconocimiento

1.Rkhunter

Se instala “rkhunter” pero primero de todo haremos un “sudo apt update” continuando con un “sudo apt install rkhunter”.

```
debian@debian:~$ sudo apt install rkhunter
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-6.1.0-22-amd64
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu bsd-mailx exim4-base
  exim4-config exim4-daemon-light fonts-lato libbinutils libctf-nobfd0 libctf0
```

```
debian@debian:~$ sudo apt update
```



Con ello se usa el comando “sudo rkhunter --check” para a continuación esperar a que la herramienta examine la máquina para posibles errores o incluso malware detectado.

```
debian@debian:~$ sudo rkhunter --check
[ Rootkit Hunter version 1.4.6 ]

Checking system commands...

  Performing 'strings' command checks
```

```
    /usr/bin/lwp-request                                    [ Warning ]
    /usr/bin/bsd-mailx                                      [ OK ]
    /usr/bin/dash                                           [ OK ]
    /usr/bin/x86_64-linux-gnu-size                          [ OK ]
    /usr/bin/x86_64-linux-gnu-strings                       [ OK ]
    /usr/bin/inetutils-telnet                               [ OK ]
    /usr/bin/which.debianutils                              [ OK ]
    /usr/lib/systemd/systemd                                [ OK ]
```

Se encuentra el primer “**Warning”** (la primera alerta), con esto se busca la dirección del fichero y se analiza.

```
debian@debian:~$ nano /usr/bin/lwp-request
```

```
  GNU nano 7.2                  /usr/bin/lwp-request
#!/usr/bin/perl

# Simple user agent using LWP library.

=head1 NAME

lwp-request - Simple command line user agent

=head1 SYNOPSIS

B<lwp-request> [B<-afPuUsSedvhx>] [B<-m> I<method>] [B<-b> I<base URL>] [B<-t> >
            [B<-i> I<if-modified-since>] [B<-c> I<content-type>]
            [B<-C> I<credentials>] [B<-p> I<proxy-url>] [B<-o> I<format>] I<url>

=head1 DESCRIPTION

This program can be used to send requests to WWW servers and your
local file system. The request content for POST and PUT
methods is read from stdin.  The content of the response is printed on
stdout.  Error messages are printed on stderr.  The program returns a
```

Tras el “nano” se analiza que la herramienta en uso de “rkhunter” ha dado un falso positivo, es decir, no es un malware o rootkit, este proceso es algo normal en el uso de “perl”.

```
   Checking for suspicious (large) shared memory segments    [ Warning ]
   Checking for Apache backdoor                              [ Not found ]

 Performing Linux specific checks
   Checking loaded kernel modules                            [ OK ]
   Checking kernel module names                              [ OK ]
```

Y nos saltará una segunda alerta, la cual nos indica que hay sospechosamente una gran cantidad de memorias segmentadas compartidas.
Se investiga como lo hecho anteriormente.

```
[17:46:03] Warning: The following suspicious (large) shared memory segments have been found:
[17:46:03]          Process: /usr/bin/mate-panel    PID: 1141    Owner: debian    Size: 4.0MB (configured size allowed: 1.0MB)
[17:46:04]          Process: /usr/bin/caja    PID: 1170    Owner: debian    Size: 32MB (configured size allowed: 1.0MB)
[17:46:04]          Process: /usr/bin/mate-terminal    PID: 554016    Owner: debian    Size: 4.0MB (configured size allowed: 1
.0MB)
[17:46:04]          Process: /usr/bin/mate-screensaver    PID: 1199    Owner: debian    Size: 32MB (configured size allowed: 1
```

Con la foto anterior, “rkhunter” avisa de un segmento de memoria compartida por procesos como “mate-panel”, “caja”, “mate-screensaver” y “mate-terminal”, estos programas son legítimos de la herramienta del escrito “MATE”,

```
Performing system configuration file checks
  Checking for an SSH configuration file                    [ Found ]
  Checking if SSH root access is allowed                    [ Warning ]
  Checking if SSH protocol v1 is allowed                    [ Not set ]
  Checking for other suspicious configuration settings      [ None found ]
  Checking for a running system logging daemon              [ Found ]
  Checking for a system logging configuration file          [ Found ]
```

Y una tercera, por el cual tenemos “root access” (acceso a la terminal en forma de Admin) y se debería de cambiar para que no sea así.

```
File properties checks...
    Files checked: 144
    Suspect files: 1

Rootkit checks...
    Rootkits checked : 497
    Possible rootkits: 6

Applications checks...
    All checks skipped

The system checks took: 4 minutes and 51 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)
```

Con ello finaliza el rkhunter, añadiendo 6 posibles rootkits encontrados en la máquina. Investigamos un poco más a fondo en el archivo de registro que nos proporciona el propio “rkhunter”, por ello haremos uso de:

```
debian@debian:~$ sudo cat /var/log/rkhunter.log
```

Encontramos el segundo aviso aunque más largo, encontrando donde se ubican.

```
[13:53:07]   Checking if SSH root access is allowed          [ Warning ]
[13:53:07] Warning: The SSH and rkhunter configuration options should be the same:
[13:53:07]          SSH configuration option 'PermitRootLogin': yes
[13:53:07]          Rkhunter configuration option 'ALLOW_SSH_ROOT_USER': no
```

Además el puerto “SSH” el 22, está mal configurado o lo han reconfigurado, encontrándose en “etc/ssh/sshd_config” usando “sudo nano etc/ssh/sshd_config”.

```
debian@debian:~$ sudo nano /etc/ssh/sshd_config
```

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

En la parte de “PermitRootLogin yes” cambiamos el si por un no. Para estar más seguros, usaremos la siguiente herramienta “chkrootkit”.

2. Chkrootkit

Es una herramienta de seguridad de código abierto diseñada para sistemas de Unix y Linux, detectando localmente rastros de “rootkits” y otras intrusiones.

La instalación es fácil, se necesita tener actualizado el software con “sudo apt update”, a continuación la instalación es sencilla con “sudo apt install chkrootkit”, aceptando la instalación ya la tendremos para usar.

```
debian@debian:~$ sudo apt install chkrootkit
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-6.1.0-22-amd64
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  chkrootkit
0 upgraded, 1 newly installed, 0 to remove and 306 not upgraded.
Need to get 307 kB of archives.
After this operation, 950 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 chkrootkit amd64 0.57-2+b8 [307 kB]
Fetched 307 kB in 0s (3,928 kB/s)
Selecting previously unselected package chkrootkit.
(Reading database ... 174308 files and directories currently installed.)
Preparing to unpack .../chkrootkit_0.57-2+b8_amd64.deb ...
Unpacking chkrootkit (0.57-2+b8) ...
Setting up chkrootkit (0.57-2+b8) ...
Created symlink /etc/systemd/system/timers.target.wants/chkrootkit.timer → /lib/systemd/system/chkrootkit.timer.
Processing triggers for man-db (2.11.2-2) ...
debian@debian:~$ sudo chkrootkit
```

Después de la instalación, se inicia la herramienta usando “sudo chkrootkit”, con esto buscará directamente rastros de “rootkits”.

1. Primer aviso

Al iniciar el programa y luego de unos minutos, encontramos el primer aviso, directorios y archivos sospechosos en la ruta de “/usr/bin/ruby/gems/3.1.0/gems/typeprof-0.21.2 /vscode/.vscodeignore”.

```
Searching for suspicious files and dirs...                    WARNING

WARNING: The following suspicious files and directories were found:
/usr/lib/ruby/gems/3.1.0/gems/typeprof-0.21.2/vscode/.vscodeignore
/usr/lib/ruby/gems/3.1.0/gems/typeprof-0.21.2/vscode/.gitignore
/usr/lib/ruby/gems/3.1.0/gems/typeprof-0.21.2/vscode/.vscode
/usr/lib/ruby/vendor_ruby/rubygems/ssl_certs/.document
/usr/lib/ruby/vendor_ruby/rubygems/tsort/.document
/usr/lib/ruby/vendor_ruby/rubygems/optparse/.document
/usr/lib/libreoffice/share/.registry
```

Estos no son malware, ya que son archivos normales, por ejemplo “VS code” es un editor de código desarrollado por Microsoft, para esto no es ningún archivo extraño o dañino, además “.git” es del repositorio de “Git”, contiene toda la información necesaria para el control de versiones y el historial de estas. Los “.document” y el “.registry”, el primero es un archivo de control para “Ruby”, es creado automáticamente y el segundo es un archivo de “LibreOffice”, guardan información de configuración y extensiones.
En conclusión es un warning que no hay nada comprometido, son archivos legítimos, únicamente el “chkrootkit” no detecta los nuevos archivos modernos, como son el “Git” y el “VS code”.
Se encuentra otro aviso en la aplicación, un posible husmeador el cual analiza nuestros paquetes.

2. **Segundo aviso**

Además de otro aviso de un posible husmeador el cual analiza nuestros paquetes.

```
Checking `sniffer'...                                         WARNING

WARNING: Output from ifpromisc:
lo: not promisc and no packet sniffer sockets
enp0s3: PACKET SNIFFER(/usr/sbin/NetworkManager[528])
```

Por ello se investiga para saber qué hace el sistema mencionado, el “ifpromisc” es una utilidad que revisa las interfaces red si está en

modo promiscuo, es decir, observar si la tarjeta recibe todo el tráfico que pasa por la red, independientemente si los paquetes están dirigidos a ella o no.
Además se explica en la siguiente línea, ya que no está en modo promiscuo ni hay intercepción en ella. Aunque en la red principal afectada hay un proceso “NetworkManager [528]” el cuál si tiene los paquetes en captura abiertos.
Este proceso es un falso positivo, al ser un procedimiento por el cual detecta redes y maneja el Wi-Fi en sí.

3. Netstat y comprobación de puertos

Al tener una vulnerabilidad en la máquina, se comprueba los puertos encendidos, para analizar las vulnerabilidades.

```
debian@debian:~$ sudo netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      545/cupsd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      579/sshd: /usr/sbin
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      669/mariadbd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      3060/exim4
tcp6       0      0 ::1:631                 :::*                    LISTEN      545/cupsd
tcp6       0      0 :::80                   :::*                    LISTEN      670/apache2
tcp6       0      0 :::21                   :::*                    LISTEN      556/vsftpd
tcp6       0      0 :::22                   :::*                    LISTEN      579/sshd: /usr/sbin
tcp6       0      0 ::1:25                  :::*                    LISTEN      3060/exim4
udp        0      0 0.0.0.0:51166           0.0.0.0:*                           507/avahi-daemon: r
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           507/avahi-daemon: r
udp6       0      0 :::5353                 :::*                                507/avahi-daemon: r
udp6       0      0 :::55573                :::*                                507/avahi-daemon: r
```

Tras el “netstat -tulpn” se observa varias cosas, el puerto 21 (vsftpd), no es necesario, ya que el puerto 22 hace su misma función agregando un cifrado de los datos en este puerto.
Otra cosa que cabe recalcar es el puerto 5353, el “avahi-daemon”, suele ser innecesario en todos los aspectos.
Por ello se analiza puerto por puerto para asegurarnos que no esté expuesto.

1. Cupsd (puerto 631)

Este puerto es utilizado como sistema de impresión, es predeterminado para sistemas como Linux y macOS, con este puerto activado facilita la comunicación entre ordenador y dispositivo de impresión.

- Para analizarlo, se usa el siguiente comando: “sudo netstat -tulpn | grep cupsd”, priorizando los dispositivos de escucha que están a la espera de este puerto.

```
debian@debian:~$ sudo netstat -tulpn | grep cupsd
[sudo] password for debian:
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
795/cupsd
tcp6       0      0 ::1:631                 :::*                    LISTEN
795/cupsd
```

Examinando a fondo el proceso, se supervisa que sólo es accesible si desde el propio equipo, no es accesible ni desde red ni de internet. En conclusión el puerto no ha sido amenazado.

2. Puerto 22 (SSH)

Es el estándar asignado hacia el servicio de SSH, fundamental para la administración de sistemas Linux y macOS para realizar conexiones remotas seguras.

1. Verificar

Para analizarlo se usará el comando anterior : “sudo netstat -tulpn | grep :22”. El “:22” prioriza las conexiones hacia el puerto 22.

```
debian@debian:~$ sudo netstat -tulpn |grep ":22"
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      574/sshd: /usr/sbin
tcp6       0      0 :::22                   :::*                    LISTEN      574/sshd: /usr/sbin
```

Este es un poco extraño, el puerto no solamente está siendo escuchado por la máquina, sino que cualquier persona con acceso a internet, puede acceder a esta máquina por el puerto 22, además con la autenticación vulnerable anterior, tendrá permisos de administrador y podrá leer o incluso modificar ficheros.
Aunque solo sea una superstición, se corrobora si alguien ha entrado en el puerto 22, también se asegurará de arreglar el puerto para futuros o posibles amenazas posteriores.
Para verificar que nadie ha podido entrar en el puerto 22, con el siguiente comando “sudo grep sshd /var/log/ auth.log” se puede saber los que han entrado en ese puerto.

```
debian@debian:~$ sudo grep sshd /var/log/auth.log
[sudo] password for debian:
grep: /var/log/auth.log: No such file or directory
debian@debian:~$ sudo grep sshd /var/log/
alternatives.log    boot.log.1          cups/              fontconfig.log      private/            wtmp
alternatives.log.1  boot.log.2          dpkg.log           installer/          README              Xorg.0.log
apache2/            btmp                dpkg.log.1         journal/            rkhunter.log        Xorg.0.log.old
apt/                btmp.1              exim4/             lastlog             runit/
boot.log            chkrootkit/         faillog            lightdm/            speech-dispatcher/
```

Tras el comando, no hay ningún fichero que sea del tipo “auth.log”, aunque se observa un “journal”, seguramente esta máquina usa journal en vez de logs. Así que para hacer la verificación, se usará el comando “sudo journalctl -u ssh | grep “Accepted””, se utiliza “-u” para filtrar intentos de inicio de sesión y además se usa “grep Accepted” para todos los intentos de autenticación aceptadas.

```
debian@debian:~$ sudo journalctl -u ssh | grep "Accepted"
Oct 08 17:40:59 debian sshd[1650]: Accepted password for root from 192.168.0.134 port 45623 ssh2
```

2. Solucionar

Para solucionar el problema de la entrada en el puerto 22, se debe reforzar para posibles próximas amenazas o incluso no repetir los mismos errores, con el cambio anterior de “PermitRootLogin yes” a un no, se debe reiniciar el sistema.

```
debian@debian:~$ sudo systemctl restart ssh
```

Para reforzar el sistema, se utilizará un firewall (“ufw”), con este firewall nos ayuda a que ningún usuario pueda llegar a encontrar el “ssh” a menos que sea la propia máquina.
La instalación es la siguiente:

```
debian@debian:~$ sudo apt install ufw
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-6.1.0-22-amd64
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  iptables libip6tc2
Suggested packages:
  firewalld rsyslog
The following NEW packages will be installed:
  iptables libip6tc2 ufw
0 upgraded, 3 newly installed, 0 to remove and 306 not upgraded.
Need to get 548 kB of archives.
```

```
debian@debian:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw allow from 192.168.1.130 to any port 22 proto tcp
Rules updated
debian@debian:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

- Lo primero de todo es la instalación del firewall que se usará, con el comando “sudo apt install ufw”.
- Se necesita definir las políticas por defecto, por ello los siguientes comandos “sudo ufw default deny incoming” y “sudo ufw default allow outgoing”.
- Permitimos el tráfico con una única <IP>, la de nuestra máquina, “sudo ufw allow from <IP de la debian> to any port 22 proto tcp”, con ello otra IP no podrá entrar al puerto 22, y solo nuestro sistema sí.

3. Mariadbd (puerto 3306)

El proceso de “Mariadbd” es el proceso que usa “Mariadb”, este es un sistema de gestión de bases de datos, diseñado para almacenar, organizar y recuperar grandes volúmenes de información de manera eficiente.

- Se analiza para comprobar su veracidad con el comando usado anteriormente sustituyendo el grep por “mariadbd”.

```
debian@debian:~$ sudo netstat -tulpn | grep mariadbd
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      710/mariadbd
```

Tras la examinación, este puerto sólo es accesible desde el propio equipo, no es accesible ni desde red ni de internet. Por ende el puerto no ha sido amenazado.

4. Apache (puerto 80)

Apache es un software que convierte un ordenador en un servidor web. Su función principal es recibir peticiones de los navegadores y enviarles los archivos de una página web para poder verlas.

1. Verificar

- Se analiza individualmente para mejorar el enfoque a este puerto en concreto.

```
debian@debian:~$ sudo netstat -tulpn | grep apache
tcp6       0      0 :::80                   :::*                    LISTEN      669/apache2
```

Este puerto es igual que el puerto “SSH”, por el cual acepta conexiones desde cualquier interfaz de la red, por ello se necesita una validación igual al puerto 22.

```
debian@debian:~$ sudo tail /var/log/apache2/access.log
[sudo] password for debian:
127.0.0.1 - - [22/Jan/2026:13:18:27 -0500] "GET / HTTP/1.1" 200 10956 "-" "curl/7.88.1"
```

Con el comando “sudo tail /var/log/apache2/access.log”, podemos comprobar los últimos accesos al servidor de apache2, el cual ha sido el día 22 del año 2026, un acceso que ha sido mediante un curl, así que podemos descartar la entrada a este puerto, aún así es necesario hacer un par de ajustes.

2. Ajustes

Para un buen uso del puerto de apache2, una actualización del puerto 80, con dos simples comandos “sudo apt update” y un “sudo apt upgrade apache2”, con esto se actualizará apache2 y se tendrá más seguridad en el puerto, además de reiniciar el puerto para aplicar los parches “sudo systemctl restart apache2”. También se agrega un firewall de forma que solo pueden acceder de forma local con el “ufw”, con el comando “sudo ufw allow in on lo”, lo que permite todo el tráfico entrante que venga solo de nuestra máquina.

```
debian@debian:~$ sudo apt update
Get:1 http://security.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Hit:2 http://deb.debian.org/debian bookworm InRelease
Get:3 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:4 http://security.debian.org/debian-security bookworm-security/main Sources [196 kB]
Get:5 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [292 kB]
Get:6 http://security.debian.org/debian-security bookworm-security/main Translation-en [178 kB]
Fetched 770 kB in 0s (1,849 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
307 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
debian@debian:~$ sudo apt upgrade apache2
```

```
debian@debian:~$ sudo systemctl restart apache2
debian@debian:~$ sudo ufw allow in on lo
Rule added
Rule added (v6)
```

Y con estos ajustes se obtiene un puerto de apache2 más seguro.

5. Vsftpd (puerto 21)

El “vsftpd” es un servidor para la transferencia de datos para los sistemas de Linux y macOS. Su función permite que un usuario suba, descargue o gestione archivos en un servidor remoto de forma eficiente.

- Se comprueba el estado del puerto vsftpd, por el con el comando anterior “sudo netstat -tulpn | grep vsftpd”.

```
debian@debian:~$ sudo netstat -tulpn | grep vsftpd
tcp6       0      0 :::21                   :::*                    LISTEN      572/vsftpd
```

El puerto vsftpd está a la espera de conexiones hacia este, por el cual comprobaremos antiguos inicio de sesión para estar más tranquilos. Con el comando “sudo journalctl -u vsftpd”.

```
debian@debian:~$ sudo journalctl -u vsftpd
Oct 08 16:09:01 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:09:01 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
Oct 08 16:10:37 debian systemd[1]: Stopping vsftpd.service - vsftpd FTP server...
Oct 08 16:10:37 debian systemd[1]: vsftpd.service: Deactivated successfully.
Oct 08 16:10:37 debian systemd[1]: Stopped vsftpd.service - vsftpd FTP server.
Oct 08 16:10:37 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:10:37 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot 342683d8f35244b08c4f3863f2978eca --
Oct 08 16:43:18 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:43:18 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot d28e179bf5884b25bf94452c79fd0afa --
Oct 08 16:48:02 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:48:02 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot af0a79f76920440c8e08594d6547449b --
Oct 08 17:28:37 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 17:28:37 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot c725eaa952de466e97fb10897bcdc9f4 --
Jan 21 16:56:30 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Jan 21 16:56:30 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot 78c0dff84bbc42ae968ebc0a5eeeac9b --
Jan 21 17:01:42 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Jan 21 17:01:42 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
-- Boot 2beacdc04f48434b9d6a4942508eeaac --
Jan 22 04:49:05 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Jan 22 04:49:05 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
```

Por lo visto, nadie ha entrado a este puerto, no hay intentos ni conexiones a este puerto, lo único es los reinicios automáticos. Por tanto este puerto no ha sido comprometido.

6. Exim4 (puerto 25)

Este es un agente de transporte de correo, para el uso de los sistemas de Unix. Su función principal es gestionar el envío y la recepción de mensajes de correo electrónico en el servidor.

- Se valida el estado del “Exim4” con la ayuda del “sudo netstat -tulpn | grep exim4”

```
debian@debian:~$ sudo netstat -tulpn | grep exim4
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1118/exim4
tcp6       0      0 ::1:25                  :::*                    LISTEN      1118/exim4
```

Tras supervisar y revisar el puerto, este sólo es accesible desde el propio equipo, no es accesible ni desde red ni de internet. Por ende el puerto no ha sido amenazado.1

7. Avahi-daemon

Por último “Avahi-daemon” es un servicio para sistemas tipo Linux y Unix . Su función principal es permitir que los dispositivos de una red local se encuentren y se comuniquen entre sí de forma automática, sin necesidad de configurar direcciones “IP” manualmente o usar un servidor DNS.

- Este se analiza aunque el puerto no está en escucha y ni siquiera es posible entrar desde este puerto. Con lo visto anteriormente se usa el comando “sudo netstat -tulpn | grep avahi-daemon”.

```
debian@debian:~$ sudo netstat -tulpn | grep avahi-daemon
[sudo] password for debian:
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           51129/avahi-daemon:
udp        0      0 0.0.0.0:57634           0.0.0.0:*                           51129/avahi-daemon:
udp6       0      0 :::56553                :::*                                51129/avahi-daemon:
udp6       0      0 :::5353                 :::*                                51129/avahi-daemon:
```

Al ser un puerto “UDP” no acepta logins ni sesiones ni ejecuta comandos, por ello este puerto no resultaría en ninguna intrusión real.

4. Conclusión

Con todo lo recolectado anteriormente, se observa que no ha habido intrusión en ningún puerto exceptuando solamente en el puerto 22, por el cual se han hecho sus respectivas observaciones y parches de seguridad para la ayuda contra este tipo de amenazas.

2. Recolección de evidencias

Con ello se debe hacer una recolección de las evidencias que tenemos por el cual se ha interceptado un acceso irregular en el puerto 22, por el cual se ha modificado y se ha recolectado su entrada a la máquina atacada, con ello se expone en este apartado todo lo recaudado para notificar al afectado lo que se necesita para su actuación.

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

```
debian@debian:~$ sudo journalctl -u ssh | grep "Accepted"
Oct 08 17:40:59 debian sshd[1650]: Accepted password for root from 192.168.0.134 port 45623 ssh2
```

Con estas dos capturas, se declara realmente una clara intrusión hacia nuestra máquina. Con las recomendaciones anteriores, se suprime la vulnerabilidad anterior y así protegemos la máquina.

3. Recomendaciones

Este sistema que se ha ofrecido, tiene defectos como todos, aunque haya una intrusión, ha sido solucionada y arreglada.
Por ello se van a sugerir actualizaciones por el cual sería más seguro para el sistema.

1. Actualizaciones

Lo primero de todo es hacer una actualización de todo el sistema, para hacerlo se necesita actualizar la máquina, “sudo apt update” y “sudo apt upgrade -y” con estos comandos, hacen que el sistema se actualice y además sin necesidad de aprobarlo ya que se ha aprobado previamente con “-y”.

```
debian@debian:~$ sudo apt update
[sudo] password for debian:
Hit:1 http://deb.debian.org/debian bookworm InRelease
Get:2 http://security.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
Get:4 http://security.debian.org/debian-security bookworm-security/main Sources [196 kB]
Get:5 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [292 kB]
Fetched 537 kB in 1s (904 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
```

```
debian@debian:~$ sudo apt upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  libdaxctl1 libndctl6 libpmem1 linux-image-6.1.0-22-amd64 linux-image-6.1.0-23-amd64
Use 'sudo apt autoremove' to remove them.
The following packages will be upgraded:
  python3-urllib3
1 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 114 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://security.debian.org/debian-security bookworm-security/main amd64 python3-urllib3 all 1.26.12-1+deb12u3 [114 kB]
Fetched 114 kB in 0s (1,287 kB/s)
apt-listchanges: Reading changelogs...
(Reading database ... 179601 files and directories currently installed.)
Preparing to unpack .../python3-urllib3_1.26.12-1+deb12u3_all.deb ...
Unpacking python3-urllib3 (1.26.12-1+deb12u3) over (1.26.12-1+deb12u2) ...
Setting up python3-urllib3 (1.26.12-1+deb12u3) ...
```

2. Eliminar puertos

Eliminar puertos innecesarios, hay puertos que hacen la misma función que otros puertos, por ejemplo, el puerto 22 hace la misma función lo único que mejora, ya que cifra los datos que pasan por ella e incluso si están en reposo. La recomendación es la siguiente:

1. **Puerto 21**

Este puerto no es necesario, ya que los datos que pasan por esta no están cifrados, además de ser un objetivo clave para ataques, el puerto será sustituido por el “SSH”, comentado anteriormente, hace la misma función pero mejorando el cifrado.

Para cerrar el servicio se usaría “sudo systemctl stop vsftpd” y “sudo systemctl disable vsftpd”.

```
debian@debian:~$ sudo systemctl stop vsftpd
debian@debian:~$ sudo systemctl disable vsftpd
Synchronizing state of vsftpd.service with SysV service script with /lib/systemd
/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable vsftpd
Removed "/etc/systemd/system/multi-user.target.wants/vsftpd.service".
debian@debian:~$
```

2. **Puerto cups (631)**

Este puerto aunque no esté expuesto a internet, si hubiese otra intrusión podría activarla para exponerla, por ello si no está en uso, es mejor apagarlo.

Con ello si se necesitase cerrar, se usa los mismos comandos mencionados anteriormente, “sudo systemctl stop cups” y “sudo systemctl disable cups”.

```
debian@debian:~$ sudo systemctl disable cups
Synchronizing state of cups.service with SysV service script with /lib/systemd/s
ystemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable cups
Removed "/etc/systemd/system/sockets.target.wants/cups.socket".
Removed "/etc/systemd/system/multi-user.target.wants/cups.path".
Removed "/etc/systemd/system/multi-user.target.wants/cups.service".
Removed "/etc/systemd/system/printer.target.wants/cups.service".
```

3. Firewalls

Aunque se hayan implementado los firewalls para dos de los puertos anteriores, se necesita un buen firewall para detener por si es expuesto algún puerto en concreto.

```
debian@debian:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw allow from 192.168.1.130 to any port 22 proto tcp
Rules updated
debian@debian:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

```
debian@debian:~$ sudo systemctl restart apache2
debian@debian:~$ sudo ufw allow in on lo
Rule added
Rule added (v6)
```

Con las dos reglas anteriores en nuestro firewall de “ufw” debemos agregar las siguientes reglas si es necesario.

1. **Puerto 631:** Si vamos a hacer uso del puerto de cups y es necesario para la máquina se usa el comando “sudo ufw allow from 192.168.1.130 to any port 631 proto tcp” y bloqueamos la conexión que no sea nuestra “sudo ufw deny 631/tcp”.
2. **Puerto 5353:** El puerto de avahi-daemon aunque no esté siendo escuchado necesita también protección, usando los mismas reglas anteriores, “sudo ufw allow from 192.168.1.130 to any port 5353 proto udp” y “sudo ufw deny 5353/udp”

3. **Puerto 21:** El puerto de vsftpd si es necesario, se usará las mismas reglas, “sudo ufw allow from 192.168.1.130 to any port 5353 proto tcp” y “sudo ufw deny 21/tcp”

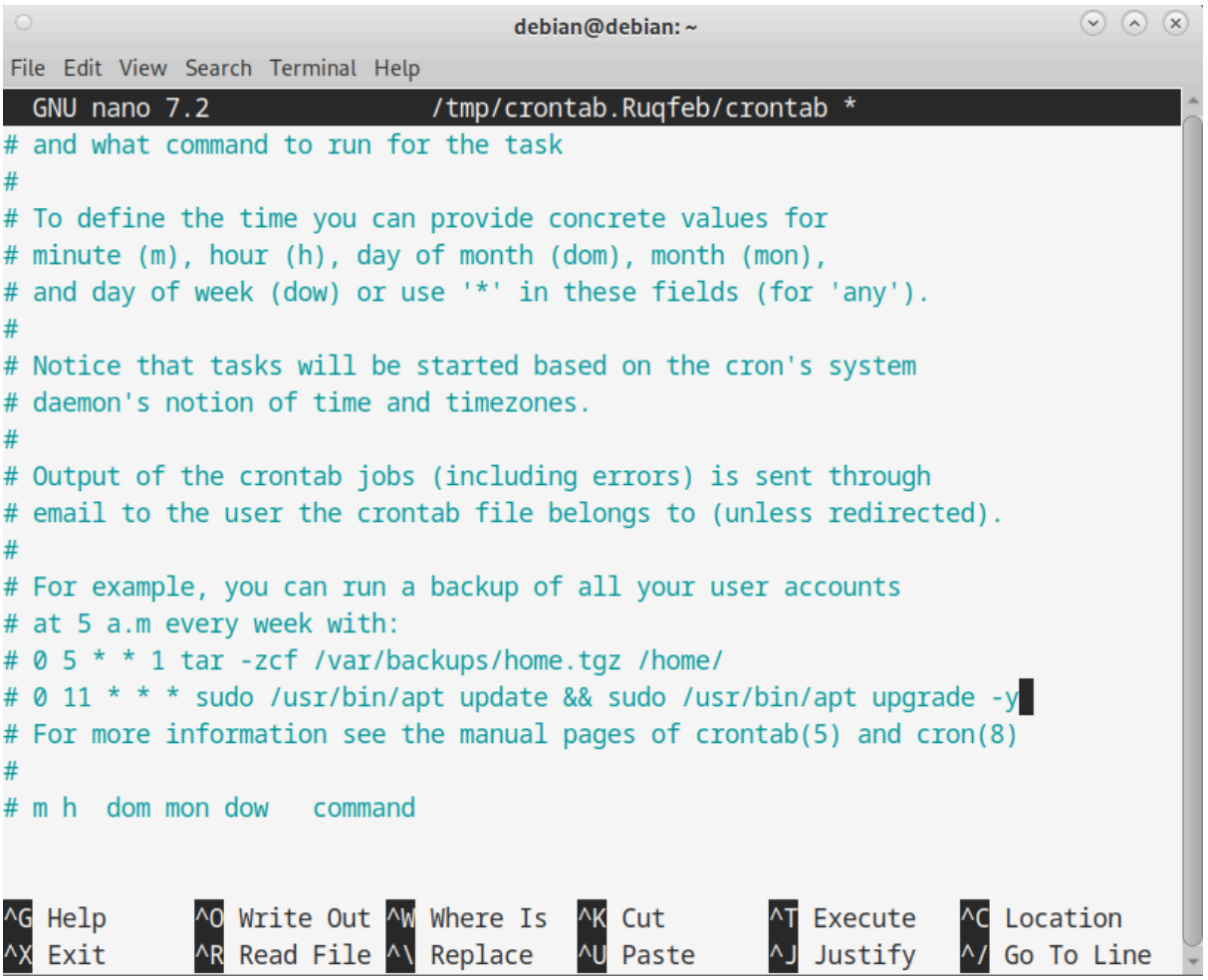
Con estas reglas anteriores en nuestro firewall, podemos estar más tranquilos conforme a nuestro sistema, ya que se necesitaría otros tipos de ataques para poder entrar en nuestra máquina.

4. Crontabs

Crontab es un archivo de texto en los sistemas de Unix que contiene listas de comandos destinados a ejecutarse de forma automática en fechas y horas específicas.

Este sistema ejecuta en segundo plano los archivos crontabs que están configurados en ella, para ello se necesita entrar en su archivos de textos se usa el comando de “crontab -e”.

```
debian@debian:~$ crontab -e
```



Para ello se programará para todos los días a las 11 de la mañana por el cual se actualizará y se mejorará automáticamente. Por ello se escribe el siguiente comando: “0 3 * * * sudo /usr/bin/apt update && sudo /usr/bin/apt upgrade -y”.

5. Conclusión final

Al haber mejorado la máquina después de la intrusión, no se debe pensar que con todos estas estas recomendaciones dadas se puede tener un sistema altamente seguro y sin riesgos ningunos, hay que pensar que hay otros tipos de malware como los phishing o incluso troyanos tras descargas infectadas. Por ello se debe tener cuidado cuando se navega por internet o incluso de los puertos que tenemos encendidos para no tener acceso comprometido hacia nuestra máquina.