

Proyecto final

Analisis Forense

**FASE I: RECONOCIMIENTO Y RECOLECCION
DE EVIDENCIAS.**

JOSE GOMEZ

INDICE

Introduccion.....	2
Logs del Sistema.....	2
SSH comprometido.....	4
Servidor FTP.....	5
Rookits y Malwares.....	6
Revision de Usuarios.....	7
WordPress.....	8
PARTE 2 : MITIGACION.....	10
Bloqueo Ips.....	10
Ssh.....	10
Mysql.....	10
WordPress.....	10
Ftp.....	11
Actualizacion.....	11
Recomendacion.....	12
FASE 2: DETECCION Y CORRECCION DE UNA VULNERABILIDAD	13
Introduccion.....	13
Objetivo.....	13
Deteccion vulnerabilidad.....	15
Explicacion Nmap.....	17
Explotacion.....	17
Medidas prevencion.....	20
FASE 3: PLAN DE RESPUESTA A INCIDENTES Y CERTIFICADOS.....	23
Conclusion	24

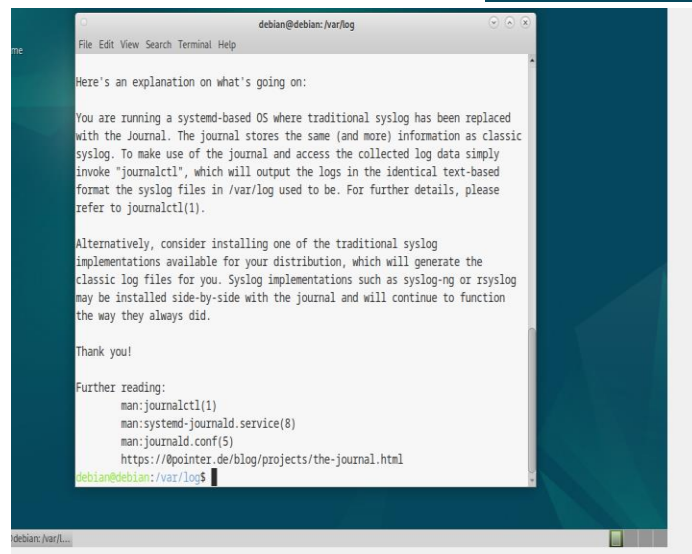
Introducción

Realizaremos un análisis exhaustivo de una máquina virtual Debian que ha sido comprometida. El objetivo es investigar el incidente, identificar las vulnerabilidades explotadas y aplicar las medidas necesarias para mitigar los riesgos y fortalecer la seguridad del sistema.

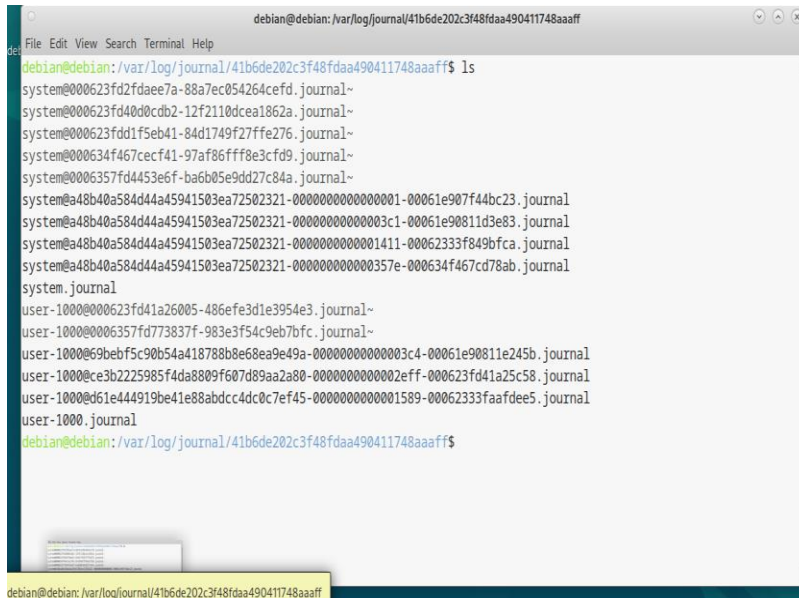
➤ LOGS DEL SISTEMA

Para acceder a los logs usamos el comando `cd /var/log/auth.log`, chequeamos el `syslog` y observamos que no aparecen, revisamos la carpeta `/var/log ls -all` y observamos un documento “README” el cual nos explica que la carpeta de logs ha sido comprometida, nos dice que usemos el “journal”.

```
bash: cd: /var/log/auth.log: No such file or directory
debian@debian:~$ cd /var/log
debian@debian:/var/log$ ls -all
total 1040
drwxr-xr-x 11 root      root      4096 May 14 12:35 .
drwxr-xr-x 12 root      root      4096 Sep 30 2024 ..
-rw-r--r-- 1 root      root      48068 Sep 30 2024 alternative
s.log
drwxr-x--- 2 root      adm       4096 Sep 30 2024 apache2
drwxr-xr-x 2 root      root      4096 Oct  8 2024 apt
-rw----- 1 root      root     86945 May 14 12:35 boot.log
-rw-rw---- 1 root      utmp     2688 Oct  8 2024 btmp
drwxr-xr-x 2 root      root      4096 Jul 31 2024 cups
-rw-r--r-- 1 root      root    765626 Oct  8 2024 dpkg.log
-rw-r--r-- 1 root      root         0 Jul 31 2024 faillog
-rw-r--r-- 1 root      root      5602 Sep 30 2024 fontconfig.
log
drwxr-xr-x 3 root      root      4096 Jul 31 2024 installer
drwxr-sr-x+ 3 root      systemd-journal 4096 Jul 31 2024 journal
-rw-rw-r-- 1 root      utmp         0 Jul 31 2024 lastlog
drwx--x--x 2 root      root      4096 May 14 12:35 lightdm
drwx----- 2 root      root      4096 Jul 31 2024 private
lrwxrwxrwx 1 root      root         39 Jul 31 2024 README -> .
./../usr/share/doc/systemd/README.logs
```

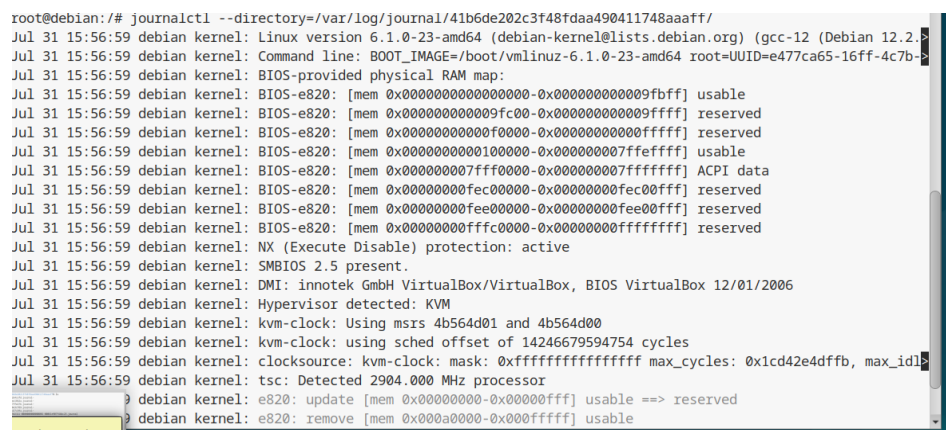


-Usamos el **"cd / var/log/journal"** y encontramos otro directorio en el cual podemos indagar un poco más los logs.



```
debian@debian: /var/log/journal/41b6de202c3f48fdaa490411748aaaff$ ls
system@000623fd2fdae7a-88a7ec054264cefd. journal~
system@000623fd4d08c0b2-12f2110dcea1862a. journal~
system@000623fdd1f5eb41-84d1749f27fe276. journal~
system@000634f467cecf41-97af86fff8e3cfd9. journal~
system@0006357fd4453e6f-ba6b05e9dd27c84a. journal~
system@a48b40a584d44a45941503ea72502321-0000000000000001-00061e907f44bc23. journal
system@a48b40a584d44a45941503ea72502321-000000000000003c1-00061e90811d3e83. journal
system@a48b40a584d44a45941503ea72502321-0000000000001411-0006233f849bfca. journal
system@a48b40a584d44a45941503ea72502321-0000000000000357e-000634f467cd78ab. journal
system. journal
user-1000@000623fd41a26005-486efe3d1e3954e3. journal~
user-1000@0006357fd773837f-983e3f54c9eb7bfc. journal~
user-1000@69beb5c90b54a418788b8e68ea9e49a-000000000000003c4-00061e90811e245b. journal
user-1000@ce3b2225985f4da8809f607d89aa2a80-0000000000002eff-000623fd41a25c58. journal
user-1000@d61e444919be41e88abdc4dc0c7ef45-0000000000001589-00062333faafdee5. journal
user-1000. journal
debian@debian: /var/log/journal/41b6de202c3f48fdaa490411748aaaff$
```

-Nos cambiamos a root con **sudo su** y usamos el **"journalctl -directory=/var/log/journal/41b6de202.(nombre entero del directorio)"**



```
root@debian: /# journalctl --directory=/var/log/journal/41b6de202c3f48fdaa490411748aaaff/
Jul 31 15:56:59 debian kernel: Linux version 6.1.0-23-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian 12.2.0-14)) #23~deb12u1 SMP PREEMPT_DYNAMIC Debian 6.1.0-23-amd64 (2024-07-25) root=UUID=e477ca65-16ff-4c7b-b000-000000000000 ro=initrd=initrd.img-6.1.0-23-amd64
Jul 31 15:56:59 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-6.1.0-23-amd64 root=UUID=e477ca65-16ff-4c7b-b000-000000000000 ro=initrd=initrd.img-6.1.0-23-amd64
Jul 31 15:56:59 debian kernel: BIOS-provided physical RAM map:
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x0000000000100000-0x00000000007fffff] usable
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x0000000007ff0000-0x0000000007ffffff] ACPI data
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x00000000fec00000-0x00000000fec0ffff] reserved
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x00000000fee00000-0x00000000fee0ffff] reserved
Jul 31 15:56:59 debian kernel: BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
Jul 31 15:56:59 debian kernel: NX (Execute Disable) protection: active
Jul 31 15:56:59 debian kernel: SMBIOS 2.5 present.
Jul 31 15:56:59 debian kernel: DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
Jul 31 15:56:59 debian kernel: Hypervisor detected: KVM
Jul 31 15:56:59 debian kernel: kvm-clock: Using msrs 4b564d01 and 4b564d00
Jul 31 15:56:59 debian kernel: kvm-clock: using sched offset of 14246679594754 cycles
Jul 31 15:56:59 debian kernel: clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb, max_idle: 0x0000000000000000
Jul 31 15:56:59 debian kernel: tsc: Detected 2904.000 MHz processor
Jul 31 15:56:59 debian kernel: e820: update [mem 0x00000000-0x00000000] usable ==> reserved
Jul 31 15:56:59 debian kernel: e820: remove [mem 0x00000000-0x00000000] usable
```

-Al examinar los logs descubrimos unos aspectos claves, la mejor manera para poder filtrar los servicios en los logs es usar **"grep"** y especiendo los comandos del sistema con **"journalctl --help"** nos aparece los siguientes comandos que podemos usar y explorar los servicios del sistema.

```

debian@debian:/var/log/journal$ journalctl -help
journalctl [OPTIONS...] [MATCHES...]

Query the journal.

Source Options:
  --system          Show the system journal
  --user            Show the user journal for the current user
  -M --machine=CONTAINER Operate on local container
  -m --merge        Show entries from all available journals
  -D --directory=PATH Show journal files from directory
  --file=PATH       Show journal file
  --root=ROOT       Operate on files below a root directory
  --image=IMAGE     Operate on files in filesystem image
  --namespace=NAMESPACE Show journal data from specified journal namespace

Filtering Options:
  -S --since=DATE    Show entries not older than the specified date
  -U --until=DATE    Show entries not newer than the specified date
  -c --cursor=CURSOR Show entries starting at the specified cursor
  --after-cursor=CURSOR Show entries after the specified cursor
  --cursor-file=FILE Show entries after cursor in FILE and update FILE

```

Filtramos con :

```

Source Options:
  --system          Show the system journal
  --user            Show the user journal for the current user
  -M --machine=CONTAINER Operate on local container

```

O

```

-u --unit=UNIT      Show logs from the specified unit

```

➤ SSH COMPROMETIDO

-Usamos ***“journalctl -u ssh.service | grep “password”*** “

```

root@debian:/home/debian# journalctl -u ssh.service | grep "password"
Oct 08 17:40:59 debian sshd[1650]: Accepted password for root from 192.168.0.134 port 45623
sh2

```

-Encontramos una actividad sospechosa si con una ip 192.168.0.134 por el puerto 45623

-Ahora que sabemos con certeza que alguien ha tenido un acceso no autorizado por el servicio SSH debemos revisar que servicios pueden haber sido comprometidos.

-Revisamos otros servicios para observar si también fueron comprometidos simplemente cambiando el comando “journalctl -u ftp.service | grep “password””.

```
ebian@debian:~$ sudo journalctl -u ftp.service
- No entries --
ebian@debian:~$
```

-Y aplicamos el filtro esta vez por fecha para verificar ya sabemos que es a partir del OCTUBRE 8

```
ebian@debian:~$ sudo journalctl -u ftp.service | grep "Oct"
[sudo] password for debian:
ebian@debian:~$ sudo journalctl -u ftp.service | grep "Oct"
```

Sin ningún resultado.

➤ SERVIDOR FTP

Revisamo el servicio antes mencionado con *“sudo nano /etc/vsftpd.conf”*

Encontramos el servicio en funcionamiento, pero con un fallo el cual permite un acceso anónimo sin restricciones.

```
# Run standalone? vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
listen=NO
#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (:::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES
#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=YES
```

➤ REVISION ROOKITS Y MALWARES

Usamos el comando *“sudo chkrootkit”*

Nos encontramos

```
Checking `sniffer'...
```

WARNING

```
WARNING: Output from ifpromisc:
```

```
lo: not promisc and no packet sniffer sockets
```

```
enp0s3: PACKET SNIFFER(/usr/sbin/NetworkManager[468], /usr/sbin/NetworkManager[468])
```

-MODO PROMISCUO (POSIBLE SNIFFING)

➤ REVISION DE USUARIOS

Revisamos los usuarios para saber si han sido comprometidos, revisamos los que tengan la especificación `"/bin/bash"` ya que son los que pueden crear shell y ejecutar comandos consultamos con `"cat /etc/passwd | grep "/bin/bash"`

```
root@debian:/home/debian# cat /etc/passwd | grep "/bin/bash"
root:x:0:0:root:/root:/bin/bash
debian:x:1000:1000:4geeks,,,:/home/debian:/bin/bash
```

-Solo los usuarios originales `"root"` y `"debian"`.

➤ MySQL

Revisamos mysql en búsqueda de mirar su base de datos para saber si han sido, modificados, usamos un comando con un archivo oculto `"sudo cat /root/.mysql_history"`

```
debian@debian:/$ sudo su
root@debian:/# cat /root/.mysql_history
_HiStOrY_V2_
CREATE\040DATABASE\040wordpress\040DEFAULT\040CHARACTER\040SET\040utf8\040COLLATE\040utf8_unicode
_ci;
CREATE\040USER\040'wordpressuser'@\040localhost'\040IDENTIFIED\040BY\040'123456';
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpress'@\040localhost';\040
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpressuser'@\040localhost';
FLUSH\040PRIVILEGES;
FLUSH\040PRIVILEGES;
EXIT;
CREATE\040USER\040'user'@\040localhost'\040IDENTIFIED\040BY\040'password';
GRANT\040ALL\040PRIVILEGES\040ON\040*.*\040TO\040'user'@\040localhost'\040WITH\040GRANT\040OPTION;
FLUSH\040PRIVILEGES;
EXIT;
```

-Observamos que hay un usuario que es capaz de obtener los permisos y no solo eso poder dárseles a futuros usuarios, `"user@localhost"` (GRANT ALL PRIVILEGES).

➤ WORDPRESS

Exploramos a través de WordPress con el comando “cd /var/www/html “, en el cual indagamos y nos fijamos en los permisos excesivos que tiene, esto es una clara vulnerabilidad ya que cualquier usuario es capaz de modificarlos.

```
lebian@debian:/var/www/html$ ls -l
total 244
-rwxrwxrwx 1 www-data www-data 10701 Sep 30 2024 index.html
-rwxrwxrwx 1 www-data www-data 405 Feb 6 2020 index.php
-rwxrwxrwx 1 www-data www-data 19915 Dec 31 2023 license.txt
-rwxrwxrwx 1 www-data www-data 7409 Jun 18 2024 readme.html
-rwxrwxrwx 1 www-data www-data 7387 Feb 13 2024 wp-activate.php
-rwxrwxrwx 9 www-data www-data 4096 Sep 10 2024 wp-admin/
-rwxrwxrwx 1 www-data www-data 351 Feb 6 2020 wp-blog-header.php
-rwxrwxrwx 1 www-data www-data 2323 Jun 14 2023 wp-comments-post.php
-rwxrwxrwx 1 www-data www-data 3017 Sep 30 2024 wp-config.php
-rwxrwxrwx 5 www-data www-data 4096 Oct 8 2024 wp-content/
-rwxrwxrwx 1 www-data www-data 5638 May 30 2023 wp-cron.php
-rwxrwxrwx 30 www-data www-data 12288 Sep 10 2024 wp-includes/
-rwxrwxrwx 1 www-data www-data 2502 Nov 26 2022 wp-links-opml.php
-rwxrwxrwx 1 www-data www-data 3937 Mar 11 2024 wp-load.php
-rwxrwxrwx 1 www-data www-data 51238 May 28 2024 wp-login.php
-rwxrwxrwx 1 www-data www-data 8525 Sep 16 2023 wp-mail.php
-rwxrwxrwx 1 www-data www-data 28774 Jul 9 2024 wp-settings.php
-rwxrwxrwx 1 www-data www-data 34385 Jun 19 2023 wp-signup.php
-rwxrwxrwx 1 www-data www-data 4885 Jun 22 2023 wp-trackback.php
-rwxrwxrwx 1 www-data www-data 3246 Mar 2 2024 xmlrpc.php
```

-Al tener tal acceso a esos servicios debemos actuar rápido y sobre todo en el archivo **wp-config.php**, en la cuales se encuentran las credenciales que comprometen el servicio.

```
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** Database username */
define( 'DB_USER', 'wordpressuser' );

/** Database password */
define( 'DB_PASSWORD', '123456' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );
```

-Accedemos al archivo y siguen las contraseñas por defecto. `wordpressuser(123456)`.

PARTE 2: MITIGACION

1. BLOQUEOS DE IPS

Usamos el firewall iptables y bloqueamos la ip conseguida anteriormente con el comando

“sudo iptables -A INPUT -s 192.168.0.134 -j DROP”

```
debian@debian:~$ sudo iptables -A INPUT -s 192.168.0.134 -j DROP
debian@debian:~$ sudo iptables-save
```

2. CONFIGURACION SSH

Entramos en “sudo nano /etc/ssh/sshd_config”

Cambiar PermitRootlogin no

```
#LoginGraceTime 2m
PermitRootLogin no
```

Cambiar PasswordAuthentication no

```
# To disable tunnelled clear text passwords, change to no here!
PasswordAuthentication no
```

3. MYSQL

Eliminamos al usuario antes mencionado con los privilegios elevados

```
debian@debian:~$ sudo mysql -e "DROP USER 'mysql'@'localhost';"
```

4. WORDPRESS

Cambiamos los permisos de los archivos vulnerables "sudo chmod 600 /var/www/html/wp-config.php

```
debian@debian:~$ sudo chmod 600 /var/www/html/wp-config.php
debian@debian:~$ ls -l /var/www/html
total 244
-rwxrwxrwx 1 www-data www-data 10701 Sep 30 2024 index.html
-rwxrwxrwx 1 www-data www-data 405 Feb 6 2020 index.php
-rwxrwxrwx 1 www-data www-data 19915 Dec 31 2023 license.txt
-rwxrwxrwx 1 www-data www-data 7409 Jun 18 2024 readme.html
-rwxrwxrwx 1 www-data www-data 7387 Feb 13 2024 wp-activate.php
drwxrwxrwx 9 www-data www-data 4096 Sep 10 2024 wp-admin
-rwxrwxrwx 1 www-data www-data 351 Feb 6 2020 wp-blog-header.php
-rwxrwxrwx 1 www-data www-data 2323 Jun 14 2023 wp-comments-post.php
-rw----- 1 www-data www-data 3017 Sep 30 2024 wp-config.php
```

5. FTP

Cambiamos el anónimo "yes" por el "no"

"Sudo nano /etc/vsftpd.conf"

```
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=NO
```

ACTUALIZACION

Sudo apt update

Sudo apt upgrade -y

➤ RECOMENDACIONES

SSH

1. Cambia el puerto predeterminado (22) → Reduce ataques automatizados.
2. Deshabilita el acceso root y usa claves SSH → Evita fuerza bruta y accesos no autorizados.

WordPress

1. Protege wp-config.php con permisos estrictos (600) → Bloquea robo de credenciales.
2. Restringe /wp-admin/ por IP → Solo usuarios autorizados pueden acceder al panel.

FTP

1. Usa SFTP (SSH) en lugar de FTP → Cifra la transferencia de archivos.
2. Habilita SSL/TLS si FTP es obligatorio → Evita credenciales en texto plano.

MySQL

1. Reduce privilegios de usuarios (no usar GRANT ALL) → Limita daños por ataques SQL.
 2. Cambia el puerto predeterminado (3306) → Dificulta escaneos automatizados.
- Firewall (bloquea puertos innecesarios).
 - Actualiza siempre el sistema cuando sea posible.

Fase II: DETECCION Y CORRECCION VULNERABILIDAD.

INTRODUCCION

En este ejercicio, realizaremos una prueba de penetración controlada contra una máquina virtual Debian, identificando posibles vulnerabilidades en servicios como SSH, FTP y Apache desde una

máquina Kali Linux. Utilizaremos herramientas como Nmap, Metasploit y WPScan para descubrir y explotar fallos de seguridad.

OBJETIVOS

El objetivo es evaluar la exposición a ataques y aplicar medidas de mitigación. Todo se ejecutará en un entorno aislado para garantizar la legalidad y ética del proceso.

➤ **DETECCION DE VULNERABILIDADES**

Realizamos desde nuestra maquina Kali un escaneo con la herramienta Nmap con el comando

`"Nmap -sV -p- 192.168.1.156"`.

Lo cual nos arroja como resultado ciertas vulnerabilidades.

```
(kali@kali)-[~]
$ nmap -sV -p- 192.168.1.156
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-29 17:51 EDT
Nmap scan report for 192.168.1.156
Host is up (0.00025s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.62 ((Debian))
MAC Address: 08:00:27:59:6C:E8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
229 Entering Extended Passive Mode (||-54551|)
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.61 seconds
```

➤ **EXPLICACION ESCANEO Nmap**

Al observar detalladamente hemos detectado ciertas vulnerabilidades

- **Ftp version vsftpd 3.0.3**

CVE-2015-1419: DoS con SIZE en symlinks → Actualizar.

CVE-2011-0762: SSL mal configurado → Deshabilitar force_local_data_ssl.

Anónimo habilitado → anonymous_enable=NO.

Fuerza bruta → Usar fail2ban.

TLS obsoleto → Forzar TLS 1.2+.

Permisos laxos → chroot_local_user=YES.

Mitigación final: Actualizar y restringir configs.

- **Ssh version OpenSSH 9.2p1**

CVE-2023-25136 (DoS en LibSSH) → Actualizar a 9.3p1+.

CVE-2021-41617 (bypass en sandbox) → Usar RestrictAddressFamilies.

Fuerza bruta → Habilitar MaxAuthTries 3 y fail2ban.

Claves obsoletas → Deshabilitar SHA1/RSA débiles (KexAlgorithms modernos).

Escalada vía authorized_keys → Usar Restrict y command=.

Privilegios elevados → Ejecutar con Privilege Separation.

Mitigación final: Actualizar y aplicar configuraciones restrictivas.

Nota: (OpenSSH 9.2p1 es relativamente seguro, pero requiere hardening).

- **Http version Apache HTTPD 2.4.62**

CVE-2023-45802 (DoS en HTTP/2) → Desactivar h2 si no es necesario o actualizar.

CVE-2023-43622 (mod_macro overflow) → Parchear o actualizar a 2.4.63+.

Exposición de información → Deshabilitar ServerTokens Prod y Options -Indexes.

Ataques Slowloris → Usar mod_reqtimeout y LimitRequestFields.

Configs inseguras → Evitar AllowOverride All y restringir permisos.

TLS débil → Forzar TLS 1.2+ y cifrados fuertes (SSLProtocol).

Mitigación final: Actualizar, minimizar módulos y aplicar hardening.

Nota:*(Apache 2.4.62 es estable, pero requiere configuración segura)*

➤ EXPLOTACION VULNERABILIDAD

Realizaremos un ataque que es uno de los más comunes por su facilidad para ser aplicado

Objetivo: Saturar un servidor web (puerto 80) con tráfico masivo desde Kali Linux hacia una máquina Debian en un entorno controlado.

"sudo hping3 --flood -S -p 80 --rand-source 192.168.1.156"

```
(kali@kali) [~]
$ sudo hping3 --flood -S -p 80 --rand-source 192.168.1.156
HPING 192.168.1.156 (eth0 192.168.1.156): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

"Podemos ser un poco más eficientes si usamos un ++80 ya que con esto incrementamos dinámicamente el puerto destino es decir ataca el puerto 81, 82, 83 y así sucesivamente lo cual tiene como propósito evadir firewall y saturar múltiples puertos."

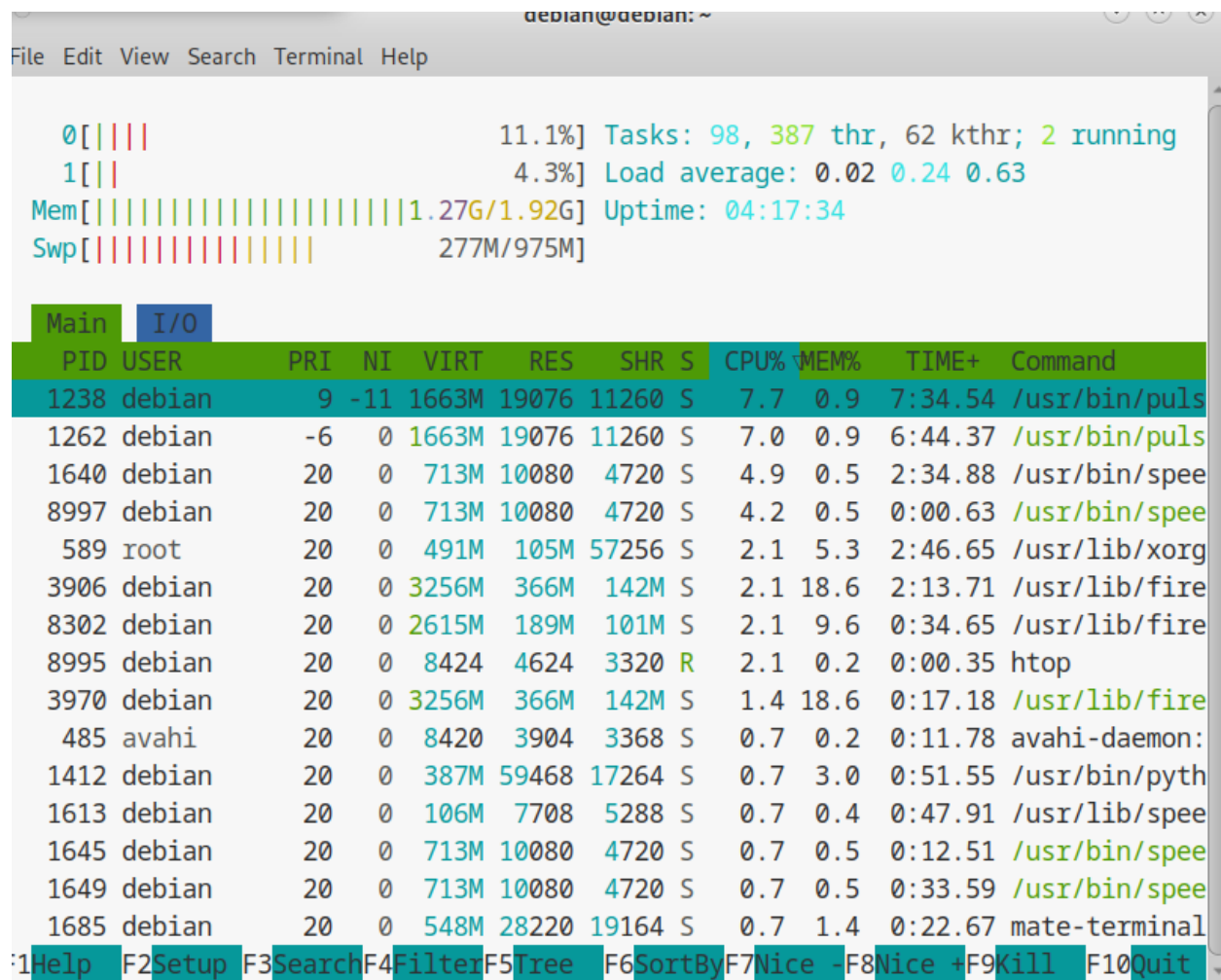
"sudo hping3 --flood -S -p ++80 --rand-source 192.168.1.156"

```
(kali@kali) [~]
$ sudo hping3 --flood -S -p ++80 --rand-source 192.168.1.156
HPING 192.168.1.156 (eth0 192.168.1.156): S set, 40 headers + 0 data bytes
```

Una vez enviada las peticiones abrimos el “htop” el cual es un panel que nos muestra el rendimiento y procesos que se están ejecutando en la maquina Debian.

```
debian@debian:~$ htop
```

Un panel normal se mostraría de la siguiente manera,



The screenshot shows the htop interface on a Debian system. At the top, there are system statistics: CPU usage at 11.1%, 387 threads, 62 kthreads, and 2 running tasks. Load averages are 0.02, 0.24, and 0.63. Memory usage is 1.27G/1.92G, and swap usage is 277M/975M. The uptime is 04:17:34. Below the statistics, there are tabs for 'Main' and 'I/O'. The 'Main' tab is selected, displaying a table of running processes. The table has columns for PID, USER, PRI, NI, VIRT, RES, SHR, S, CPU%, MEM%, TIME+, and Command. The processes listed include various system daemons and user processes, with the htop process itself at PID 8995.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
1238	debian	9	-11	1663M	19076	11260	S	7.7	0.9	7:34.54	/usr/bin/puls
1262	debian	-6	0	1663M	19076	11260	S	7.0	0.9	6:44.37	/usr/bin/puls
1640	debian	20	0	713M	10080	4720	S	4.9	0.5	2:34.88	/usr/bin/spee
8997	debian	20	0	713M	10080	4720	S	4.2	0.5	0:00.63	/usr/bin/spee
589	root	20	0	491M	105M	57256	S	2.1	5.3	2:46.65	/usr/lib/xorg
3906	debian	20	0	3256M	366M	142M	S	2.1	18.6	2:13.71	/usr/lib/fire
8302	debian	20	0	2615M	189M	101M	S	2.1	9.6	0:34.65	/usr/lib/fire
8995	debian	20	0	8424	4624	3320	R	2.1	0.2	0:00.35	htop
3970	debian	20	0	3256M	366M	142M	S	1.4	18.6	0:17.18	/usr/lib/fire
485	avahi	20	0	8420	3904	3368	S	0.7	0.2	0:11.78	avahi-daemon:
1412	debian	20	0	387M	59468	17264	S	0.7	3.0	0:51.55	/usr/bin/pyth
1613	debian	20	0	106M	7708	5288	S	0.7	0.4	0:47.91	/usr/lib/spee
1645	debian	20	0	713M	10080	4720	S	0.7	0.5	0:12.51	/usr/bin/spee
1649	debian	20	0	713M	10080	4720	S	0.7	0.5	0:33.59	/usr/bin/spee
1685	debian	20	0	548M	28220	19164	S	0.7	1.4	0:22.67	mate-terminal

At the bottom of the window, there is a footer bar with function key shortcuts: F1 Help, F2 Setup, F3 Search, F4 Filter, F5 Tree, F6 SortBy, F7 Nice, F8 Nice, F9 Kill, and F10 Quit.

Luego de lanzar el hping3 desde la kali nos muestra la saturación en el cpu y se evidencia la lentitud en la maquina y su rendimiento.


```
debian@debian: ~  
File Edit View Search Terminal Help  
  
0[||||||| 18.1%] Tasks: 81, 134 thr, 61 kthr; 2 running  
1[||||||| 100.0%] Load average: 0.56 0.33 0.15  
Mem[||||||| 825M/1.92G] Uptime: 03:30:16  
Swp[ 0K/975M]  
  
Main I/O  
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command  
1640 debian 20 0 713M 16548 5660 S 6.0 0.8 1:00.83 /usr/bin/spee  
1238 debian 9 -11 1663M 32304 23116 S 5.3 1.6 5:02.22 /usr/bin/puls  
1262 debian -6 0 1663M 32304 23116 S 4.7 1.6 4:38.10 /usr/bin/puls  
1613 debian 20 0 106M 12312 6596 S 4.7 0.6 0:18.00 /usr/lib/spee  
3712 debian 20 0 713M 16548 5660 S 4.7 0.8 0:00.34 /usr/bin/spee  
589 root 20 0 445M 131M 73004 S 1.3 6.7 0:50.45 /usr/lib/xorg  
1412 debian 20 0 379M 71064 32832 S 0.7 3.5 0:11.46 /usr/bin/pyth  
1649 debian 20 0 713M 16548 5660 S 0.7 0.8 0:15.76 /usr/bin/spee  
3222 debian 20 0 8036 4364 3420 R 0.7 0.2 0:07.06 htop  
1 root 20 0 164M 12408 9080 S 0.0 0.6 0:00.96 /sbin/init sp  
240 root 20 0 49604 20176 14796 S 0.0 1.0 0:00.35 /lib/systemd/  
275 root 20 0 27544 7016 4680 S 0.0 0.3 0:00.13 /lib/systemd/  
284 systemd-ti 20 0 90104 6684 5760 S 0.0 0.3 0:00.16 /lib/systemd/  
483 systemd-ti 20 0 90104 6684 5760 S 0.0 0.3 0:00.00 /lib/systemd/  
484 root 20 0 231M 9468 6512 S 0.0 0.5 0:00.37 /usr/libexec/  
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice + F9Kill F10Quit
```

Y luego,

```
debian@debian: ~  
File Edit View Search Terminal Help  
  
0[||||||| 91.9%] Tasks: 93, 341 thr, 64 kthr; 2 running  
1[||||||| 94.8%] Load average: 1.78 1.11 0.82  
Mem[||||||| 1.20G/1.92G] Uptime: 03:51:21  
Swp[|| 29.0M/975M]  
  
Main I/O  
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command  
3906 debian 20 0 3078M 423M 202M R 45.8 21.5 0:45.50 /usr/lib/fir  
5645 debian 20 0 2398M 145M 102M R 37.1 7.4 0:01.56 /usr/lib/fir  
589 root 20 0 459M 143M 80696 R 30.7 7.3 1:57.09 /usr/lib/xor  
3972 debian 20 0 3078M 423M 202M S 9.9 21.5 0:05.78 /usr/lib/fir  
1412 debian 20 0 385M 75348 31776 S 6.4 3.7 0:35.28 /usr/bin/pyt  
1238 debian 9 -11 1663M 23972 14784 S 5.8 1.2 6:23.38 /usr/bin/pul  
3971 debian 20 0 3078M 423M 202M S 4.6 21.5 0:00.08 /usr/lib/fir  
1262 debian -6 0 1663M 23972 14784 S 4.1 1.2 5:44.35 /usr/bin/pul
```

Observamos una saturacion en el cpu y rendimiento de la maquina en la cual comprobamos que el ataque mediante DDoS ha sido efectivo.

➤ MEDIDAS DE PREVENCIÓN CONTRA DDoS

Aunque nunca se exento al 100% de ataques de DDoS con estas medidas podemos protegernos y así evitar ser tan vulnerable para el atacante.

Aplicamos unas reglas en el firewall que tengamos instalado en este caso el “**iptables**”

```
“sudo iptables -A INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 3 -j ACCEPT”
```

```
“sudo iptables -A INPUT -p tcp --syn -j DROP”
```

```
“sudo iptables -A INPUT -p tcp --dport 80 -m connlimit --connlimit-above 50 -j DROP”
```

```
debian@debian:~$ sudo iptables -A INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 3 -j ACCEPT
[sudo] password for debian:
debian@debian:~$ sudo iptables -A INPUT -p tcp --syn -j DROP
debian@debian:~$ sudo iptables -A INPUT -p tcp --dport 80 -m connlimit-above 50 -j DROP
iptables v1.8.9 (nf_tables): Couldn't load match 'connlimit-above':No such file or directory

Try `iptables -h' or 'iptables --help' for more information.
debian@debian:~$ sudo iptables -A INPUT -p tcp --dport 80 -m connlimit --connlimit-above 50 -j DROP
```

Hacemos las Reglas persistentes con

```
“sudo apt install iptables-persistent”
```

```
“sudo netfilter-persistent save”
```

FAIL2BAN

Como medida adicional descargamos fail2ban para banear ips maliciosas

```
“Sudo apt install fail2ban”
```

Creamos una copia de jail.conf

“sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local”

Y escribimos

```
[sshd]
enabled = true
maxretry = 3

[http-get-dos]
enabled = true
port = http,https
filter = http-get-dos
logpath = /var/log/apache2/access.log
maxretry = 100
findtime = 60
bantime = 3600
```



Importante

Ninguna medida es 100% efectiva contra un ataque DDoS masivo.

FASE III: RESPUESTA ANTE INCIDENTES Y CERTIFICACION.

➤ **Plan de Respuesta a Incidentes (NIST SP 800-61)**

Contexto: Un plan estructurado permite reaccionar rápidamente ante brechas de seguridad, minimizando daños y tiempo de inactividad.

1. Identificación: Implementar herramientas como SIEM (Splunk, Wazuh) para correlacionar eventos en tiempo real. Ejemplo: Alertas automáticas al detectar múltiples intentos de login fallidos en SSH.
2. Contención:
 - a. A corto plazo: Bloquear IPs atacantes con iptables o fail2ban.
 - b. A largo plazo: Aislar segmentos de red comprometidos (VLANs separadas).
3. Erradicación: Eliminar backdoors encontrados (ej: analizar procesos sospechosos con ps aux). Actualizar servicios vulnerables (ej: sudo apt update && sudo apt upgrade openssl).
4. Recuperación: Restaurar sistemas desde backups cifrados previamente probados. Realizar pruebas de penetración post-recuperación para asegurar que no quedan vulnerabilidades.

➤ **Respuesta a un Ataque Similar**

Contexto: Basado en un supuesto ataque DDoS o intrusión vía SSH.

1. Respuesta Inmediata: Activar el equipo de respuesta a incidentes (CSIRT) y notificar a stakeholders. Ejemplo: Si el ataque es DDoS, redirigir tráfico a un servicio de mitigación (Cloudflare).
2. Forensia Digital: Analizar logs (/var/log/auth.log, /var/log/apache2/access.log) para identificar patrones. Usar herramientas como Autopsy o The Sleuth Kit para análisis forense en discos.

3. Prevención: Implementar listas de control de acceso (ACLs) en firewalls. Ejemplo:
Restringir SSH solo a IPs corporativas con iptables `-A INPUT -p tcp --dport 22 -s 192.168.1.0/24 -j ACCEPT`.
4. Concientización: Simulacros de phishing mensuales y talleres sobre contraseñas seguras.

➤ **Protección de Datos**

Contexto: Garantizar la integridad y confidencialidad de información crítica.

1. Backups Automatizados: Usar BorgBackup para backups incrementales y cifrados, almacenados en un servidor offline. Ejemplo: `borg create /backup::$(date +%Y-%m-%d) /datos_criticos`.
2. Cifrado:
 - a. En tránsito: TLS 1.3 para comunicaciones internas (ej: configurar en Nginx/Apache).
 - b. En reposo: LUKS para discos y gpg para archivos sensibles.
3. Controles de Acceso: MFA obligatorio en todos los sistemas (ej: Google Authenticator + contraseñas). Roles basados en RBAC (ej: sudo solo para administradores).
4. Retención de Logs: Centralizar logs con ELK Stack (Elasticsearch, Logstash, Kibana).
Política de retención: 90 días para logs de acceso, 1 año para logs de autenticación.

➤ **Implementación de SGSI (ISO 27001)**

Contexto: Alinear procesos de seguridad con estándares internacionales.

1. Análisis de Riesgos: Usar OCTAVE Allegro para evaluar amenazas a activos críticos (ej: servidores de BD). Priorizar riesgos con matrices de impacto/probabilidad.
2. Políticas de Seguridad: Documentar políticas para:

- a. Acceso remoto (ej: VPN con certificados).
 - b. BYOD (ej: dispositivos deben tener cifrado activo).
- 3. Planes de Acción: Parcheo crítico en 24 horas (ej: vulnerabilidades Zero-Day). Ejemplo:
sudo apt --only-upgrade install openssl ante un CVE crítico.
- 4. Auditorías Periódicas: Escaneos trimestrales con OpenVAS para detectar vulnerabilidades.
Cumplimiento de regulaciones locales (ej: GDPR o PDPL en Latinoamérica).

CONCLUSION

El análisis forense realizado en la máquina Debian comprometida reveló múltiples vectores de ataque explotados por intrusos, destacando configuraciones inseguras en servicios clave:

1. SSH: Se detectaron intentos de fuerza bruta exitosos debido a contraseñas débiles y falta de autenticación en dos factores. Se implementó Fail2Ban y se restringió el acceso por IP.
2. FTP (vsftpd): El servicio permitía acceso anónimo y transferencias sin cifrado, facilitando la filtración de datos. Se migró a SFTP con autenticación por claves.
3. HTTP (Apache): Directorios web listables y versiones obsoletas expusieron información sensible. Se aplicó hardening con directivas como Options -Indexes y WAF (ModSecurity).

Para garantizar cumplimiento normativo, se adoptaron:

- ISO/IEC 27001: Se documentaron controles de acceso (A.9.4.1) y gestión de vulnerabilidades (A.12.6.1).
- NIST SP 800-61: Se estableció un plan de respuesta a incidentes con etapas de contención (aislamiento de servicios) y recuperación (restauración desde backups cifrados).
- Protección de datos: Se cifraron archivos sensibles (LUKS) y se auditó el cumplimiento del RGPD (registros de acceso, consentimiento).

Este caso subraya la necesidad de:

- Monitoreo continuo (SIEM como ELK Stack).
- Parcheo ágil (actualizaciones automáticas con unattended-upgrades).

- Concienciación del personal (simulacros de phishing).

Lección clave: La seguridad requiere un enfoque estratificado (defensa en profundidad), combinando tecnología, normativas y formación.