

Reto Día 8: Gestión de Servicios y Automatización con Systemd en Linux

Por Daniel Ariza

19/06/2025

Fase 1: Análisis de servicios del sistema

- Listar todos los servicios activos del sistema usando **systemctl list-units --type=service**.

Para realizar esto ejecutaremos este comando

systemctl list-units --type=service --state=running

que nos mostrará los servicios que tenemos en ejecución.

```
daniel@ubuntu:~$ systemctl list-units --type=service --state=running
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
accounts-daemon.service            loaded active running Accounts Service
apache2.service                     loaded active running LSB: Apache2 web server
cron.service                       loaded active running Regular background program processing daemon
dbus.service                       loaded active running D-Bus System Message Bus
getty@tty1.service                 loaded active running Getty on tty1
rsyslog.service                    loaded active running System Logging Service
ssh.service                         loaded active running OpenBSD Secure Shell server
systemd-journald.service            loaded active running Journal Service
systemd-logind.service              loaded active running Login Service
systemd-udevd.service               loaded active running udev Kernel Device Manager
user@1000.service                   loaded active running User Manager for UID 1000
vboxadd-service.service             loaded active running vboxadd-service.service

LOAD    = Reflects whether the unit definition was properly loaded.
ACTIVE  = The high-level unit activation state, i.e. generalization of SUB.
SUB     = The low-level unit activation state, values depend on unit type.

12 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
daniel@ubuntu:~$
```

- Identificar y documentar 3 servicios activos fundamentales (ej.: ssh, cron, networking).

1. **SSH service:** Permite conexiones remotas seguras mediante el protocolo SSH. Es esencial para la administración remota del servidor sin necesidad de entorno gráfico. Es crítico para el acceso remoto seguro, especialmente en servidores virtuales o en la nube.
2. **cron.service:** Ejecuta tareas programadas de forma automática. Permite definir tareas recurrentes (ej. backups, limpieza de logs, sincronizaciones). Es fundamental para la automatización de tareas administrativas del sistema.
3. **systemd-networkd.service:** Aunque no se ve explícitamente en la captura, el servicio de red es siempre esencial. En este caso, puede estar bajo otro nombre o gestionado por [netplan](#) directamente.

● Comprobar si el servidor web (instalado el día anterior) está activo, habilitado y funcionando.

Con este comando comprobamos si está activo `systemctl status apache2`

```
daniel@ubuntucodearts:~$ systemctl status apache2
* apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad: vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
            '-apache2-systemd.conf
   Active: active (running) since vie 2025-06-20 07:10:02 CEST; 31min ago
     Docs: man:systemd-sysv-generator(8)
  Process: 773 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
    CGroup: /system.slice/apache2.service
            └─981 /usr/sbin/apache2 -k start
            └─984 /usr/sbin/apache2 -k start
            └─985 /usr/sbin/apache2 -k start
daniel@ubuntucodearts:~$ _
```

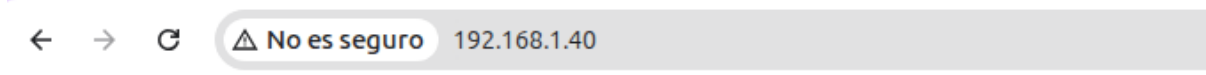
Vemos que si lo está.

Con este otro vemos si está habilitado al arrancar el sistema `systemctl is-enabled apache2`

```
daniel@ubuntucodearts:~$ systemctl is-enabled apache2
apache2.service is not a native service, redirecting to systemd-sysv-install
Executing /lib/systemd/systemd-sysv-install is-enabled apache2
enabled
daniel@ubuntucodearts:~$ _
```

La palabra **enabled** nos indica que si lo está.

Para comprobar si está funcionando correctamente introducimos la ip de nuestro servidor virtualizado en el navegador web de otro equipo, en este caso desde mi equipo real.



Servidor Apache funcionando correctamente

Vemos que nos devuelve el mensaje que pusimos ayer, lo que nos confirma que funciona correctamente.

Fase 2: Gestión avanzada de servicios con Systemd

- **Detener, reiniciar y habilitar al arranque el servicio web apache2.**

Para detener el servicio apache2 utilizamos este comando

sudo systemctl stop apache2

```
[sudo] password for daniel:
daniel@ubuntucodearts:~$ systemctl status apache2
systemctl: no se puede efectuar stat sobre /proc/sys/status: No existe el archivo o el directorio
systemctl: no se puede efectuar stat sobre /proc/sys/apache2: No existe el archivo o el directorio
daniel@ubuntucodearts:~$ systemctl status apache2
* apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
            `--apache2-systemd.conf
   Active: inactive (dead) since vie 2025-06-20 07:57:11 CEST; 1min 7s ago
     Docs: man:systemd-sysv-generator(8)
   Process: 1202 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
   Process: 773 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
daniel@ubuntucodearts:~$
```

Vemos como se ha detenido verificando su estado con **systemctl status apache2**.

Para reiniciarlo utilizaremos el siguiente comando `sudo systemctl restart apache2`.

```
daniel@ubuntucodearts:~$ sudo systemctl restart apache2
sudo: systemctl: orden no encontrada
daniel@ubuntucodearts:~$ sudo systemctl restart apache2
daniel@ubuntucodearts:~$ systemctl status apache2
* apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
            -apache2-systemd.conf
   Active: active (running) since vie 2025-06-20 08:02:05 CEST; 21s ago
     Docs: man:systemd-sysv-generator(8)
   Process: 1202 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
   Process: 1238 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
    CGroup: /system.slice/apache2.service
            └─1254 /usr/sbin/apache2 -k start
              └─1257 /usr/sbin/apache2 -k start
                └─1258 /usr/sbin/apache2 -k start
daniel@ubuntucodearts:~$ _
```

Comprobamos que se ha reiniciado correctamente.

Y con este comando habilitaremos Apache2 al arranque del sistema `sudo systemctl enable apache2`.

```
daniel@ubuntucodearts:~$ sudo systemctl enable apache2
apache2.service is not a native service, redirecting to systemd-sysv-install
Executing /lib/systemd/systemd-sysv-install enable apache2
daniel@ubuntucodearts:~$ systemctl is-enabled apache2
No se ha encontrado la orden «systemctl», quizás quiso decir:
La orden «systemctl» del paquete «systemd» (main)
systemctl: no se encontró la orden
daniel@ubuntucodearts:~$ systemctl is-enabled apache2
apache2.service is not a native service, redirecting to systemd-sysv-install
Executing /lib/systemd/systemd-sysv-install is-enabled apache2
enabled
daniel@ubuntucodearts:~$ _
```

Como podemos ver la palabra `enabled` lo confirma.

- **Modificar la configuración de uno de los servicios para que se reinicie automáticamente si falla (Restart=always en su .service).**

Utilizaremos en el ejemplo Apache2, primero creamos un archivo de sobrecarga (override) para Apache2 esto permite modificar su configuración sin tocar el archivo original del sistema.

Ejecutamos `sudo systemctl edit apache2` Esto abre un editor vacío en el que incluiremos estas líneas: `[Service]`

`Restart=always` reinicia el servicio incluso si falla.

`RestartSec=5` espera 5 segundos antes de reiniciarlo.

Guardamos y cerramos el editor.

```
daniel@ubuntucodearts:~$ sudo systemctl daemon-reexec
daniel@ubuntucodearts:~$ sudo systemctl daemon-reload
daniel@ubuntucodearts:~$ sudo systemctl restart apache2
daniel@ubuntucodearts:~$
```

Y con estos comandos recargamos systemd y reiniciamos Apache2.

Ahora verificamos la configuración aplicada con

`systemctl cat apache2`

```

# /etc/systemd/system/apache2.service.d/override.conf
[Service]
Restart=always
RestartSec=5
lines 5-40/40 (END)
```

Aquí vemos que el archivo está en uso.

● Crear un alias para reiniciar rápidamente el servicio desde `.bashrc` o `.zshrc`.

Crear un alias nos permite ejecutar comandos largos con una palabra corta, crearemos uno par reiniciar Apache2 fácilmente con:

`alias rapache='sudo systemctl restart apache2'`

Primero abrimos el archivo `.bashrc` con un editor usando

`nano ~/.bashrc`

Vamos al final del archivo y escribimos esta línea:

`alias rapache='sudo systemctl restart apache2'`

```
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
alias rapache='sudo systemctl restart apache2'_
```

Guardamos y cerramos el archivo.

Con `source ~/.bashrc` aplicamos los cambios.
Ahora al escribir la palabra `rapache` Apache2 se reiniciará.

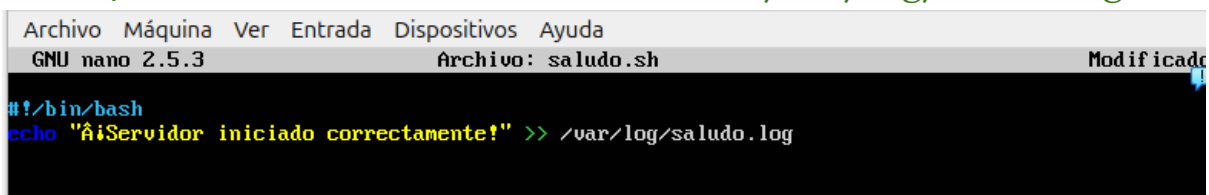
Fase 3: Creación de un servicio personalizado

- **Crear un script Bash llamado `saludo.sh` que escriba "¡Servidor iniciado correctamente!" en un archivo `/var/log/saludo.log`.**

En primer lugar crearemos el archivo `saludo.sh` con el comando `nano saludo.sh` y escribiremos dentro este contenido:

```
#!/bin/bash
```

```
echo "¡Servidor iniciado correctamente!" >> /var/log/saludo.log
```

A screenshot of the nano text editor interface. The top bar shows 'Archivo', 'Máquina', 'Ver', 'Entrada', 'Dispositivos', 'Ayuda', 'GNU nano 2.5.3', 'Archivo: saludo.sh', and 'Modificado'. The main area shows the script content:

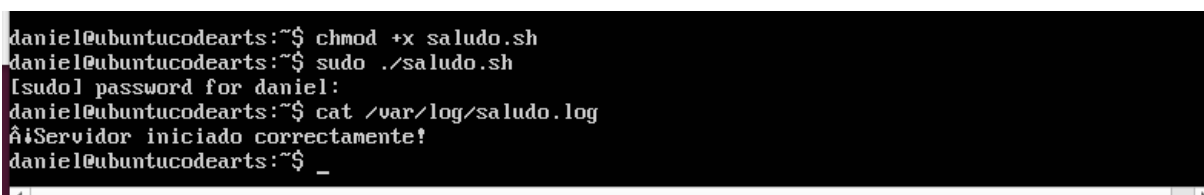
```
#!/bin/bash
echo "¡Servidor iniciado correctamente!" >> /var/log/saludo.log
```

Guardamos y cerramos el archivo.

Ahora le daremos permisos de ejecución con `chmod +x saludo.sh`

Y ejecutaremos el script con `sudo ./saludo.sh`

Verificamos que se escribió el mensaje con `cat /var/log/saludo.log`

A screenshot of a terminal window. The commands and output are:

```
daniel@ubuntucodearts:~$ chmod +x saludo.sh
daniel@ubuntucodearts:~$ sudo ./saludo.sh
[sudo] password for daniel:
daniel@ubuntucodearts:~$ cat /var/log/saludo.log
¡Servidor iniciado correctamente!
daniel@ubuntucodearts:~$ _
```

- **Crear un nuevo servicio de systemd llamado `saludo.service` que ejecute ese script automáticamente al iniciar el sistema.**

Primero movemos el script a una ruta que esté siempre disponible

```
sudo mv saludo.sh /usr/local/bin/saludo.sh
```

```
sudo chmod +x /usr/local/bin/saludo.sh
```

y le damos permiso de ejecución.

```
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
daniel@ubuntucodearts:~$ sudo mv saludo.sh /usr/local/bin/saludo.sh
daniel@ubuntucodearts:~$ sudo chmod +x /usr/local/bin/saludo.sh
daniel@ubuntucodearts:~$
```

Ahora crearemos el archivo del servicio con
`sudo nano /etc/systemd/system/saludo.service`

Y escribiremos dentro este contenido:

[Unit]

Description=Mensaje de saludo al iniciar el sistema

After=network.target

[Service]

Type=oneshot

ExecStart=/usr/local/bin/saludo.sh

[Install]

WantedBy=multi-user.target

```
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 2.5.3      Archivo: /etc/systemd/system/saludo.service  Modificado
[Unit]
Description=Mensaje de saludo al iniciar el sistema
After=network.target

[Service]
Type=oneshot
ExecStart=/usr/local/bin/saludo.sh

[Install]
WantedBy=multi-user.target_
```

Guardamos los cambios y cerramos el archivo.

Ahora recargaremos systemd para registrar el nuevo servicio
ejecutando

`sudo systemctl daemon-reload`

Y habilitaremos el servicio para que se ejecute al arrancar
`sudo systemctl enable saludo.service`

```
daniel@ubuntucodearts:~$ sudo systemctl daemon-reload
[sudo] password for daniel:
daniel@ubuntucodearts:~$ sudo systemctl enable saludo.service
Created symlink from /etc/systemd/system/multi-user.target.wants/saludo.service to /etc/systemd/system/saludo.service.
daniel@ubuntucodearts:~$
```

Probaremos el servicio manualmente con

`sudo systemctl start saludo.service`

Y verificamos el contenido del archivo log con

`cat /var/log/saludo.log`

```
em/saludo.service.
daniel@ubuntucodearts:~$ sudo systemctl start saludo.service
daniel@ubuntucodearts:~$ cat /var/log/saludo.log
^iServidor iniciado correctamente!
^iServidor iniciado correctamente!
daniel@ubuntucodearts:~$ _
```

- **Comprobar que el servicio:**
 - **Está habilitado**

Ejecutamos `systemctl is-enabled saludo.service`

```
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
daniel@ubuntucodearts:~$ systemctl is-enabled saludo.service
enabled
daniel@ubuntucodearts:~$ _
```

La

palabra **enabled** nos indica que está habilitado.

- **Se ejecuta al arrancar**

Con `systemctl list-unit-files | grep saludo` revisamos si está programado para iniciarse.

```
enabled
daniel@ubuntucodearts:~$ systemctl list-unit-files | grep saludo
saludo.service                                enabled
daniel@ubuntucodearts:~$
```

enabled

nos confirma que sí.

- **Crea el archivo de log correctamente**

Verificamos si existe y su contenido con

`ls -l /var/log/saludo.log` y luego `cat /var/log/saludo.log`

```
saludo.service                                enabled
daniel@ubuntucodearts:~$ ls -l /var/log/saludo.log
-rw-r--r-- 1 root root 70 jun 20 11:04 /var/log/saludo.log
daniel@ubuntucodearts:~$ cat /var/log/saludo.log
!Servidor iniciado correctamente!
!Servidor iniciado correctamente!
daniel@ubuntucodearts:~$ _
```

Podemos ver que está todo correcto.

Fase 4: Monitorización y logs

- **Visualizar los logs de los servicios anteriores con `journalctl`.**

Ejecutamos `sudo journalctl -u apache2` para ver los logs de Apache2.

```
daniel@ubuntucodearts:~$ sudo journalctl -u apache2
[sudo] password for daniel:
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 11:40:02 CEST. --
jun 20 07:09:59 ubuntucodearts systemd[1]: Starting LSB: Apache2 web server...
jun 20 07:09:59 ubuntucodearts apache2[7731]: * Starting Apache httpd web server apache2
jun 20 07:10:00 ubuntucodearts apache2[7731]: AH00558: apache2: Could not reliably determine the se
jun 20 07:10:02 ubuntucodearts apache2[7731]: *
jun 20 07:10:02 ubuntucodearts systemd[1]: Started LSB: Apache2 web server.
jun 20 07:57:10 ubuntucodearts systemd[1]: Stopping LSB: Apache2 web server...
jun 20 07:57:10 ubuntucodearts apache2[12021]: * Stopping Apache httpd web server apache2
jun 20 07:57:11 ubuntucodearts apache2[12021]: *
jun 20 07:57:11 ubuntucodearts systemd[1]: Stopped LSB: Apache2 web server.
jun 20 08:02:03 ubuntucodearts systemd[1]: Stopped LSB: Apache2 web server.
jun 20 08:02:03 ubuntucodearts systemd[1]: Starting LSB: Apache2 web server...
jun 20 08:02:03 ubuntucodearts apache2[12381]: * Starting Apache httpd web server apache2
jun 20 08:02:04 ubuntucodearts apache2[12381]: AH00558: apache2: Could not reliably determine the s
jun 20 08:02:05 ubuntucodearts apache2[12381]: *
jun 20 08:02:05 ubuntucodearts systemd[1]: Started LSB: Apache2 web server.
jun 20 08:34:16 ubuntucodearts systemd[1]: Stopping LSB: Apache2 web server...
jun 20 08:34:16 ubuntucodearts apache2[14221]: * Stopping Apache httpd web server apache2
jun 20 08:34:17 ubuntucodearts apache2[14221]: *
jun 20 08:34:17 ubuntucodearts systemd[1]: Stopped LSB: Apache2 web server.
jun 20 08:34:17 ubuntucodearts systemd[1]: Starting LSB: Apache2 web server...
jun 20 08:34:17 ubuntucodearts apache2[14541]: * Starting Apache httpd web server apache2
jun 20 08:34:17 ubuntucodearts apache2[14541]: AH00558: apache2: Could not reliably determine the s
jun 20 08:34:18 ubuntucodearts apache2[14541]: *
jun 20 08:34:18 ubuntucodearts systemd[1]: Started LSB: Apache2 web server.
jun 20 09:13:50 ubuntucodearts systemd[1]: Stopping LSB: Apache2 web server...
jun 20 09:13:51 ubuntucodearts apache2[15771]: * Stopping Apache httpd web server apache2
jun 20 09:13:51 ubuntucodearts apache2[15771]: *
jun 20 09:13:51 ubuntucodearts systemd[1]: Stopped LSB: Apache2 web server.
jun 20 09:13:51 ubuntucodearts systemd[1]: Starting LSB: Apache2 web server...
jun 20 09:13:51 ubuntucodearts apache2[16011]: * Starting Apache httpd web server apache2
jun 20 09:13:51 ubuntucodearts apache2[16011]: AH00558: apache2: Could not reliably determine the s
jun 20 09:13:52 ubuntucodearts apache2[16011]: *
jun 20 09:13:52 ubuntucodearts systemd[1]: Started LSB: Apache2 web server.
[Type 1 24 24] (END)
```

Para ver los logs de `saludo.service` ejecutamos `sudo journalctl -u saludo.service`

```
daniel@ubuntucodearts:~$ sudo journalctl -u saludo.service
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 11:42:12 CEST. --
jun 20 11:04:17 ubuntucodearts systemd[1]: Starting Mensaje de saludo al iniciar el sistema...
jun 20 11:04:17 ubuntucodearts systemd[1]: Started Mensaje de saludo al iniciar el sistema.
daniel@ubuntucodearts:~$ sudo journalctl -u saludo.service -r
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 11:42:43 CEST. --
jun 20 11:04:17 ubuntucodearts systemd[1]: Started Mensaje de saludo al iniciar el sistema.
jun 20 11:04:17 ubuntucodearts systemd[1]: Starting Mensaje de saludo al iniciar el sistema...
daniel@ubuntucodearts:~$
```

Y para ver los logs del arranque completo del sistema ejecutamos **sudo journalctl -b**

```
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 11:47:11 CEST. --
jun 20 07:09:56 ubuntucodearts systemd-journal[1951]: Runtime journal (/run/log/journal/) is 2.5M
jun 20 07:09:56 ubuntucodearts kernel: Initializing cgroup subsys cpuset
jun 20 07:09:56 ubuntucodearts kernel: Initializing cgroup subsys cpu
jun 20 07:09:56 ubuntucodearts kernel: Initializing cgroup subsys cpuacct
jun 20 07:09:56 ubuntucodearts kernel: Linux version 4.4.0-210-generic (build@lgw01-amd64-009) (g
jun 20 07:09:56 ubuntucodearts kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.4.0-210-generic ro
jun 20 07:09:56 ubuntucodearts kernel: KERNEL supported cpus:
jun 20 07:09:56 ubuntucodearts kernel:   Intel GenuineIntel
jun 20 07:09:56 ubuntucodearts kernel:   AMD AuthenticAMD
jun 20 07:09:56 ubuntucodearts kernel:   Centaur CentaurHauls
jun 20 07:09:56 ubuntucodearts kernel: tseg: 0000000000
jun 20 07:09:56 ubuntucodearts kernel: [Firmware Bug]: TSC doesn't count with P0 frequency!
jun 20 07:09:56 ubuntucodearts kernel: x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256
jun 20 07:09:56 ubuntucodearts kernel: x86/fpu: Supporting XSAVE feature 0x01: 'x87 floating point
jun 20 07:09:56 ubuntucodearts kernel: x86/fpu: Supporting XSAVE feature 0x02: 'SSE registers'
jun 20 07:09:56 ubuntucodearts kernel: x86/fpu: Supporting XSAVE feature 0x04: 'AUX registers'
jun 20 07:09:56 ubuntucodearts kernel: x86/fpu: Enabled xstate features 0x7, context size is 832 b
jun 20 07:09:56 ubuntucodearts kernel: e820: BIOS-provided physical RAM map:
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usab
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] rese
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x000000000009f000-0x000000000000ffff] rese
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x0000000000100000-0x00000000007ffeffff] usab
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x00000000007fff0000-0x00000000007fffffff] ACPI
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] rese
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] rese
jun 20 07:09:56 ubuntucodearts kernel: BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] rese
jun 20 07:09:56 ubuntucodearts kernel: NX (Execute Disable) protection: active
jun 20 07:09:56 ubuntucodearts kernel: SMBIOS 2.5 present.
jun 20 07:09:56 ubuntucodearts kernel: DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12
jun 20 07:09:56 ubuntucodearts kernel: Hypervisor detected: KVM
jun 20 07:09:56 ubuntucodearts kernel: Kernel/User page tables isolation: disabled
jun 20 07:09:56 ubuntucodearts kernel: e820: update [mem 0x00000000-0x00000fff] usable ==> reserve
jun 20 07:09:56 ubuntucodearts kernel: e820: remove [mem 0x000a0000-0x0000ffff] usable
jun 20 07:09:56 ubuntucodearts kernel: e820: last_pfn = 0x7fff0 max_arch_pfn = 0x400000000
jun 20 07:09:56 ubuntucodearts kernel: MTRR default type: uncachable
```

- Filtrar los mensajes de error o advertencia (journalctl -p 3 -xb).

Para filtrar mensajes de error o advertencia registrados durante el último arranque del sistema ejecutamos

sudo journalctl -p 3 -xb

```
daniel@ubuntucodearts:~$ sudo journalctl -p 3 -xb
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 11:51:51 CEST. --
jun 20 07:09:56 ubuntucodearts kernel: Spectre U2 : Spectre mitigation: LFENCE not serializing, sw
lines 1-2/2 (END)
```

- Registrar el estado del servicio saludo y guardar una copia del log en /srv/logs/saludo_journal.log.

Creamos la carpeta `/srv/logs` con `sudo mkdir -p /srv/logs`
Y exportamos los logs del servicio `saludo.service` con `sudo journalctl -u saludo.service | sudo tee /srv/logs/saludo_journal.log > /dev/null`

Y con `cat /srv/logs/saludo_journal.log` verificamos que se creó correctamente.

```
laniel@ubuntucodearts:~$ cat /srv/logs/saludo_journal.log
-- Logs begin at vie 2025-06-20 07:09:56 CEST, end at vie 2025-06-20 12:10:35 CEST. --
jun 20 11:04:17 ubuntucodearts systemd[1]: Starting Mensaje de saludo al iniciar el sistema...
jun 20 11:04:17 ubuntucodearts systemd[1]: Started Mensaje de saludo al iniciar el sistema.
laniel@ubuntucodearts:~$ _
```