Los registros (logs) del sistema

1. Fundamentos

Cuando el sistema se inicia, se pone en marcha y efectúa cualquier tipo de operación, se registran sus acciones y las de la mayoría de sus servicios en diferentes archivos. Tres servicios están especializados en la recepción de los mensajes que tienen como destino estos archivos:

- **syslogd**: *system log daemon*, encargado de la gestión de la información emitida por cualquier tipo de servicio y, si procede, del núcleo. A menudo reemplazado por syslog.ng o rsyslog
- **journald**: componente de systemd, encargado de recolectar e indexar las trazas que provienen de cualquier servicio a través de sus archivos o una API, en particular systemd, pero interconectada también con syslog o kmsh.
- **klogd**: *kernel log daemon*, encargado de la gestión de la información emitida por el núcleo. Desde hace unos años no está tan presente porque rsyslog o journald también pueden recuperar los eventos del núcleo.

Los mensajes importantes emitidos por un componente del sistema deberían pasar por los servicios syslogd o journald. Esto no impide, al contrario, que un servicio pueda gestionar sus propias entradas en sus propios archivos. No se deberían colocar entradas de aplicaciones en el registro del sistema. Las trazas de acceso a las páginas web de un servidor Apache no tienen nada que hacer ahí. En cambio, las de conexión al sistema (mediante la consola, ssh, telnet, etc.) tienen mucho interés y deben estar presentes en los archivos logs del sistema.

A partir de este punto y en el resto del libro, las entradas en el registro se llamarán por su nombre de uso corriente: los **logs**.

2. Los mensajes

El servicio **klogd** gestiona los mensajes emitidos por el núcleo. Dispone de dos fuentes de

acceso a los mensajes:

- el sistema de archivos virtual **/proc**, utilizado por defecto si está presente, y en particular `/proc/kmsg` ;
- las llamadas al sistema mediante la API del núcleo, en particular `sys_syslog` , si **/proc** no está presente o si se ha pasado el parámetro `-s` a klogd.

Los demonios rsyslog o journald son capaces de leer e interpretar de forma directa el contenido de /proc/kmsg y así remplazar a klogd. Si su sistema operativo utiliza un de estos dos servicios, klogd estará ausente. No hay que preocuparse.

Los mensajes del núcleo tienen niveles de prioridad diferentes, escalados de 0 (alta prioridad) a 7 (mensaje de depuración):

Nivel	Alias sistema	Significado
0	**EMERG**	No se puede utilizar el sistema.
1	**ALERT**	Se debe efectuar una acción inmediatamente.
2	**CRIT**	Problema crítico.
3	**ERR**	Error.
4	**WARNING**	Aviso.
5	**NOTICE**	Normal, pero necesita una atención particular.
6	**INFO**	Información estándar.
7	**DEBUG**	Traza de depuración del núcleo.

Si klogd está presente, le devuelve los mensajes de nivel 0 a 6 a syslogd, que los redirigirá a los archivos de logs y si es preciso a las consolas correspondientes. Por defecto, no se traza la información de depuración de nivel 7.

Los servicios **syslogd** o **journald** reciben los mensajes. Luego los reparten según el transmisor, la gravedad, en archivos, en consolas, en forma de mails a los usuarios del sistema (root, por ejemplo), etc.

Las acciones más corrientes son la escritura de los logs en archivos, la redirección de mensajes hacia una consola (muchas veces se trata de la 10 o 12) o el envío de mensajes a root.

3. Configuración de syslog

El archivo de configuración `/etc/syslog.conf` permite definir el origen, la importancia y el destino de cada mensaje, en forma de dos campos.

- El origen define de hecho un conjunto de **sistemas** y de **subsistemas** (núcleo, servicios). La lista, extensible, se compone en el origen de los elementos siguientes. El asterisco define el conjunto de los subsistemas.

Subsistemas	Significado
auth/authpriv	Servicio de seguridad y autenticación.
cron	Servicio cron.
daemon	Los demonios del sistema.
kern	El núcleo.
lpr	El servicio de impresión.
mail	La mensajería.
news	La red.
syslog	El propio Syslog.
user	Mensajes de los procesos de los usuarios.
uucp	Unix to Unix CoPy.
local0->7	Mensajes procedentes de klogd, la cifra representa el nivel.

- La importancia o **nivel** define el nivel de gravedad del mensaje. El asterisco define el conjunto de todos los niveles. Los niveles emitidos por **klogd** y **syslogd** son equivalentes.

Nivel	Significado
emerg	No se puede utilizar el sistema.
alert	Una intervención inmediata es indispensable.
crit	Error crítico para el subsistema.
err	Error de funcionamiento.
warning	Aviso.
notice	Evento normal que merece ser señalado.
info	Sólo como información.
debug	Mensaje mandado para el ajuste.
none	Ignorar los mensajes.

- El destino o la **acción** puede ser un archivo, un mensaje a un usuario, la consola, una lista de usuarios... El asterisco indica todo el mundo.

Se inscriben los mensajes syslog en los archivos `/var/log/messages` y `/var/log/syslog` o en cualquier otro archivo configurado en `/etc/syslog.conf` .

El ejemplo siguiente procede de una instalación RHEL usando syslogd.

```
# Se coloca todo (salvo mail.*) en /var/log/messages
*.info;mail.none;authpriv.none;cron.none    /var/log/messages

# The authpriv file has restricted access.
authpriv.*                  /var/log/secure

# Mails
mail.*                      -/var/log/maillog

# Crontab
cron.*                      /var/log/cron

# Mensajes de alerta
*.emerg                     *

# errores uucp y news
uucp,news.crit              /var/log/spooler

# Mensajes de boot
local7.*                    /var/log/boot.log
```

Usted mismo puede mandar, directamente o en sus scripts, mensajes a **syslogd** mediante el comando **logger**.

4. El caso de rsyslog

El gestor de registros **rsyslog** se usa como sustituto de syslog y syslog-ng. Su configuración principal está en `/etc/rsyslog.conf` . Las configuraciones adicionales están en `/etc/rsyslog.early.conf` , utilizado antes de la activación de las capas de red, y el contenido de `/etc/rsyslog.d/*` se carga con el archivo de configuración principal.

Además de la sintaxis clásica de syslog, rsyslog dispone de módulos y de una sintaxis más completa. Esta sintaxis permite especialmente redirigir los mensajes en función de reglas más complejas que pueden describirse con la ayuda de comandos de programación. La redirección de los mensajes no se simplifica sólo en función de los

niveles y subsistemas, sino que también analiza el contenido de los mensajes.

La siguiente configuración tiene como objetivo redirigir los mensajes provenientes de la aplicación Network Manager a su propio archivo de registro. Lo hará a partir del nombre o de la porción de nombre del programa emisor.

```
if   ($programname == 'NetworkManager') or \
   ($programname startsWith 'nm-') \
then  -/var/log/NetworkManager
&    ~
```

Los archivos de syslog son compatibles con rsyslog, pero no hay reciprocidad: syslog no puede leer la sintaxis extendida de rsyslog. Se puede iniciar rsyslog en modo compatibilidad con el parámetro **-c0**.

5. systemd y journald

Journald es un servicio arrancado por systemd. Puede utilizarse como complemento de Syslog o para reemplazarlo. Se ha convertido en el estándar para la mayoría de las distribuciones Linux. Ofrece varias ventajas:

- Gestión de trazas del núcleo a través de /dev/kmsg.
- Gestión de trazas de los servicios y aplicaciones interceptando las llamadas a syslog.
- API nativa para la generación de trazas estructuradas.
- Recuperación e indexación de las entradas y salidas estándar de los servicios.
- Gestión de las entradas de auditoría de sistema.

Los datos recolectados se gestionan de forma segura. En teoría, estos no pueden intercambiarse mediante la adición de metadatos. Por defecto, los registros se almacenan en formato binario en el directorio **/run/log/journal** o **/var/log/journal**.

```
$ ls -l /run/log/journal/b8869de6c79d4b25be724208cc49a45c/system.journal
```

```
-rw-r-----+ 1 root systemd-journal 5181440 ene 29 09:51
/run/log/journal/b8869de6c79d4b25be724208cc49a45c/system.journal
```

El sistema de archivos /run es volátil, y por tanto está vacío en cada reinicio. Para tomar registros persistentes al hacer reboot hay que desplazarlos a /var como sigue:

```
# mkdir -p /var/log/journal
# systemd-tmpfiles --create --prefix /var/log/journal
# systemctl restart systemd-journald
# # ls /var/log/journal
b8869de6c79d4b25be724208cc49a45c
```

Este formato binario ha hecho correr mucha tinta, ya que no respeta la lógica Unix de archivos de texto plano. Los comandos filtro y las herramientas no pueden usarse con este formato, en beneficio del comando **journalctl**.

El archivo de configuración de journald es **/etc/systemd/journald.conf**. Sus entradas están por lo general comentadas, dejando los valores por defecto de systemd.

```
# cat /etc/systemd/journald.conf
[Journal]
#Storage=auto
#Compress=yes
#Seal=yes
#SplitMode=uid
#SyncIntervalSec=5m
#RateLimitIntervalSec=30s
#RateLimitBurst=1000
...
```

Éstos son algunos valores destacables:

- **Storage**: define el lugar de almacenamiento de los registros. El valor «**auto**» escribirá en /var/log/journal si existe y si no vuelve a /run/log/journal. Si se indica «**persistent**», entonces journald creará la carpeta /var/log/journal para almacenar ahí sus datos. En caso de configurar «**volatil**», forzará el uso de la ruta

/run/log/journal y por último, si se configura en «**none**», no se almacenará ningún rastro.

- **SystemMaxUse / RunMaxUse**: tasa de ocupación máxima del espacio de disco antes de reciclar. Por defecto, 10%.
- **MaxRetentionSec**: duración máxima de retención de los mensajes. El valor por defecto, 0, es suficiente: éste será el tamaño de ocupación de disco que gobernará la retención.
- **ForwardToSyslog**: por defecto configurado en yes. Los mensajes recopilados son reenviados a syslog, permitiendo la generación de los archivos contenidos en **/var/log**.
- **MaxLevelStore**: nivel de criticidad de los registros que se han de tratar. Por defecto se configura en debug (7).

En las versiones antiguas, para que rsyslog pueda leer los registros tratados por journald y funcionar como de costumbre, se debe conectar a un archivo de tipo socket propuesto por journald. Por ejemplo, en RHEL 7 o CentOS 7:

```
# cat /etc/rsyslog.d/listen.conf
$SystemLogSocketName /run/systemd/journal/syslog
```

En las versiones más recientes, journald enviará los registros a rsyslog. El parámetro de la configuración `ForwardToSyslog`, que se encuentra activado por defecto, gestiona este funcionamiento.

```
# cat /etc/systemd/journald.conf | grep ForwardToSyslog
ForwardToSyslog=yes
```

6. Los registros

Por convención se ubican los logs del sistema en `/var/log`. No todos los archivos de logs de este directorio provienen de **syslogd** o **journald**. Es el caso, por ejemplo, de la información de conexión. Veamos un ejemplo del contenido de este directorio. Contiene

varios archivos de texto y directorios. Unos servicios pueden decidir, sin pasar por **syslogd**, concentrar y escribir sus mensajes en esta estructura.

```
# cd /var/log ; ls -l
-rw-r-----  1 root  root     2460 feb  7 05:34 acpid
drwxr-x---  2 root  root     4096 mar  5  2007 audit
-rw-------  1 root  root      116 mar 27 04:02 boot.log
-rw-------  1 root  root    75487 mar 28 11:10 cron
drwxr-xr-x  2 lp    sys      4096 mar 27 04:02 cups
-rw-r--r--  1 root  root    28359 feb  7 05:34 dmesg
drwx------  2 root  root     4096 ago  7  2007 httpd
-r--------  1 root  root 18747276 mar 28 11:08 lastlog
drwxr-xr-x  2 root  root     4096 jun  1  2007 mail
-rw-------  1 root  root     4537 mar 28 04:02 maillog
-rw-------  1 root  root   178348 mar 28 11:10 messages
drwx------  2 root  root     4096 oct 16 23:21 samba
-rw-------  1 root  root   214999 mar 28 11:08 secure
-rw-r--r--  1 root  root     2734 mar 28 11:01 snmpd.log
-rw-------  1 root  root        0 mar 23 04:02 spooler
drwxr-x---  2 squid squid    4096 ene 22  2007 squid
-rw-------  1 root  root    62165 mar 28 09:13 sudo.log
drwxr-xr-x  2 root  root     4096 oct  5  2004 vbox
-rw-rw-r--  1 root  utmp   127872 mar 28 11:10 wtmp
-rw-------  1 root  root    40557 mar 28 11:03 xferlog
```

7. Journalctl

El comando **journactl** permite acceder a las trazas almacenadas por journald.

A continuación se presentan varios ejemplos de uso del comando **journactl**.

- Ver todos los registros:

```
# journalctl
```

- Esperar a nuevos registros con un equivalente de la opción `-f` de `tail`:

```
# journalctl -f
```

- Mostrar según el comando que las ha enviado los registros. Ojo, en las primeras versiones de journalctl la unidad debe precisarse usando el parámetro `_SYSTEMD_UNIT=xxx.servicio`. Este método sigue siendo compatible con las versiones actuales, pero el parámetro `-u` es bastante más práctico.

```
journalctl -u ssh
-- Logs begin at Sat 2021-04-17 17:06:41 UTC, end at Wed 2021-05-05 20:17:01 UTC. --
Apr 17 17:06:57 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on 0.0.0.0 port 22.
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on :: port 22.
Apr 17 17:06:58 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Apr 17 17:07:12 ubuntu sshd[838]: Received signal 15; terminating.
Apr 17 17:07:12 ubuntu systemd[1]: Stopping OpenBSD Secure Shell server...
Apr 17 17:07:12 ubuntu systemd[1]: ssh.service: Succeeded.
Apr 17 17:07:12 ubuntu systemd[1]: Stopped OpenBSD Secure Shell server.
Apr 17 17:07:12 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 17 17:07:12 ubuntu sshd[1384]: Server listening on 0.0.0.0 port 22.
Apr 17 17:07:12 ubuntu sshd[1384]: Server listening on :: port 22.
Apr 17 17:07:12 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Apr 17 17:19:23 ubuntu sshd[1570]: Accepted password for alejandro from
192.168.1.22 port 61202 ssh2
Apr 17 17:19:23 ubuntu sshd[1570]: pam_unix(sshd:session): session opened for user
alejandro by (uid=0)
Apr 19 06:06:09 ubuntu sshd[4942]: Accepted password for alejandro from 192.168.1.22
port 50932 ssh2
Apr 19 06:06:09 ubuntu sshd[4942]: pam_unix(sshd:session): session opened for user
alejandro by (uid=0)
Apr 19 06:08:15 ubuntu sshd[5100]: Connection closed by authenticating user alejandro
192.168.1.22 port 50980 [preauth]
Apr 19 06:09:26 ubuntu sshd[5102]: Accepted publickey for alejandro from 192.168.1.22
port 51004 ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:09:26 ubuntu sshd[5102]: pam_unix(sshd:session): session opened for user
alejandro by (uid=0)
Apr 19 06:09:57 ubuntu sshd[5212]: Accepted publickey for alejandro from 192.168.1.22
port 51028 ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:09:57 ubuntu sshd[5212]: pam_unix(sshd:session): session opened for user
```

```
alejandro by (uid=0)
Apr 19 06:11:15 ubuntu sshd[5308]: Accepted publickey for alejandro from 192.168.1.22
port 51042 ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:11:15 ubuntu sshd[5308]: pam_unix(sshd:session): session opened for user
alejandro by (uid=0)
Apr 19 06:12:30 ubuntu sshd[5421]: Accepted publickey for alejandro from 192.168.1.22
port 51056 ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:12:30 ubuntu sshd[5421]: pam_unix(sshd:session): session opened for user
alejandro by (uid=0)
```

Si pulsa la tecla [Tab] después de introducir el caracter "_", el auto completado se activa y ofrece todas las posibilidades. Esto es válido para cada etapa de la introducción de parámetros en journalctl.

```
# journalctl _ [tab]
_AUDIT_FIELD_APPARMOR=    _AUDIT_FIELD_REQUESTED_MASK=  _CAP_EFFECTIVE=
_KERNEL_DEVICE=          _STREAM_ID=                _SYSTEMD_USER_SLICE=
...
# journalctl -u [tab]
accounts-daemon.service
colord.service
multipathd.service
session-7.scope
systemd-timedated.service
acpid.service
cron.service
...
```

De esta manera puede poner el nombre de identificador syslog (la aplicación o el servicio) con el parámetro `-t`, normalmente es el nombre del programa o del binario:

```
# journalctl -t sshd
-- Logs begin at Sat 2021-04-17 17:06:41 UTC, end at Wed 2021-05-05 20:17:01 UTC. --
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on 0.0.0.0 port 22.
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on :: port 22.
```

```
...
```

Usted puede definir un intervalo de tiempo con **-S** (since) y **-U** (until):

```
journalctl -u ssh -S "2021-04-01" -U "2021-05-02"
-- Logs begin at Sat 2021-04-17 17:06:41 UTC, end at Wed 2021-05-05 20:17:01 UTC. --
Apr 17 17:06:57 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on 0.0.0.0 port 22.
Apr 17 17:06:58 ubuntu sshd[838]: Server listening on :: port 22.
Apr 17 17:06:58 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Apr 17 17:07:12 ubuntu sshd[838]: Received signal 15; terminating.
Apr 17 17:07:12 ubuntu systemd[1]: Stopping OpenBSD Secure Shell server...
Apr 17 17:07:12 ubuntu systemd[1]: ssh.service: Succeeded.
Apr 17 17:07:12 ubuntu systemd[1]: Stopped OpenBSD Secure Shell server.
Apr 17 17:07:12 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 17 17:07:12 ubuntu sshd[1384]: Server listening on 0.0.0.0 port 22.
Apr 17 17:07:12 ubuntu sshd[1384]: Server listening on :: port 22.
Apr 17 17:07:12 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Apr 17 17:19:23 ubuntu sshd[1570]: Accepted password for alejandro from 192.168.1.22 port 61202 ssh2
Apr 17 17:19:23 ubuntu sshd[1570]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
Apr 19 06:06:09 ubuntu sshd[4942]: Accepted password for alejandro from 192.168.1.22 port 50932 ssh2
Apr 19 06:06:09 ubuntu sshd[4942]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
Apr 19 06:08:15 ubuntu sshd[5100]: Connection closed by authenticating user alejandro 192.168.1.22
port 50980 [preauth]
Apr 19 06:09:26 ubuntu sshd[5102]: Accepted publickey for alejandro from 192.168.1.22 port 51004
ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:09:26 ubuntu sshd[5102]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
Apr 19 06:09:57 ubuntu sshd[5212]: Accepted publickey for alejandro from 192.168.1.22 port 51028
ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:09:57 ubuntu sshd[5212]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
Apr 19 06:11:15 ubuntu sshd[5308]: Accepted publickey for alejandro from 192.168.1.22 port 51042
ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:11:15 ubuntu sshd[5308]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
Apr 19 06:12:30 ubuntu sshd[5421]: Accepted publickey for alejandro from 192.168.1.22 port 51056
```

```
ssh2: RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 19 06:12:30 ubuntu sshd[5421]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
-- Reboot --
Apr 22 21:35:30 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 22 21:35:33 ubuntu sshd[640]: Server listening on 0.0.0.0 port 22.
Apr 22 21:35:33 ubuntu sshd[640]: Server listening on :: port 22.
Apr 22 21:35:33 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Apr 22 21:35:36 ubuntu sshd[659]: Accepted publickey for alejandro from 192.168.1.42 port 51304 ssh2:
RSA SHA256:anCS+Gb3OQ7MNEh/bN6RGGXf3Ehz/XBi8XH1U6Wl3UE
Apr 22 21:35:36 ubuntu sshd[659]: pam_unix(sshd:session): session opened for user alejandro
by (uid=0)
-- Reboot --
Apr 26 20:39:17 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Apr 26 20:39:18 ubuntu sshd[630]: Server listening on 0.0.0.0 port 22.
Apr 26 20:39:18 ubuntu sshd[630]: Server listening on :: port 22.
Apr 26 20:39:18 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
```

Por último, puede modificar el formato de salida con -o:

```
# journalctl -u ssh -S "2021-04-01" -U "2021-05-02" -o json-pretty
{
        "__MONOTONIC_TIMESTAMP" : "22065251",
        "SYSLOG_IDENTIFIER" : "systemd",
        "PRIORITY" : "6",
        "CODE_LINE" : "574",
        "_SELINUX_CONTEXT" : "unconfined\n",
        "CODE_FILE" : "src/core/job.c",
        "_GID" : "0",
        "_BOOT_ID" : "21302cc63ae743d4a113faef23e830d1",
        "__REALTIME_TIMESTAMP" : "1618679217837754",
        "_EXE" : "/usr/lib/systemd/systemd",
        "JOB_ID" : "104",
        "_PID" : "1",
        "_SYSTEMD_SLICE" : "-.slice",
        "UNIT" : "ssh.service",
        "_UID" : "0",
        "_HOSTNAME" : "ubuntu",
```

```
        "_CAP_EFFECTIVE" : "3fffffffff",
        "_TRANSPORT" : "journal",
        "_SYSTEMD_CGROUP" : "/init.scope",
        "_SOURCE_REALTIME_TIMESTAMP" : "1618679217836720",
        "__CURSOR" : "s=6d4a5988d4cd432ab88befa63cf5ebba;i=421;b=
21302cc63ae743d4a113faef23e830d1;m=150b063;t=5c02e1fc8d6ba;x=58001bed2e5525f0",
        "CODE_FUNC" : "job_log_begin_status_message",
        "_CMDLINE" : "/sbin/init maybe-ubiquity",
        "SYSLOG_FACILITY" : "3",
...
```

8. Emisión de mensajes

El comando **logger** permite enviar mensajes que se tratarán como logs. Se podrá ver el resultado, dependiendo de la configuración de syslog, en `/var/log/messages`, o `/var/log/syslog`.

```
$ logger "Mensaje de pruebas"
$ tail -1 /var/log/syslog
May  5 20:51:43 ubuntu alejandro: Mensaje de pruebas
```

Puede precisar el servicio y el nivel con `-p`:

```
$ logger -p local0.warn "Cuidado"
```

Otro comando práctico es **systemd-cat**: toma un archivo en entrada o un pipe y lo escribe en el registro.

```
$ echo Hola | systemd-cat -t miscript -p warning
$ journalctl -f
May 05 20:54:56 ubuntu miscript[6686]: Hola
```

```
# journalctl  -t miscript
-- Logs begin at Sat 2021-04-17 17:06:41 UTC, end at Wed 2021-05-05 20:54:56 UT>
May 05 20:54:56 ubuntu miscript[6686]: Hola
```

9. Rotación de logs

Con el paso del tiempo, los archivos de logs van consumiendo más espacio hasta poder llegar a saturar el sistema de archivos, existen diferentes motivos para ello:

- Un programa o servicio puede ser, de manera natural, muy hablador, incluso con sus parámetros por defecto.
- Algunos servicios se ejecutan en modo depuración o «debug», incluso en un entorno de producción.
- Nadie piensa en limpiar los antiguos logs, los cuales van acumulándose.

Después de algunos meses, años o simplemente algunas horas, los logs pueden ocupar varios gigas. Pueden incluso saturar un disco, lo que puede conllevar consecuencias nefastas, hasta hacer que el sistema no funcione. Por lo tanto es necesario limpiarlos regularmente.

a. logrotate

El comando **logrotate** se creó para este cometido. Su principio es simple: efectúa una rotación de los archivos de logs, almacenando los antiguos mensajes en archivos de respaldo, eventualmente comprimiéndolos, y después de un cierto número de rotaciones, borra los archivos más antiguos. Los archivos de configuración son `/etc/logrotate.conf` acompañado de los que se encuentran en el directorio `/etc/logrotate.d/*` .

He aquí la configuración de logrotate para el archivo /var/log/syslogg:

```
/var/log/syslog
```

```
{
  rotate 7
  daily
  missingok
  notifempty
  delaycompress
  compress
  postrotate
      /usr/lib/rsyslog/rsyslog-rotate
  endscript
}
```

Y aquí el resultado en /var/log después de algunos días:

```
# ls -1 /var/log/syslog*
/var/log/syslog
/var/log/syslog.1
/var/log/syslog.2.gz
/var/log/syslog.3.gz
/var/log/syslog.4.gz
/var/log/syslog.5.gz
/var/log/syslog.6.gz
/var/log/syslog.7.gz
```

Logrotate toma muchos parámetros, he aquí los más corrientes:

- **rotate**: número de rotaciones que se van a conservar. Un valor de 7 guarda los últimos siete archivos. Los archivos que resultan de la operación son numerados en consecuencia.
- **daily**: el comando se ejecutará cada día.
- **missingok**: en el caso de que el archivo no se encuentre presente, no se provocará un error.
- **notifempty**: no efectuar la rotación si el archivo está vacío.
- **compress**: los archivos que se traten serán comprimidos.
- **delaycompress**: la compresión del archivo se efectuará en la siguiente iteración. Es por esto por lo que el archivo syslog.1 no está comprimido.

- **postrotate**: el archivo indicado se ejecuta en shell, puede ser una línea de comandos o un script. Una vez que se haya efectuado la rotación, un nuevo archivo vacío se creará. Algunos servicios tienen que recargarse para tomar en cuenta este cambio.
- **minsize**: la talla mímima del archivo que necesitará una rotación. Por debajo de esta talla, no habrá rotación antes del intervalo configurado.
- **maxsize**: si la talla máxima es alcanzada antes del intervalo dado, una rotación tendrá lugar.
- **nocreate**: ningún nuevo archivo será creado por logrotate. El servicio encargado deberá hacerlo.

b. journald

Para conocer el espacio usado por los archivos de journald, use este comando:

```
# journalctl -disk-usage
Archived and active journals take up 120.0M in the file system.
```

Los umbrales de rotación de los logs están configurados en `/etc/systemd/journald.conf` :

- **SystemKeepFree**: el espacio que journald debe dejar libre en el disco si los logs son persistentes.
- **RuntimeKeepFree**: el mismo caso que el anterior, pero solamente para los logs no persistentes.
- **SystemMaxUse**: el espacio que journald puede utilizar como máximo en el disco.
- **RuntimeMaxUse**: igual, pero para los logs no persistentes.
- **MaxFileSec**: conservar solamente los logs del periodo de tiempo indicado. 5days solo guardará 5 días de logs.

A veces podrá tener la impresión de que a journald no le importan sus opciones: usted le indica una talla máxima de 100 MB, y sin embargo sus registros ocupan 200 ó 300 MB. La razón es la siguiente: journald usará el parámetro que le permitirá almacenar el mayor espacio posible. Observe que systemd está limitado a 4 GB por registro.

Si su sistema de archivos dispode de 4GB y usted quiere limitar la ocupación a 400-500 MB, deberá configurar journald con unas instrucciones similares a:

```
SystemMaxUse=400M
SystemKeepFree=3500M
```

Puede hacer una limpieza manual de los logs con el comando siguiente:

```
# journalctl --vacuum-size=10M
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
system@00059c6d61b6e3ee-b73696f87c71990e.journal~ (16.0M).
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
user-1000@00059c6d642e3cc7-788624063e7d7bfb.journal~ (8.0M).
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
system@00059dd856ba11e2-8f2e6b1e0f661dba.journal~ (24.0M).
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
user-1000@00059e0c1d517797-a60265665becf8ae.journal~ (24.0M).
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
system@00059e0c191c9946-e7f50420d48fb300.journal~ (16.0M).
Deleted archived journal /var/log/journal/3cbf896d8fa745f485f614295f24b5d9/
system@00059e124f1b2b2a-0cc819ab963bbc3e.journal~ (8.0M).
Vacuuming done, freed 96.0M of archived journals from /var/log/journal/
3cbf896d8fa745f485f614295f24b5d9.
# journalctl -disk-usage
Archived and active journals take up 24.0M in the file system.
```