Journald

Els registres del sistema (els quals poden representar des de simples missatges informatius fins missatges crítics d'errors i poden ser rebuts directament per part del kernel o bé de processos d'usuari o de dimonis, entre d'altres origens) són recopilats per un demoni central anomenat systemd-journald. Per tant, haurà d'estar encés i habilitat per a què funcioni aquesta recopilació de registres: *systemctl start systemd-journald && systemctl enable systemd-journald*

Basic configuration

The journal stores log data in /run/log/journal/ by default (in binary form). Because the /run/ directory is volatile by nature (it's located on tmpfs, which is, in fact, RAM), log data is lost at reboot. To make the log data persistent, the directory /var/log/journal/ must exist because it's the only place where the systemd-journald service can store its data. Systemd will generally create the directory for you and switch to persistent logging if you edit /etc/systemd/journald.conf file leaving following line like this:

Storage=persistent

NOTA: If "/var/log/journal" can't be created (because the filesystem where it's mounted is read-only, for instance), the "persistent" value will fallback to save log data in "/run/log/journal" (RAM-based) automatically. On the other hand, if we wanted to save log data only in "/run/log/journal" (that is: without never touching disk), we should use the "volatile" value, instead. Also, there is the "auto" value, which is equal to "persistent" if /var/log/journal folder exists and equal to "volatile" if it doesn't (making, therefore, the existence of this folder the "switch" of such distinct behaviour). Finally, it is also possible to use the value "none"

Journal occupies by default 10% of available partition space, and when it gets it, it rotates the logs like a list (deleting the last one). To find out the disk that journal currently uses, you can execute *journalctl --disk-usage* command. If you want to change the maximum possible usage, you can edit /etc/systemd/journald.conf file to leave following lines like this:

SystemMaxUse=50G

NOTA: Existeix una directiva semblant però que afecta a l'emmagatzematge del Journal en RAM (és a dir, a /run/log/journal) anomenada *RuntimeMaxUse*
NOTA: La directiva *SystemMaxFileSize* (o *RuntimeMaxFileSize*) defineixen el tamany màxim (en K,M,G,T...) que tindrà un fitxer .journal abans de comprimir-se, rotar-se i arxivar-se. Per defecte val 1/8 del tamany indicat a SystemMaxUse, i això vol dir, per tant, que es podran tenir fins a 7 arxius .journal arxivats més l'actiu.

Alternativament, es pot indicar no un tamany màxim sinó un temps màxim, passat el qual s'esborraran els logs més antics. Això es pot fer indicant la directiva següent (on el número per defecte representa segons però es pot convertir en minuts, hores, dies, setmanes, mesos o anys si s'afegeix el sufixe "m","h","day","week","month" o "year", respectivament). Per defecte val 0, que vol dir que no s'aplica:

MaxRetentionSec=nº

NOTA: La directiva *MaxFileSec=nº* ofereix una forma alternativa de rotació de logs que, en comptes de tenir en compte el tamany màxim dels arxius arxivats (com passava amb *SystemMaxFileSize* té en compte el temps màxim que ha de passar per arxivar un arxiu .journal

On the other hand, you can forward the journal to a terminal device to inform you about system messages on a preferred terminal screen, for example /dev/tty12. Change the following options in /etc/systemd/journald.conf to:

ForwardToConsole=yes
TTYPath=/dev/tty12
MaxLevelConsole=err #El seu significat és similar al del paràmetre -p de la comanda journalctl

NOTA: Una manera alternativa de hacer lo mismo sin necesidad de tocar el archivo de configuración general sería creando un archivo de configuración suplementario tal como este, /etc/systemd/journald.conf.d/hola.conf con el siguiente contenido:

```
[Journal]
ForwardToConsole=yes
TTYPath=/dev/tty12
MaxLevelConsole=err
```

By default, Journal users without root privileges can only see log files generated by them. Adding an user to "systemd-journal" or "adm" group gives him access to all logs.

Parámetros de journalctl

-f	Similar a tail -f
-r	Muestra los mensajes al revés (el más moderno primero)
-e	Muestra todos los mensajes y se mueve hasta el último (por defecto se queda en el primero)
-n nº	Muestra solo los últimos nº mensajes. Si nº no se escribe, por defecto es 10
-x	Añade información extra respecto las líneas mostradas (enlaces a documentación, contextos de los mensajes, posibles soluciones a errores, etc)
_PID=nº	Filtra solo los mensajes relacionados con el PID indicado. Además de este campo _PID, se puede saber la lista de otros campos por los cuales filtrar ejecutando *journalctl -N* (o en la página *man systemd.journal-fields*)
_EXE=/ruta/ejecutable	Filtra solo los mensajes relacionados con el ejecutable indicado. En el caso de ser un script, es mejor utilizar el campo _COMM. En cualquier caso, se puede usar el campo _CMDLINE para filtrar no solamente por la ruta del ejecutable sino también por sus parámetros. Por otro lado, también se puede utilizar el campo _KERNEL_DEVICE para filtrar los mensajes relacionados con un determinado dispositivo /dev/xxx. Si se escribe como parámetro de journalctl directamente la ruta del ejecutable (o dispositivo) se entenderá que se quiere filtrar por los campos _EXE, _COMM o _KERNEL_DEVICE indistintamente.
_UID=nº	Filtra solo los mensajes relacionados con el UID indicado. Además de este campo _UID, se puede saber la lista de otros campos por los cuales filtrar ejecutando *journalctl -N* (o en la página *man systemd.journal-fields*)
_HOSTNAME=nomMaquina	Filtra solo los mensajes relacionados con el host indicado. Interesante en el caso de tener centralizados los mensajes de varias máquinas en un solo registro (ver más abajo). Además de este campo _HOSTNAME, se puede saber la lista de otros campos por los cuales filtrar ejecutando *journalctl -N* (o en la página *man systemd.journal-fields*)
-F _HOSTNAME	Muestra todos los valores guardados en el journal del campo indicado. Además de este campo _HOSTNAME, se puede saber la lista de otros campos por los cuales filtrar ejecutando *journalctl -N* (o en la página *man systemd.journal-fields*)
El escribir más de un campo a la vez es equivalente a un AND, a no ser que entre ellos se escriba un "+", interpretándose entonces un OR. Por ejemplo: *journalctl _UID=1000 _PID=1234 + _UID=1001 _PID=1234* mostraría los mensajes generados por el programa con PID=1234 siempre que lo haya ejecutado o bien el usuario 1000 o bien el usuario 1001.	
-g exprReg	Fltra solo los mensajes cuyo valor del campo MESSAGE coincida con la expresión regular indicada. Para conocer la sintaxis admitida se puede consultar *man pcre2pattern* Si la expresión regular se escribe toda en minúsculas, la búsqueda será case-insensitive; en caso contrario, será case-sensitive; este comportamiento se puede modificar con el uso del parámetro extra *--case-sensitive={yes|no}*

-u nombreUnit	Filtra solo los mensajes generados (o relacionados) con el "unit" indicado
-t unaTag	Filtra solo los mensajes etiquetados con el valor indicado (ver más abajo *systemd-cat*)
-p nomPrioritat	Las prioridades son (en vez del nombre se podría indicar su número): "debug" (7), "info" (6), "notice" (5), "warning" (4), "err" (3), "crit" (2), "alert" (1), "emerg" (0). Este parámetro filtra solo los mensajes con una prioridad igual o inferior a la indicada (es decir, muestra mensajes igual o más importantes). También se puede poner -p pri1..pri2 para establecer un rango concreto de prioridades a mostrar.
-S xxxx	Filtra solo los mensajes ocurridos desde la fecha y hora indicadas NOTA:El formato de la fecha ha de ser "YYYY-MM-DD hh:mm:ss"; si se omite el tiempo se asume 00:00:00; si se omiten los segundos se asume :00; si se omite la fecha se asume hoy. También se pueden indicar las expresiones "yesterday", "today" y "tomorrow", que se refieren a la medianoche del día anterior, actual o siguiente. También se puede escribir "now", que se refiere a la hora actual (útil con -f). También se pueden indicar tiempos relativos respecto la hora actual con – (antes) o + (después), así: "-1week", "-1month", "-20day" (ver man systemd.time)
-U xxx	Filtra solo los mensajes ocurridos hasta la fecha y hora indicadas
-o { short | verbose | json | json-pretty | cat | export | short-iso}	Muestra salida en el modo por defecto (mostrando los campos fecha y hora, _HOSTNAME, _COMM, _PID y MESSAGE); en modo verboso (viendo todos los campos de cada registro); en modo (casi) verboso con formato JSON; en modo (casi) verboso con formato JSON pero pensado para ser visto por humanos; mostrando solo el texto del mensaje propiamente dicho; en formato binario (pensado para exportaciones a través de la red); en formato similar al por defecto pero añadiendo el año a la fecha mostrada (cosa que por defecto no se hace)...hay varias variantes más de formatos tipo "short-xxx" que modifican la presentación de la fecha y hora de formas varias.
--no-pager	No usa paginador para mostrar los mensajes sino que los "vomita" todos de golpe (útil para entubar la salida a otro comando)
--utc	Muestra la fecha de cada mensaje en formato Universal en vez de Local (que es como Systemd las muestra por defecto)
-b nº	Filtra solo los mensajes pertenecientes al uso de la máquina tras el arranque nº indicado (donde nº=0 -o no ponerlo- representa el arranque actual, nº=-1 el anterior, y así) NOTA:Para ver todos los inicios posibles hacer *journalctl –list-boots* El segundo campo mostrado por este comando es el "boot ID", el cual sirve para referirse a un arranque concreto y se podría usar, por ejemplo, para filtrar de esta manera: *journalctl _BOOT_ID=valorBootID*. A continuación del "boot ID" aparece la fecha y hora de la primera entrada guardada para cada uno de esos arranques. Finalmente, aparecen la fecha y hora de la última entrada
--vacuum-size = 1{ K|M|G|T }	Borra del disco las entradas necesarias (empezando por las más antiguas) para que el registro ocupe solo 1G en disco. No opera con las entradas activas, solo las archivadas.
--vacuum-time = 1{ s|m|h|days|weeks |months|years }	Borra del disco las entradas más antiguas que el tiempo indicado. Si s'invoca juntament amb --vacuum-size s'aplicarà la restricció que esborri més. Si un fichero .journal incluye entradas más antiguas pero también más nuevas, no será borrado.
--flush	Asks the journal daemon to flush any log data stored in RAM (specifically, in /run/log/journal) into /var/log/journal, if persistent storage is enabled. This call does not return until the operation is complete. Note that the data is only flushed once during system runtime, so this command exits cleanly without executing any operation if this has already happened. There is a service called *systemd-journal-flush* which is normally invoked at startup, responsible of executing this command every boot.
--disk-usage	Tal com ja s'ha dit, mostra el tamany ocupat en disc pels arxius journal (actius i arxivats)
--rotate	Força una rotació dels fitxers .journal
--verify	Comprueba la integridad interna del registro
-D /ruta/journal	Útil cuando se quiere inspeccionar desde un sistema Live el journal (previo montaje de la partición donde reside) de un sistema no arrancado/arrancable

De momento no hay la posibilidad de indicar filtros excluyentes del tipo "todo excepto esto" (https://github.com/systemd/systemd/issues/2720)

Para saber la vía utilizada por los distintos programas para enviar sus respectivos mensajes al demonio Journald se puede consultar el valor del campo _TRANSPORT. Valores posibles son: "journal" (donde los programas hacen uso directamente del mecanismo nativo ofrecido por de Journald para recibir mensajes), "syslog" (donde los programas hacen uso del mecanismo tradicional pre-Systemd pero al que Journald atiende también), "kernel" (mecanismo exclusivo reservado para mensajes provenientes del kernel...existe un parámetro exclusivo para ello que es -k), "stdout" (vía usada por Journald para recopilar los mensajes enviados por los distintos servicios del sistema simplemente a stdout), etc. Respecto esto, es interesante leer el siguiente cuadro:

systemd-journald is a system service that collects and stores logging data. It creates and maintains structured, indexed journals based on logging information that is received from the kernel, from user processes via the libc syslog(3) call, from STDOUT/STDERR of system services or via its native API. As far as protocols are concerned, `systemd-journald` …

- … is the listener on a stream socket named `/run/systemd/journal/stdout`. systemd connects the raw standard outputs and errors of services (that have defaulted to or that explicitly have `StandardOutput=journal`/`StandardError=journal`) to this socket. It thus receives the protocol of variable length free-format records terminated with linefeeds.
- … is the listener on datagram sockets named `/run/systemd/journal/dev-log`, which is symbolically linked from `/dev/log`. This receives the protocol that the `syslog()` library function in the GNU C library, linked into applications, speaks.
- … tries to be a client of another service listening on a datagram socket named `/run/systemd/journal/syslog`. This also receives the protocol that the `syslog()` library function in the GNU C library speaks (although `systemd-journald` actually uses another library and another function to speak it).
- … is a reader from a character device named `/dev/kmsg`. This receives the protocol that the Linux kernel speaks, which is a protocol of variable length, largely free-format, records terminated with linefeeds.
- … is the listener on a datagram socket named `/run/systemd/journal/socket`. This is analogous to the GNU C library case in that applications link to a library that speaks a certain protocol to this socket; except that the function is `sd_journal_sendv()`, it is in a systemd C library that applications link to, and the protocol is a systemd-only protocol comprising an array of key=value pairs, and optionally a readable file descriptor in each datagram.

Para enviar un mensaje manualment al registro existe el comando específico llamado "systemd-cat", que funciona así: *echo "Backup Failed" | systemd-cat -t "nombreSimuladoAplicacion" -p "crit"* (la fecha se añade sola) O también así: *systemd-cat -t "nombreSimuladoAplicacion" -p "crit" echo "Backup Failed"* (lo mismo que el caso anterior pero además registra stderr). Una alternativa a "systemd-cat" es el venerable comando *logger -s -t "nombre" -p " crit"*

Existen programas complementarios que lanzan alertas (ejecución de un determinado programa, envío de mails o similares, etc) cuando se detecta un determinado mensaje en el journal. Ejemplos son:

* https://github.com/jjk-jacky/journal-triggerd (su página web es https://jjacky.com/journal-triggerd)
* https://github.com/The-Compiler/journalwatch (solo sirve para enviar mails)

NOTA: En el caso de usar contenedores systemd-nspawn, existe la opción de indicar si deseamos que los mensajes de log de esos contenedores vayan a parar al journal del propio contenedor (creando entonces un enlace a ellos en la máquina anfitriona) o bien al de la máquina anfitriona (creando entonces un enlace a ellos dentro del contenedor); esto se consigue añadiendo el parámetro *--link-journal* al comando *systemd-nspawn -b -M nombreContenedor ...* usado para iniciar el contenedor, indicándole el valor *"guest"* o *"host"*, respectivamente. A partir de entonces para ver los mensajes de log de un determinado contenedor en la máquina anfitriona habrá que ejecutar indistintamente *journalctl -m _HOSTNAME=nombreContenedor ...*

Accés remot de logs

Si volem veure els missatges de registre d'una màquina remota es pot fer via HTTP gràcies a un "miniservidor web" específic anomenat "systemd-journal-gatewayd", activable mitjançant socket (i que ve inclòs dins del paquet "systemd-journal-remote", el qual caldrà instal.lar prèviament). Si es mira la línia ListenStream del seu arxiu de configuració es pot comprovar que per defecte escolta a totes les interfícies pel port 19531 (procura doncs que aquest port estigui obert per l'eventual tallafocs que hi hagi al sistema!)

NOTA: Aquest servidor també pot oferir els registres via HTTPS si s'indica a l'executable presenta a la línia ExecStart= del seu arxiu .service un determinat certificat i clau privada amb els paràmetres --cert= i --key=, respectivament

Un cop funcionant el servidor gatewayd (*systemctl start systemd-journal-gatewayd*), per veure els logs en una altra màquina via HTTP podem fer servir qualsevol client com ara un navegador o curl. En aquest darrer cas, això es faria així:

curl -s -H"Accept:application/vnd.fdo.journal" http://ip.Del.Servidor:19531/entries

on el valor de la capçalera de client Accept indica el format de les dades a rebre (altres formats vàlids són "text/plain" (per defecte) o "application/json").

NOTA: També es pot afegir una altra capçalera per indicar un rang d'entrades concretes a obtenir, així: *"Range:entries=cursor[[:nºskip]:nºentrades]"*

Després de la ruta "/entries" es poden indicar diferents paràmetres en forma de "querystring" (és a dir, així /entries?param1&param2&param3...), com ara:

boot : Similar al paràmetre -b de journalctl
follow : Similar al paràmetre -f de journalctl
nomCamp=valor : Per filtrar per un valor concret d'un camp propi del Journal (_UID, _HOSTNAME, etc)

D'altra banda, en substitució de la ruta "/entries" també es poden escriure aquestes altres rutes:

/browse : Especialment pensat per utilitzar un navegador: permet visualitzar els registres de forma interactiva via web
/fields/nomCamp : Mostra els diferents valors trobats pel camp propi del Journal (_UID, _HOSTNAME, etc) indicat

Enregistrament remot de logs

Hi ha dos modes per enregistrar logs de màquines remotes en una màquina central: el mode "passiu" i el mode "actiu". En el primer, la màquina central actua com a servidor clàssic escoltant en un port i les màquines remotes actuen com clients que envien al servidor (per a què els emmagatzemi) els missatges de registre (això ho fan internament mitjançant peticions de tipus POST a la url "/upload" amb la capçalera Content-Type tenint el valor "application/vnd.fdo.journal"). No obstant, aquest mode actualment té un bug (https://github.com/systemd/systemd/issues/5744) i per tant no l'estudiarem. En el segon mode la màquina-magatzem central és qui obté de les màquines clients els missatges perquè els hi demana. En qualsevol cas, cal tenir instal.lat a totes les màquines el paquet "systemd-journal-remote". A continuació es mostren les passes necessàries a realitzar en els dos modes:

Mode actiu

1.-Als clients-oferidors només cal fer una cosa: iniciar i/o habilitar el socket "systemd-journal-gatewayd.socket" (o, alternativament, el servei "systemd-journal-gatewayd")

NOTA: Si es mira la línia ListenStream del seu arxiu de configuració es pot comprovar que per defecte escolta a totes les interfícies pel port 19531

2.-Al servidor-recolector cal executar (com a root) la següent comanda cada cop que es vulgui obtenir els logs disponibles des de la darrera vegada que es va preguntar (i guardar-los a una carpeta local anomenada/var/log/journal/remote/remote-ip.Del.Client.journal):

/lib/systemd/systemd-journal-remote --url http://ip.Del.Client:19531

Mode passiu

1.-Al servidor-magatzem cal fer el següent:

a) Opcional. Editar l'arxiu "/lib/systemd/system/systemd-journal-remote.socket" (o un d'equivalent a la carpeta /etc...) per a què la línia *ListenStream* s'adequi al port on volem que escolti el servidor

b) Editar l'arxiu "/lib/systemd/system/systemd-journal-remote.service" (o un d'equivalent a la carpeta /etc...) i canviar el paràmetre *--listen-https=-3* de la comanda indicada a la directiva *ExecStart per --listen-http=-3*

c) Crear, si no ho està ja, la carpeta /var/log/journal/remote amb posar com a propietari l'usuari "systemd-journal-remote" i grup "systemd-journal".

d) Recarregar Systemd (*systemctl daemon-reload*) i iniciar -i/o habilitar el servei- *systemctl start systemd-journal-remote.socket*

NOTA: El servei systemd-journal-remote té /etc/systemd/journal-remote.conf com arxiu de configuració. Allà hi ha una secció [Remote] que pot tenir diferents opcions, com ara *SplitMode=host*

2.-Als clients-enviadors cal fer el següent:

NOTA: Com alternativa al procediment descrit, al clients es pot utilitzar el servei "systemd-netlogd" (https://github.com/systemd/systemd-netlogd). Aquest servei no té el bug del servei "systemd-journal-upload".

a) Editar l'arxiu /etc/systemd/journal-upload.conf per a què la línia URL= sota la secció [Upload] apunti a la IP i port on estarà escoltant el servidor-magatzem. Així, per exemple:

[Upload]
URL=http://ip.Del.Servidor:nºportServidor

NOTA: De forma alternativa, si no es vol editar l'arxiu anterior directament (perquè una actualització el pugui matxacar) es pot crear un arxiu amb nom qualsevol però extensió ".*conf" dins de la carpeta /etc/systemd/journal-upload.conf.d amb el mateix contingut

b) Iniciar -i/o habilitar el servei- *systemctl start systemd-journal-upload.service*

Per veure al servidor els registres obtinguts de clients remots (ja sigui de manera passiva o activa) es pot executar la comanda *journalctl -m* (la qual mescla tots els registres ja siguin locals o remots, provinguin d'on provinguin...però si es vol consultar només els d'una màquina concret s'hi podria afegir el filtre _HOSTNAME=nomMaquina). També es pot especificar voler consultar el fitxer "*.journal" concret on es guarden els logs del client desitjat, amb la comanda *journalctl --file /var/log/journal/remote/remote-ipClient.journal* (o tots els fitxers journal que hi ha a una carpeta amb *journalctl -D /var/log/journal/remote*)