

NAME

systemd-journald.service, systemd-journald.socket, systemd-journald-dev-log.socket, systemd-journald-audit.socket, systemd-journald – Journal service

SYNOPSIS

systemd-journald.service

systemd-journald.socket

systemd-journald-dev-log.socket

systemd-journald-audit.socket

/lib/systemd/systemd-journald

DESCRIPTION

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 a variety of sources:

- Kernel log messages, via `kmsg`
- Simple system log messages, via the libc `syslog(3)` call
- Structured system log messages via the native Journal API, see `sd_journal_print(3)`
- Standard output and standard error of service units. For further details see below.
- Audit records, originating from the kernel audit subsystem

The daemon will implicitly collect numerous metadata fields for each log messages in a secure and unfakeable way. See **systemd.journal-fields(7)** for more information about the collected metadata.

Log data collected by the journal is primarily text-based but can also include binary data where necessary. Individual fields making up a log record stored in the journal may be up to $2^{64}-1$ bytes in size.

The journal service stores log data either persistently below `/var/log/journal` or in a volatile way below `/run/log/journal/` (in the latter case it is lost at reboot). By default, log data is stored persistently if `/var/log/journal/` exists during boot, with an implicit fallback to volatile storage otherwise. Use *Storage=* in **journal.conf(5)** to configure where log data is placed, independently of the existence of `/var/log/journal/`.

On systems where `/var/log/journal/` does not exist yet but where persistent logging is desired (and the default `journal.conf` is used), it is sufficient to create the directory, and ensure it has the correct access modes and ownership:

```
mkdir -p /var/log/journal
systemd-tmpfiles --create --prefix /var/log/journal
```

See **journal.conf(5)** for information about the configuration of this service.

STREAM LOGGING

The systemd service manager invokes all service processes with standard output and standard error connected to the journal by default. This behaviour may be altered via the *StandardOutput=*/*StandardError=* unit file settings, see **systemd.exec(5)** for details. The journal converts the log byte stream received this way into individual log records, splitting the stream at newline ("`\n`", ASCII **10**) and **NUL** bytes.

If `systemd-journald.service` is stopped, the stream connections associated with all services are terminated. Further writes to those streams by the service will result in **EPIPE** errors. In order to react gracefully in this case it is recommended that programs logging to standard output/error ignore such errors. If the **SIGPIPE** UNIX signal handler is not blocked or turned off, such write attempts will also result in such process signals being generated, see **signal(7)**. To mitigate this issue, systemd service manager explicitly turns off the **SIGPIPE** signal for all invoked processes by default (this may be changed for each unit individually via the *IgnoreSIGPIPE=* option, see **systemd.exec(5)** for details). After the standard output/standard error streams have been terminated they may not be recovered until the services they are associated with are restarted. Note that during normal operation, `systemd-journald.service` stores copies of the file descriptors

for those streams in the service manager. If `systemd-journald.service` is restarted using **systemctl restart** or equivalent operation instead of a pair of separate **systemctl stop** and **systemctl start** commands (or equivalent operations), these stream connections are not terminated and survive the restart. It is thus safe to restart `systemd-journald.service`, but stopping it is not recommended.

Note that the log record metadata for records transferred via such standard output/error streams reflect the metadata of the peer the stream was originally created for. If the stream connection is passed on to other processes (such as further child processes forked off the main service process), the log records will not reflect their metadata, but will continue to describe the original process. This is different from the other logging transports listed above, which are inherently record based and where the metadata is always associated with the individual record.

In addition to the implicit standard output/error logging of services, stream logging is also available via the **systemd-cat(1)** command line tool.

Currently, the number of parallel log streams `systemd-journald` will accept is limited to 4096. When this limit is reached further log streams may be established but will receive **EPIPE** right from the beginning.

SIGNALS

SIGUSR1

Request that journal data from `/run/` is flushed to `/var/` in order to make it persistent (if this is enabled). This must be used after `/var/` is mounted, as otherwise log data from `/run` is never flushed to `/var` regardless of the configuration. The **journalctl --flush** command uses this signal to request flushing of the journal files, and then waits for the operation to complete. See **journalctl(1)** for details.

SIGUSR2

Request immediate rotation of the journal files. The **journalctl --rotate** command uses this signal to request journal file rotation.

SIGRTMIN+1

Request that all unwritten log data is written to disk. The **journalctl --sync** command uses this signal to trigger journal synchronization, and then waits for the operation to complete.

KERNEL COMMAND LINE

A few configuration parameters from `journald.conf` may be overridden on the kernel command line:

```
systemd.journald.forward_to_syslog=, systemd.journald.forward_to_kmsg=,
systemd.journald.forward_to_console=, systemd.journald.forward_to_wall=
```

Enables/disables forwarding of collected log messages to syslog, the kernel log buffer, the system console or wall.

See **journald.conf(5)** for information about these settings.

ACCESS CONTROL

Journal files are, by default, owned and readable by the "systemd-journal" system group but are not writable. Adding a user to this group thus enables them to read the journal files.

By default, each logged in user will get their own set of journal files in `/var/log/journal/`. These files will not be owned by the user, however, in order to avoid that the user can write to them directly. Instead, file system ACLs are used to ensure the user gets read access only.

Additional users and groups may be granted access to journal files via file system access control lists (ACL). Distributions and administrators may choose to grant read access to all members of the "wheel" and "adm" system groups with a command such as the following:

```
# setfacl -Rnm g:wheel:rx,d:g:wheel:rx,g:adm:rx,d:g:adm:rx /var/log/journal/
```

Note that this command will update the ACLs both for existing journal files and for future journal files created in the `/var/log/journal/` directory.

FILES

/etc/systemd/journald.conf

Configure **systemd-journald** behavior. See **journald.conf(5)**.

/run/log/journal/machine-id/*.journal, /run/log/journal/machine-id/*.journal~,
/var/log/journal/machine-id/*.journal, /var/log/journal/machine-id/*.journal~

systemd-journald writes entries to files in /run/log/journal/machine-id/ or /var/log/journal/machine-id/ with the ".journal" suffix. If the daemon is stopped uncleanly, or if the files are found to be corrupted, they are renamed using the ".journal~" suffix, and **systemd-journald** starts writing to a new file. /run is used when /var/log/journal is not available, or when **Storage=volatile** is set in the **journald.conf(5)** configuration file.

When **systemd-journald** ceases writing to a journal file, it will be renamed to "*original-name@suffix.journal*" (or "*original-name@suffix.journal~*"). Such files are "archived" and will not be written to any more.

In general, it is safe to read or copy any journal file (active or archived). **journalctl(1)** and the functions in the **sd-journal(3)** library should be able to read all entries that have been fully written.

systemd-journald will automatically remove the oldest archived journal files to limit disk use. See **SystemMaxUse=** and related settings in **journald.conf(5)**.

/dev/kmsg, /dev/log, /run/systemd/journal/dev-log, /run/systemd/journal/socket,
/run/systemd/journal/stdout

Sockets and other paths that **systemd-journald** will listen on that are visible in the file system. In addition to these, **journald** can listen for audit events using netlink.

SEE ALSO

systemd(1), **journalctl(1)**, **journald.conf(5)**, **systemd.journal-fields(7)**, **sd-journal(3)**, **systemd-coredump(8)**, **setfacl(1)**, **sd_journal_print(3)**, **pydoc systemd.journal**