NAME

journalctl - Query the systemd journal

SYNOPSIS

journalctl [OPTIONS...] [MATCHES...]

DESCRIPTION

journalctl may be used to query the contents of the **systemd(1)** journal as written by **systemd-journald.service**(8).

If called without parameters, it will show the full contents of the journal, starting with the oldest entry collected.

If one or more match arguments are passed, the output is filtered accordingly. A match is in the format FIELD=VALUE, e.g. _SYSTEMD_UNIT=httpd.service, referring to the components of a structured journal entry. See **systemd.journal-fields**(7) for a list of well-known fields. If multiple matches are specified matching different fields, the log entries are filtered by both, i.e. the resulting output will show only entries matching all the specified matches of this kind. If two matches apply to the same field, then they are automatically matched as alternatives, i.e. the resulting output will show entries matching any of the specified matches for the same field. Finally, if the character + appears as a separate word on the command line, all matches before and after are combined in a disjunction (i.e. logical OR).

As shortcuts for a few types of field/value matches, file paths may be specified. If a file path refers to an executable file, this is equivalent to an _EXE= match for the canonicalized binary path. Similarly, if a path refers to a device node, this is equivalent to a _KERNEL_DEVICE= match for the device.

Output is interleaved from all accessible journal files, whether they are rotated or currently being written, and regardless of whether they belong to the system itself or are accessible user journals.

All users are granted access to their private per-user journals. However, by default, only root and users who are members of the systemd-journal group get access to the system journal and the journals of other users.

The output is paged through **less** by default, and long lines are truncated to screen width. The hidden part can be viewed by using the left-arrow and right-arrow keys. Paging can be disabled; see the **--no-pager** option and the Environment section below.

When outputing to a tty, lines are colored according to priority: lines of level ERROR and higher are colored red; lines of level NOTICE and higher are highlighted; other lines are displayed normally.

OPTIONS

The following options are understood:

--no-full, **--full**, **-l**
: Ellipsize fields when they do not fit in available columns. The default is to show full fields, allowing them to wrap or be truncated by the pager, if one is used.

  The old options **-l**/**--full** are not useful anymore, except to undo **--no-full**.

-a, **--all**
: Show all fields in full, even if they include unprintable characters or are very long.

-f, **--follow**
: Show only the most recent journal entries, and continuously print new entries as they are appended to the journal.

-e, **--pager-end**
: Immediately jump to the end of the journal inside the implied pager tool. This implies **-n1000** to guarantee that the pager will not buffer logs of unbounded size. This may be overridden with an explicit **-n** with some other numeric value on the command line. Note

that this option is only supported for the less(1) pager.

-n, **--lines=**
Show the most recent journal events and limit the number of events shown. If **--follow** is used, this option is implied. The argument, a positive integer, is optional, and defaults to 10.

--no-tail
Show all stored output lines, even in follow mode. Undoes the effect of **--lines=**.

-r, **--reverse**
Reverse output so that the newest entries are displayed first.

-o, **--output=**
Controls the formatting of the journal entries that are shown. Takes one of the following options:

short
is the default and generates an output that is mostly identical to the formatting of classic syslog files, showing one line per journal entry.

short-iso
is very similar, but shows ISO 8601 wallclock timestamps.

short-precise
is very similar, but shows timestamps with full microsecond precision.

short-monotonic
is very similar, but shows monotonic timestamps instead of wallclock timestamps.

verbose
shows the full-structured entry items with all fields.

export
serializes the journal into a binary (but mostly text-based) stream suitable for backups and network transfer (see **Journal Export Format**[1] for more information).

json
formats entries as JSON data structures, one per line (see **Journal JSON Format**[2] for more information).

json-pretty
formats entries as JSON data structures, but formats them in multiple lines in order to make them more readable by humans.

json-sse
formats entries as JSON data structures, but wraps them in a format suitable for **Server-Sent Events**[3].

cat
generates a very terse output, only showing the actual message of each journal entry with no metadata, not even a timestamp.

-x, **--catalog**
Augment log lines with explanation texts from the message catalog. This will add explanatory help texts to log messages in the output where this is available. These short help texts will explain the context of an error or log event, possible solutions, as well as pointers to support forums, developer documentation, and any other relevant manuals. Note that help texts are not available for all messages, but only for selected ones. For more information on the message catalog, please refer to the **Message Catalog Developer Documentation**[4].

Note: when attaching **journalctl** output to bug reports, please do *not* use **-x**.

-q, **--quiet**
Suppresses any warning messages regarding inaccessible system journals when run as a normal user.

-m, **--merge**
Show entries interleaved from all available journals, including remote ones.

-b [*ID*][±*offset*], **--boot=**[*ID*][±*offset*]
Show messages from a specific boot. This will add a match for _BOOT_ID=.

The argument may be empty, in which case logs for the current boot will be shown.

If the boot ID is omitted, a positive *offset* will look up the boots starting from the beginning of the journal, and a equal-or-less-than zero *offset* will look up boots starting from the end of the journal. Thus, **1** means the first boot found in the journal in chronological order, **2** the second and so on; while **-0** is the last boot, **-1** the boot before last, and so on. An empty *offset* is equivalent to specifying **-0**, except when the current boot is not the last boot (e.g. because **--directory** was specified to look at logs from a different machine).

If the 32-character *ID* is specified, it may optionally be followed by *offset* which identifies the boot relative to the one given by boot *ID*. Negative values mean earlier boots and a positive values mean later boots. If *offset* is not specified, a value of zero is assumed, and the logs for the boot given by *ID* are shown.

--list-boots
Show a tabular list of boot numbers (relative to the current boot), their IDs, and the timestamps of the first and last message pertaining to the boot.

-k, **--dmesg**
Show only kernel messages. This implies **-b** and adds the match _TRANSPORT=kernel.

-u, **--unit=***UNIT*|*PATTERN*
Show messages for the specified systemd unit *UNIT*, or for any of the units matched by *PATTERN*. If a pattern is specified, a list of unit names found in the journal is compared with the specified pattern and all that match are used. For each unit name, a match is added for messages from the unit (_SYSTEMD_UNIT=*UNIT*), along with additional matches for messages from systemd and messages about coredumps for the specified unit.

This parameter can be specified multiple times.

--user-unit=
Show messages for the specified user session unit. This will add a match for messages from the unit (_SYSTEMD_USER_UNIT= and _UID=) and additional matches for messages from session systemd and messages about coredumps for the specified unit.

This parameter can be specified multiple times.

-p, **--priority=**
Filter output by message priorities or priority ranges. Takes either a single numeric or textual log level (i.e. between 0/emerg and 7/debug), or a range of numeric/text log levels in the form FROM..TO. The log levels are the usual syslog log levels as documented in **syslog(3)**, i.e. emerg (0), alert (1), crit (2), err (3), warning (4), notice (5), info (6), debug (7). If a single log level is specified, all messages with this log level or a lower (hence more important) log level are shown. If a range is specified, all messages within the range are shown, including both the start and the end value of the range. This will add PRIORITY= matches for the specified priorities.

-c, **--cursor=**
Start showing entries from the location in the journal specified by the passed cursor.

--after-cursor=
Start showing entries from the location in the journal *after* the location specified by the this cursor. The cursor is shown when the **--show-cursor** option is used.

--show-cursor
The cursor is shown after the last entry after two dashes:

-- cursor: s=0639...

The format of the cursor is private and subject to change.

--since=, **--until=**
Start showing entries on or newer than the specified date, or on or older than the specified date, respectively. Date specifications should be of the format 2012-10-30 18:17:16. If the time part is omitted, 00:00:00 is assumed. If only the seconds component is omitted, :00 is assumed. If the date component is omitted, the current day is assumed. Alternatively the strings yesterday, today, tomorrow are understood, which refer to 00:00:00 of the day before the current day, the current day, or the day after the current day, respectively. now refers to the current time. Finally, relative times may be specified, prefixed with - or +, referring to times before or after the current time, respectively.

-F, **--field=**
Print all possible data values the specified field can take in all entries of the journal.

--system, **--user**
Show messages from system services and the kernel (with **--system**). Show messages from service of current user (with **--user**). If neither is specified, show all messages that the user can see.

-M, **--machine=**
Show messages from a running, local container. Specify a container name to connect to.

-D *DIR*, **--directory=***DIR*
Takes a directory path as argument. If specified, journalctl will operate on the specified journal directory *DIR* instead of the default runtime and system journal paths.

--file=*GLOB*
Takes a file glob as an argument. If specified, journalctl will operate on the specified journal files matching *GLOB* instead of the default runtime and system journal paths. May be specified multiple times, in which case files will be suitably interleaved.

--root=*ROOT*
Takes a directory path as an argument. If specified, journalctl will operate on catalog file hierarchy underneath the specified directory instead of the root directory (e.g. **--update-catalog** will create *ROOT*/var/lib/systemd/catalog/database).

--new-id128
Instead of showing journal contents, generate a new 128-bit ID suitable for identifying messages. This is intended for usage by developers who need a new identifier for a new message they introduce and want to make recognizable. This will print the new ID in three different formats which can be copied into source code or similar.

--header
Instead of showing journal contents, show internal header information of the journal fields accessed.

--disk-usage
Shows the current disk usage of all journal files.

--list-catalog [*128-bit-ID...*]
List the contents of the message catalog as a table of message IDs, plus their short description strings.

If any *128-bit-ID*s are specified, only those entries are shown.

--dump-catalog [*128-bit-ID...*]
Show the contents of the message catalog, with entries separated by a line consisting of two dashes and the ID (the format is the same as .catalog files).

If any *128-bit-ID*s are specified, only those entries are shown.

--update-catalog
Update the message catalog index. This command needs to be executed each time new catalog files are installed, removed, or updated to rebuild the binary catalog index.

--setup-keys
Instead of showing journal contents, generate a new key pair for Forward Secure Sealing (FSS). This will generate a sealing key and a verification key. The sealing key is stored in the journal data directory and shall remain on the host. The verification key should be stored externally. Refer to the **Seal=** option in **journald.conf**(5) for information on Forward Secure Sealing and for a link to a refereed scholarly paper detailing the cryptographic theory it is based on.

--force
When **--setup-keys** is passed and Forward Secure Sealing (FSS) has already been configured, recreate FSS keys.

--interval=
Specifies the change interval for the sealing key when generating an FSS key pair with **--setup-keys**. Shorter intervals increase CPU consumption but shorten the time range of undetectable journal alterations. Defaults to 15min.

--verify
Check the journal file for internal consistency. If the file has been generated with FSS enabled and the FSS verification key has been specified with **--verify-key=**, authenticity of the journal file is verified.

--verify-key=
Specifies the FSS verification key to use for the **--verify** operation.

-h, **--help**
Print a short help text and exit.

--version
Print a short version string and exit.

--no-pager
Do not pipe output into a pager.

EXIT STATUS

On success, 0 is returned; otherwise, a non-zero failure code is returned.

ENVIRONMENT

SYSTEMD_PAGER
Pager to use when **--no-pager** is not given; overrides *PAGER*. Setting this to an empty string or the value cat is equivalent to passing **--no-pager**.

SYSTEMD_LESS
Override the default options passed to **less** (FRSXMK).

EXAMPLES

Without arguments, all collected logs are shown unfiltered:

journalctl

With one match specified, all entries with a field matching the expression are shown:

journalctl _SYSTEMD_UNIT=avahi-daemon.service

If two different fields are matched, only entries matching both expressions at the same time are shown:

journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=28097

If two matches refer to the same field, all entries matching either expression are shown:

journalctl _SYSTEMD_UNIT=avahi-daemon.service _SYSTEMD_UNIT=dbus.service

If the separator + is used, two expressions may be combined in a logical OR. The following will show all messages from the Avahi service process with the PID 28097 plus all messages from the D-Bus service (from any of its processes):

journalctl _SYSTEMD_UNIT=avahi-daemon.service _PID=28097 + _SYSTEMD_UNIT=dbus.service

Show all logs generated by the D-Bus executable:

journalctl /usr/bin/dbus-daemon

Show all logs of the kernel device node /dev/sda:

journalctl /dev/sda

Show all kernel logs from previous boot:

journalctl -k -b -1

SEE ALSO

systemd(1), **systemd-journald.service**(8), **systemctl(1)**, **coredumpctl(1)**, **systemd.journal-fields**(7), **journald.conf**(5)

NOTES

1. Journal Export Format
   http://www.freedesktop.org/wiki/Software/systemd/export
2. Journal JSON Format
   http://www.freedesktop.org/wiki/Software/systemd/json
3. Server-Sent Events
   https://developer.mozilla.org/en-US/docs/Server-sent_events/Using_server-sent_events
4. Message Catalog Developer Documentation
   http://www.freedesktop.org/wiki/Software/systemd/catalog