journalctl

- Description
- Source Options
- Filtering Options
- Output Options
- Pager Control Options
- Forward Secure Sealing (FSS) Options
- Commands
- Exit status
- Examples
- See Also

systemd

journalctl

1

journalctl

Print log entries from the systemd journal

journalctl

OPTIONS

MATCHES

Description

`journalctl` is used to print the log entries stored in the journal by `systemd-journald.service(8)` and `systemd-journal-remote.service(8)`.

If called without parameters, it will show the contents of the journal accessible to the calling user, 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, the character `+` may appear as a separate word between other terms on the command line. This causes all matches before and after to be combined in a disjunction (i.e. logical OR).

It is also possible to filter the entries by specifying an absolute file path as an argument. The file path may be a file or a symbolic link and the file must exist at the time of the query. If a file path refers to an executable binary, an `_EXE=` match for the canonicalized binary path is added to the query. If a file path refers to an executable script, a `_COMM=` match for the script name is added to the query. If a file path refers to a device node, `_KERNEL_DEVICE=` matches for the kernel name of the device and for each of its ancestor devices is added to the query. Symbolic links are dereferenced, kernel names are synthesized, and parent devices are identified from the environment at the time of the query. In general, a device node is the best proxy for an actual device, as log entries do not usually contain fields that identify an actual device. For the resulting log entries to be correct for the actual device, the relevant parts of the environment at the time the entry was logged, in particular the actual device corresponding to the device node, must have been the same as those at the time of the query. Because device nodes generally change their corresponding devices across reboots, specifying a device node path causes the resulting entries to be restricted to those from the current boot.

Additional constraints may be added using options `--boot`, `--unit=`, etc., to further limit what entries will be shown (logical AND).

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. The `--header` option can be used to identify which files *are* being shown.

The set of journal files which will be used can be modified using the `--user`, `--system`, `--directory=`, and `--file=` options, see below.

All users are granted access to their private per-user journals. However, by default, only root and users who are members of a few special groups are granted access to the system journal and the journals of other users. Members of the groups `systemd-journal`, `adm`, and `wheel` can read all journal files. Note that the two latter groups traditionally have additional privileges specified by the distribution. Members of the `wheel` group can often perform administrative tasks.

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 outputting to a tty, lines are colored according to priority: lines of level ERROR and higher are colored red; lines of level WARNING are colored yellow; lines of level NOTICE are highlighted; lines of level INFO are displayed normally; lines of level DEBUG are colored grey.

To write entries *to* the journal, a few methods may be used. In general, output from systemd units is automatically connected to the journal, see `systemd-journald.service(8)`. In addition, `systemd-cat(1)` may be used to send messages to the journal directly.

Source Options

The following options control where to read journal records from:

`--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.

The `--user` option affects how `--unit=` arguments are treated. See `--unit=`.

Note that `--user` only works if persistent logging is enabled, via the `Storage=` setting in `journald.conf(5)`.

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

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

`-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.

`-i GLOB`; `--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 journal directories and catalog file hierarchy underneath the specified directory instead of the root directory (e.g. `--update-catalog` will create `ROOT/var/lib/systemd/catalog/database`, and journal files under `ROOT/run/journal/` or `ROOT/var/log/journal/` will be displayed).

`--image=IMAGE`
Takes a path to a disk image file or block device node. If specified, `journalctl` will operate on the file system in the indicated disk image. This option is similar to `--root=`, but operates on file systems stored in disk images or block devices, thus providing an easy way to extract log data from disk images. The disk image should either contain just a file system or a set of file systems within a GPT partition table, following the UAPI.2 Discoverable Partitions Specification. For further information on supported disk images, see `systemd-nspawn(1)`'s switch of the same name.

`--namespace=NAMESPACE`
Takes a journal namespace identifier string as argument. If not specified, the data collected by the default namespace is shown. If specified, shows the log data of the specified namespace instead. If the namespace is specified as `*` data from all namespaces is shown, interleaved. If the namespace identifier is prefixed with `+` data from the specified namespace and the default namespace is shown, interleaved, but no other. For details about journal namespaces see `systemd-journald.service(8)`.

Filtering Options

The following options control how to filter journal records:

`-S`; `--since=`; `-U`; `--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. For complete time and date specification, see `systemd.time(7)`. Note that `--output=short-full` prints timestamps that follow precisely this format.

`-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 passed cursor. The cursor is shown when the `--show-cursor` option is used.

`--cursor-file=FILE`
If <FILE> exists and contains a cursor, start showing entries *after* this location. Otherwise, show entries according to the other given options. At the end, write the cursor of the last entry to <FILE>. Use this option to continually read the journal by sequentially calling `journalctl`.

`-b ID±offset|all`; `--boot=ID±offset|all`
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 an 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 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.

The special argument `all` can be used to negate the effect of an earlier use of `-b`.

`-u`; `--unit=UNIT|PATTERN`
Show messages for the specified systemd unit <UNIT> (such as a service 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. A match is also added for `_SYSTEMD_SLICE=UNIT`, such that if the provided <UNIT> is a `systemd.slice(5)` unit, all logs of children of the slice will be shown.

With `--user`, all `--unit=` arguments will be converted to match user messages as if specified with `--user-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. A match is also added for `_SYSTEMD_USER_SLICE=UNIT`, such that if the provided <UNIT> is a `systemd.slice(5)` unit, all logs of children of the unit will be shown.

This parameter can be specified multiple times.

`-I`; `--invocation=ID±offset|offset`
Show messages from a specific invocation of unit. This will add a match for `_SYSTEMD_INVOCATION_ID=`, `OBJECT_SYSTEMD_INVOCATION_ID=`, `INVOCATION_ID=`, `USER_INVOCATION_ID=`.

A positive <offset> will look up the invocations of a systemd unit from the beginning of the journal, and zero or a negative offset will look up invocations starting from the end of the journal. Thus, `1` means the first invocation found in the journal in chronological order, `2` the second and so on; while `0` is the latest invocation, `-1` the invocation before the latest, and so on.

If the 32-character <ID> is specified, it may optionally be followed by <±offset> which identifies the invocation relative to the one given by invocation <ID>. Negative values mean earlier invocations and positive values mean later invocations. If <±offset> is not specified, a value of zero is assumed, and the logs for the invocation given by <ID> will be shown.

`-I` is equivalent to `--invocation=0`, and logs for the latest invocation will be shown.

When an offset is specified, a unit name must be specified with `-u/--unit=` or `--user-unit=` option.

When specified with `-b/--boot=`, then invocations are searched within the specified boot.

`-t`; `--identifier=SYSLOG_IDENTIFIER`
Show messages for the specified syslog identifier <SYSLOG_IDENTIFIER>.

This parameter can be specified multiple times.

If the argument contains a shell-style glob, it is matched against the syslog identifiers actually present in the journal, as for `--unit=`. If no glob matches anything, and no exact identifier was given either, an error is returned.

`-T`; `--exclude-identifier=SYSLOG_IDENTIFIER`
Exclude messages for the specified syslog identifier <SYSLOG_IDENTIFIER>.

This parameter can be specified multiple times.

If the argument contains a shell-style glob, it is matched against the syslog identifiers actually present in the journal, as for `--identifier=`. Patterns which do not match anything are silently skipped.

`-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.

`--facility=`
Filter output by syslog facility. Takes a comma-separated list of numbers or facility names. The names are the usual syslog facilities as documented in `syslog(3)`. `--facility=help` may be used to display a list of known facility names and exit.

`-g`; `--grep=`
Filter output to entries where the `MESSAGE=` field matches the specified regular expression. PERL-compatible regular expressions are used, see `pcre2pattern(3)` for a detailed description of the syntax.

If the pattern is all lowercase, matching is case insensitive. Otherwise, matching is case sensitive. This can be overridden with the `--case-sensitive` option, see below.

When used with `--lines=` (not prefixed with `+`), `--reverse` is implied.

`--case-sensitive=BOOLEAN`
Make pattern matching case sensitive or case insensitive.

`-k`; `--dmesg`
Show only kernel messages. This adds the match `_TRANSPORT=kernel`. This implies `--boot=0` unless explicitly specified otherwise.

Output Options

The following options control how journal records are printed:

`-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-full`
is very similar, but shows timestamps in the format the `--since=` and `--until=` options accept. Unlike the timestamp information shown in `short` output mode this mode includes weekday, year and timezone information in the output, and is locale-independent.

`short-iso`
is very similar, but shows timestamps in the RFC 3339 profile of ISO 8601.

`short-iso-precise`
as for `short-iso` but includes full microsecond precision.

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

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

`short-delta`
as for `short-monotonic` but includes the time difference to the previous entry. Maybe unreliable time differences are marked by a `*`.

`short-unix`
is very similar, but shows seconds passed since January 1st 1970 UTC instead of wallclock timestamps ("UNIX time"). The time is shown with microsecond accuracy.

`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 for more information). To import the binary stream back into native journald format use `systemd-journal-remote(8)`.

`json`
formats entries as JSON objects, separated by newline characters (see Journal JSON Format for more information). Field values are generally encoded as JSON strings, with three exceptions:

1. Fields larger than 4096 bytes are encoded as `null` values. (This may be turned off by passing `--all`, but be aware that this may allocate overly long JSON objects.)
2. Journal entries permit non-unique fields within the same log entry. JSON does not allow non-unique fields within objects. Due to this, if a non-unique field is encountered a JSON array is used as field value, listing all field values as elements.
3. Fields containing non-printable or non-UTF8 bytes are encoded as arrays containing the raw bytes individually formatted as unsigned numbers.

Note that this encoding is reversible (with the exception of the size limit).

`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 .

`json-seq`
formats entries as JSON data structures, but prefixes them with an ASCII Record Separator character (0x1E) and suffixes them with an ASCII Line Feed character (0x0A), in accordance with JavaScript Object Notation (JSON) Text Sequences (`application/json-seq`).

`cat`
generates a very terse output, only showing the actual message of each journal entry with no

metadata, not even a timestamp. If combined with the `--output-fields=` option will output the listed fields for each log record, instead of the message.

`with-unit`
similar to `short-full`, but prefixes the unit and user unit names instead of the traditional syslog identifier. Useful when using templated instances, as it will include the arguments in the unit names.

`--truncate-newline`
Truncate each log message at the first newline character on output, so that only the first line of each message is displayed.

`--output-fields=`
A comma separated list of the fields which should be included in the output. This has an effect only for the output modes which would normally show all fields (`verbose`, `export`, `json`, `json-pretty`, `json-sse` and `json-seq`), as well as on `cat`. For the former, the `__CURSOR`, `__REALTIME_TIMESTAMP`, `__MONOTONIC_TIMESTAMP`, and `_BOOT_ID` fields are always printed.

`-n`; `--lines=`
Show the most recent journal events and limit the number of events shown. The argument is a positive integer or `all` to disable the limit. Additionally, if the number is prefixed with `+`, the oldest journal events are used instead. The default value is 10 if no argument is given.

If `--follow` is used, this option is implied. When not prefixed with `+` and used with `--grep=`, `--reverse` is implied.

`-r`; `--reverse`
Reverse output so that the newest entries are displayed first.

`--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.

`--utc`
Express time in Coordinated Universal Time (UTC).

`-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, see Journal Message Catalogs.

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

`-W`; `--no-hostname`
Do not show the hostname field of log messages. This switch has an effect only on the `short` family of output modes (see above).

Note: this option does not remove occurrences of the hostname from log entries themselves, so it does not prevent the hostname from being visible in the logs.

`--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. By default, fields with unprintable characters are abbreviated as "blob data". (Note that the pager may escape unprintable characters again.)

`-f`; `--follow`
Show only the most recent journal entries, and continuously print new entries as they are appended to the journal, until Ctrl-C is hit (or the tool is otherwise terminated).

`journalctl` will send an `sd_notify(3)` `READY=1` message once it initialized and successfully established its watch on the journal.

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

`-q`; `--quiet`
Suppresses all informational messages (i.e. "-- Journal begins at ...", "-- Reboot --"), any warning messages regarding inaccessible system journals when run as a normal user.

`--synchronize-on-exit=`
Takes a boolean argument. If true and operating in `--follow` mode, a journal synchronization request (equivalent to `journalctl --sync`) is issued when `SIGTERM`/`SIGINT` is received, and log output continues until this request completes. This is useful for synchronizing journal log output to the runtime of services or external events, ensuring that any log data enqueued to the logging subsystem by the time `SIGTERM`/`SIGINT` is issued is guaranteed to be processed and displayed by the time log output ends. Defaults to false.

Pager Control Options

The following options control page support:

`-e`; `--pager-end`
Immediately jump to the end of the journal inside the implied pager tool. This implies `--lines=1000` and `--boot=0` unless explicitly specified otherwise, to guarantee that the pager will not buffer logs of unbounded size. Note that this option is only supported for the `less(1)` pager.

Forward Secure Sealing (FSS) Options

The following options may be used together with the `--setup-keys` command described below:

`--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.

Note, `--output=json-sse` and `--output=json-seq` are silently migrated to `--output=json`.

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

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

Commands

The following commands are understood. If none is specified the default is to display journal records:

`-N`; `--fields`
Print all field names currently used in all entries of the journal.

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

`--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. When specified with `-n/--lines=+N` option, only the first (when the number prefixed with `+`) or the last (without prefix) <N> entries will be shown. When specified with `-r/--reverse`, the list will be shown in the reverse order.

`--list-invocations`
List invocation IDs of a unit. Requires a unit name with `-u/--unit=` or `--user-unit=`. Show a tabular list of invocation numbers (relative to the current or latest invocation), their IDs, and the timestamps of the first and last message pertaining to the invocation. When `-b/-boot` is specified, invocations in the boot will be shown. When specified with `-n/--lines=+N` option, only the first (when the number prefixed with `+`) or the last (without prefix) <N> entries will be shown. When specified with `-r/--reverse`, the list will be shown in the reverse order.

`--disk-usage`
Shows the current disk usage of all journal files. This shows the sum of the disk usage of all archived and active journal files.

`--vacuum-size=`; `--vacuum-time=`; `--vacuum-files=`
`--vacuum-size=` removes the oldest archived journal files until the disk space they use falls below the specified size. Accepts the usual `K`, `M`, `G` and `T` suffixes (to the base of 1024).

`--vacuum-time=` removes archived journal files older than the specified timespan. Accepts the usual `s` (default), `m`, `h`, `days`, `weeks`, `months`, and `years` suffixes, see `systemd.time(7)` for details.

`--vacuum-files=` leaves only the specified number of separate journal files.

Note that running `--vacuum-size=` has only an indirect effect on the output shown by `--disk-usage`, as the latter includes active journal files, while the vacuuming operation only operates on archived journal files. Similarly, `--vacuum-files=` might not actually reduce the number of journal files to below the specified number, as it will not remove active journal files.

`--vacuum-size=`, `--vacuum-time=` and `--vacuum-files=` may be combined in a single invocation to enforce any combination of a size, a time and a number of files limit on the archived journal files. Specifying any of these three parameters as zero is equivalent to not enforcing the specific limit, and is thus redundant.

These three switches may also be combined with `--rotate` into one command. If so, all active files are rotated first, and the requested vacuuming operation is executed right after. The rotation has the effect that all currently active files are archived (and potentially new, empty journal files opened as replacement), and hence the vacuuming operation has the greatest effect as it can take all log data written so far into account.

`--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.

`--sync`
Asks the journal daemon to write all yet unwritten journal data to the backing file system and synchronize all journals. This call does not return until the synchronization operation is complete. This command guarantees that any log messages written before its invocation are safely stored on disk at the time it returns.

`--relinquish-var`
Asks the journal daemon for the reverse operation to `--flush`: if requested the daemon will write further log data to `/run/log/journal/` and stops writing to `/var/log/journal/`. A subsequent call to `--flush` causes the log output to switch back to `/var/log/journal/`, see above.

`--smart-relinquish-var`
Similar to `--relinquish-var`, but executes no operation if the root file system and `/var/log/journal/` reside on the same mount point. This operation is used during system shutdown in order to make the journal daemon stop writing data to `/var/log/journal/` in case that directory is located on a mount point that needs to be unmounted.

`--flush`
Asks the journal daemon to flush any log data stored 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 this call is idempotent: the data is only flushed from `/run/log/journal/` into `/var/log/journal/` once during system runtime (but see `--relinquish-var` below), and this command exits cleanly without executing any operation if this has already happened. This command effectively guarantees that all data is flushed to `/var/log/journal/` at the time it returns.

`--rotate`
Asks the journal daemon to rotate journal files. This call does not return until the rotation operation is complete. Journal file rotation has the effect that all currently active journal files are marked as archived and renamed, so that they are never written to in future. New (empty) journal files are then created in their place. This operation may be combined with `--vacuum-size=`, `--vacuum-time=` and `--vacuum-file=` into a single command, see above.

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

This option is particularly useful when trying to identify out-of-order journal entries, as happens for example when the machine is booted with the wrong system time.

`--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.

Exit status

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

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
journalctl _SYSTEMD_CGROUP=/user.slice/user-42.slice/session-c1.scope
```

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
```

To show all fields emitted *by* a unit and *about* the unit, option `-u`/`--unit=` should be used. `journalctl -u name` expands to a complex filter similar to

```
_SYSTEMD_UNIT=name.service
  + UNIT=name.service _PID=1
  + OBJECT_SYSTEMD_UNIT=name.service _UID=0
  + COREDUMP_UNIT=name.service _UID=0 MESSAGE_ID=fc2e22bc6ee647b6b90729ab34a250b1
```

(see `systemd.journal-fields(7)` for an explanation of those patterns).

Show all logs generated by the D-Bus executable:

```
journalctl /usr/bin/dbus-daemon
```

Show all kernel logs from previous boot:

```
journalctl -k -b -1
```

Show a live log display from a system service `apache.service`:

```
journalctl -f -u apache
```

See Also

`systemd(1)`, `systemd-cat(1)`, `systemd-journald.service(8)`, `systemctl(1)`, `coredumpctl(1)`, `systemd.journal-fields(7)`, `journald.conf(5)`, `systemd.time(7)`, `systemd-journal-remote.service(8)`, `systemd-journal-upload.service(8)`

□□□ #1
□ □□□ □□□ 2026-10-31 14:46:40 UTC
□ □□□ □□□ 2026-10-31 14:46:40 UTC