NAME

journald.conf - Journal service configuration file

SYNOPSIS

/etc/systemd/journald.conf

DESCRIPTION

This file configures various parameters of the systemd journal service, **systemd-journald.service**(8).

OPTIONS

All options are configured in the [Journal] section:

Storage=

Controls where to store journal data. One of volatile, persistent, auto and none. If volatile, journal log data will be stored only in memory, i.e. below the /run/log/journal hierarchy (which is created if needed). If persistent, data will be stored preferably on disk, i.e. below the /var/log/journal hierarchy (which is created if needed), with a fallback to /run/log/journal (which is created if needed), during early boot and if the disk is not writable. auto is similar to persistent but the directory /var/log/journal is not created if needed, so that its existence controls where log data goes. none turns off all storage, all log data received will be dropped. Forwarding to other targets, such as the console, the kernel log buffer or a syslog daemon will still work however. Defaults to auto.

Compress=

Takes a boolean value. If enabled (the default), data objects that shall be stored in the journal and are larger than a certain threshold are compressed with the XZ compression algorithm before they are written to the file system.

Seal=

Takes a boolean value. If enabled (the default), and a sealing key is available (as created by **journalctl(1)**s **--setup-keys** command), Forward Secure Sealing (FSS) for all persistent journal files is enabled. FSS is based on **Seekable Sequential Key Generators**[1] by G. A. Marson and B. Poettering (doi:10.1007/978-3-642-40203-6_7) and may be used to protect journal files from unnoticed alteration.

SplitMode=

Controls whether to split up journal files per user. One of uid, login and none. If uid, all users will get each their own journal files regardless of whether they possess a login session or not, however system users will log into the system journal. If login, actually logged-in users will get each their own journal files, but users without login session and system users will log into the system journal. If none, journal files are not split up by user and all messages are instead stored in the single system journal. Note that splitting up journal files by user is only available for journals stored persistently. If journals are stored on volatile storage (see above), only a single journal file for all user IDs is kept. Defaults to uid.

RateLimitInterval=, *RateLimitBurst=*

Configures the rate limiting that is applied to all messages generated on the system. If, in the time interval defined by *RateLimitInterval=*, more messages than specified in *RateLimitBurst=* are logged by a service, all further messages within the interval are dropped until the interval is over. A message about the number of dropped messages is generated. This rate limiting is applied per-service, so that two services which log do not interfere with each others limits. Defaults to 1000 messages in 30s. The time specification for *RateLimitInterval=* may be specified in the following units: s, min, h, ms, us. To turn off any kind of rate limiting, set either value to 0.

SystemMaxUse=, *SystemKeepFree=*, *SystemMaxFileSize=*, *RuntimeMaxUse=*, *RuntimeKeepFree=*, *RuntimeMaxFileSize=*

Enforce size limits on the journal files stored. The options prefixed with System apply to the journal files when stored on a persistent file system, more specifically /var/log/journal. The options prefixed with Runtime apply to the journal files when stored on a volatile in-memory file system, more specifically /run/log/journal. The former is used only when /var is mounted, writable, and the directory /var/log/journal exists. Otherwise, only the latter applies. Note that this means that during early boot and if the administrator disabled persistent logging, only the latter options apply, while the former apply if persistent logging is enabled and the system is fully booted up. **journalctl** and **systemd-journald** ignore all files with names not ending with .journal or .journal˜, so only such files, located in the appropriate directories, are taken into account when calculating current disk usage.

SystemMaxUse= and *RuntimeMaxUse=* control how much disk space the journal may use up at maximum. *SystemKeepFree=* and *RuntimeKeepFree=* control how much disk space systemd-journald shall leave free for other uses. **systemd-journald** will respect both limits and use the smaller of the two values.

The first pair defaults to 10% and the second to 15% of the size of the respective file system. If the file system is nearly full and either *SystemKeepFree=* or *RuntimeKeepFree=* is violated when systemd-journald is started, the value will be raised to percentage that is actually free. This means that if there was enough free space before and journal files were created, and subsequently something else causes the file system to fill up, journald will stop using more space, but it will not be removing existing files to go reduce footprint either.

SystemMaxFileSize= and *RuntimeMaxFileSize=* control how large individual journal files may grow at maximum. This influences the granularity in which disk space is made available through rotation, i.e. deletion of historic data. Defaults to one eighth of the values configured with *SystemMaxUse=* and *RuntimeMaxUse=*, so that usually seven rotated journal files are kept as history. Specify values in bytes or use K, M, G, T, P, E as units for the specified sizes (equal to 1024, 1024,... bytes). Note that size limits are enforced synchronously when journal files are extended, and no explicit rotation step triggered by time is needed.

MaxFileSec=

The maximum time to store entries in a single journal file before rotating to the next one. Normally, time-based rotation should not be required as size-based rotation with options such as *SystemMaxFileSize=* should be sufficient to ensure that journal files do not grow without bounds. However, to ensure that not too much data is lost at once when old journal files are deleted, it might make sense to change this value from the default of one month. Set to 0 to turn off this feature. This setting takes time values which may be suffixed with the units year, month, week, day, h or m to override the default time unit of seconds.

MaxRetentionSec=

The maximum time to store journal entries. This controls whether journal files containing entries older then the specified time span are deleted. Normally, time-based deletion of old journal files should not be required as size-based deletion with options such as *SystemMaxUse=* should be sufficient to ensure that journal files do not grow without bounds. However, to enforce data retention policies, it might make sense to change this value from the default of 0 (which turns off this feature). This setting also takes time values which may be suffixed with the units year, month, week, day, h or m to override the default time unit of seconds.

SyncIntervalSec=

The timeout before synchronizing journal files to disk. After syncing, journal files are placed in the OFFLINE state. Note that syncing is unconditionally done immediately after a log message of priority CRIT, ALERT or EMERG has been logged. This setting hence applies only to messages of the levels ERR, WARNING, NOTICE, INFO, DEBUG. The default timeout is 5 minutes.

ForwardToSyslog=, *ForwardToKMsg=*, *ForwardToConsole=*, *ForwardToWall=*

Control whether log messages received by the journal daemon shall be forwarded to a traditional syslog daemon, to the kernel log buffer (kmsg), to the system console, or sent as wall messages to all logged-in users. These options take boolean arguments. If forwarding to syslog is enabled but no syslog daemon is running, the respective option has no effect. By default, only forwarding to syslog and wall is enabled. These settings may be overridden at boot time with the kernel command line options systemd.journald.forward_to_syslog=, systemd.journald.forward_to_kmsg=, systemd.journald.forward_to_console= and systemd.journald.forward_to_wall=. When forwarding to the console, the TTY to log to can be changed with *TTYPath=*, described below.

MaxLevelStore=, *MaxLevelSyslog=*, *MaxLevelKMsg=*, *MaxLevelConsole=*, *MaxLevelWall=*

Controls the maximum log level of messages that are stored on disk, forwarded to syslog, kmsg, the console or wall (if that is enabled, see above). As argument, takes one of emerg, alert, crit, err, warning, notice, info, debug or integer values in the range of 0..7 (corresponding to the same levels). Messages equal or below the log level specified are stored/forwarded, messages above are dropped. Defaults to debug for *MaxLevelStore=* and *MaxLevelSyslog=*, to ensure that the all messages are written to disk and forwarded to syslog. Defaults to notice for *MaxLevelKMsg=*, info for *MaxLevelConsole=* and emerg for *MaxLevelWall=*.

TTYPath=

Change the console TTY to use if *ForwardToConsole=yes* is used. Defaults to /dev/console.

SEE ALSO

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

NOTES

1. Seekable Sequential Key Generators
   https://eprint.iacr.org/2013/397