Rocky Enterprise Linux 9.2 Manual Pages on command 'systemd-system.conf.5'

$ man systemd-system.conf.5

SYSTEMD-SYSTEM.CONF(5) systemd-system.conf SYSTEMD-SYSTEM.CONF(5)

NAME

systemd-system.conf, system.conf.d, systemd-user.conf, user.conf.d -

System and session service manager configuration files

SYNOPSIS

/etc/systemd/system.conf, /etc/systemd/system.conf.d/*.conf,

/run/systemd/system.conf.d/*.conf,

/usr/lib/systemd/system.conf.d/*.conf

~/.config/systemd/user.conf, /etc/systemd/user.conf,

/etc/systemd/user.conf.d/*.conf, /run/systemd/user.conf.d/*.conf,

/usr/lib/systemd/user.conf.d/*.conf

DESCRIPTION

When run as a system instance, systemd interprets the configuration

file system.conf and the files in system.conf.d directories; when run

as a user instance, it interprets the configuration file user.conf

(either in the home directory of the user, or if not found, under

/etc/systemd/) and the files in user.conf.d directories. These

configuration files contain a few settings controlling basic manager

operations.

See systemd.syntax(7) for a general description of the syntax.

CONFIGURATION DIRECTORIES AND PRECEDENCE

The default configuration is set during compilation, so configuration is only needed when it is necessary to deviate from those defaults. Initially, the main configuration file in /etc/systemd/ contains commented out entries showing the defaults as a guide to the administrator. Local overrides can be created by editing this file or by creating drop-ins, as described below. Using drop-ins for local configuration is recommended over modifications to the main configuration file.

In addition to the "main" configuration file, drop-in configuration snippets are read from /usr/lib/systemd/*.conf.d/, /usr/local/lib/systemd/*.conf.d/, and /etc/systemd/*.conf.d/. Those drop-ins have higher precedence and override the main configuration file. Files in the *.conf.d/ configuration subdirectories are sorted by their filename in lexicographic order, regardless of in which of the subdirectories they reside. When multiple files specify the same option, for options which accept just a single value, the entry in the file sorted last takes precedence, and for options which accept a list of values, entries are collected as they occur in the sorted files.

When packages need to customize the configuration, they can install drop-ins under /usr/. Files in /etc/ are reserved for the local administrator, who may use this logic to override the configuration files installed by vendor packages. Drop-ins have to be used to override package drop-ins, since the main configuration file has lower precedence. It is recommended to prefix all filenames in those subdirectories with a two-digit number and a dash, to simplify the ordering of the files.

To disable a configuration file supplied by the vendor, the recommended way is to place a symlink to /dev/null in the configuration directory in /etc/, with the same filename as the vendor configuration file.

OPTIONS

All options are configured in the [Manager] section:

LogColor=, LogLevel=, LogLocation=, LogTarget=, LogTime=, DumpCore=yes, CrashChangeVT=no, CrashShell=no, CrashReboot=no, ShowStatus=yes, DefaultStandardOutput=journal, DefaultStandardError=inherit

Configures various parameters of basic manager operation. These options may be overridden by the respective process and kernel command line arguments. See systemd(1) for details.

CtrlAltDelBurstAction=

Defines what action will be performed if user presses Ctrl-Alt-Delete more than 7 times in 2s. Can be set to "reboot-force", "poweroff-force", "reboot-immediate", "poweroff-immediate" or disabled with "none". Defaults to "reboot-force".

CPUAffinity=

Configures the CPU affinity for the service manager as well as the default CPU affinity for all forked off processes. Takes a list of CPU indices or ranges separated by either whitespace or commas. CPU ranges are specified by the lower and upper CPU indices separated by a dash. This option may be specified more than once, in which case the specified CPU affinity masks are merged. If the empty string is assigned, the mask is reset, all assignments prior to this will have no effect. Individual services may override the CPU affinity for their processes with the CPUAffinity= setting in unit files, see systemd.exec(5).

NUMAPolicy=

Configures the NUMA memory policy for the service manager and the default NUMA memory policy for all forked off processes. Individual services may override the default policy with the NUMAPolicy= setting in unit files, see systemd.exec(5).

NUMAMask=

Configures the NUMA node mask that will be associated with the selected NUMA policy. Note that default and local NUMA policies don't require explicit NUMA node mask and value of the option can

be empty. Similarly to NUMAPolicy=, value can be overridden by individual services in unit files, see systemd.exec(5).

RuntimeWatchdogSec=, RebootWatchdogSec=, KExecWatchdogSec=

Configure the hardware watchdog at runtime and at reboot. Takes a timeout value in seconds (or in other time units if suffixed with "ms", "min", "h", "d", "w"), or the special strings "off" or "default". If set to "off" (alternatively: "0") the watchdog logic is disabled: no watchdog device is opened, configured, or pinged. If set to the special string "default" the watchdog is opened and pinged in regular intervals, but the timeout is not changed from the default. If set to any other time value the watchdog timeout is configured to the specified value (or a value close to it, depending on hardware capabilities).

If RuntimeWatchdogSec= is set to a non-zero value, the watchdog hardware (/dev/watchdog0 or the path specified with WatchdogDevice= or the kernel option systemd.watchdog-device=) will be programmed to automatically reboot the system if it is not contacted within the specified timeout interval. The system manager will ensure to contact it at least once in half the specified timeout interval.

This feature requires a hardware watchdog device to be present, as it is commonly the case in embedded and server systems. Not all hardware watchdogs allow configuration of all possible reboot timeout values, in which case the closest available timeout is picked.

RebootWatchdogSec= may be used to configure the hardware watchdog when the system is asked to reboot. It works as a safety net to ensure that the reboot takes place even if a clean reboot attempt times out. Note that the RebootWatchdogSec= timeout applies only to the second phase of the reboot, i.e. after all regular services are already terminated, and after the system and service manager process (PID 1) got replaced by the systemd-shutdown binary, see system bootup(7) for details. During the first phase of the shutdown operation the system and service manager remains running

and hence RuntimeWatchdogSec= is still honoured. In order to define a timeout on this first phase of system shutdown, configure JobTimeoutSec= and JobTimeoutAction= in the [Unit] section of the shutdown.target unit. By default RuntimeWatchdogSec= defaults to 0 (off), and RebootWatchdogSec= to 10min.

KExecWatchdogSec= may be used to additionally enable the watchdog when kexec is being executed rather than when rebooting. Note that if the kernel does not reset the watchdog on kexec (depending on the specific hardware and/or driver), in this case the watchdog might not get disabled after kexec succeeds and thus the system might get rebooted, unless RuntimeWatchdogSec= is also enabled at the same time. For this reason it is recommended to enable KExecWatchdogSec= only if RuntimeWatchdogSec= is also enabled.

These settings have no effect if a hardware watchdog is not available.

RuntimeWatchdogPreSec=

Configure the hardware watchdog device pre-timeout value. Takes a timeout value in seconds (or in other time units similar to RuntimeWatchdogSec=). A watchdog pre-timeout is a notification generated by the watchdog before the watchdog reset might occur in the event the watchdog has not been serviced. This notification is handled by the kernel and can be configured to take an action (i.e. generate a kernel panic) using RuntimeWatchdogPreGovernor=. Not all watchdog hardware or drivers support generating a pre-timeout and depending on the state of the system, the kernel may be unable to take the configured action before the watchdog reboot. The watchdog will be configured to generate the pre-timeout event at the amount of time specified by RuntimeWatchdogPreSec= before the runtime watchdog timeout (set by RuntimeWatchdogSec=). For example, if the we have RuntimeWatchdogSec=30 and RuntimeWatchdogPreSec=10, then the pre-timeout event will occur if the watchdog has not pinged for 20s (10s before the watchdog would fire). By default, RuntimeWatchdogPreSec= defaults to 0 (off). The value set for

RuntimeWatchdogPreSec= must be smaller than the timeout value for RuntimeWatchdogSec=. This setting has no effect if a hardware watchdog is not available or the hardware watchdog does not support a pre-timeout and will be ignored by the kernel if the setting is greater than the actual watchdog timeout.

RuntimeWatchdogPreGovernor=

Configure the action taken by the hardware watchdog device when the pre-timeout expires. The default action for the pre-timeout event depends on the kernel configuration, but it is usually to log a kernel message. For a list of valid actions available for a given watchdog device, check the content of the /sys/class/watchdog/watchdogX/pretimeout_available_governors file. Typically, available governor types are noop and panic.

Availability, names and functionality might vary depending on the specific device driver in use. If the pretimeout_available_governors sysfs file is empty, the governor might be built as a kernel module and might need to be manually loaded (e.g. pretimeout_noop.ko), or the watchdog device might not support pre-timeouts.

WatchdogDevice=

Configure the hardware watchdog device that the runtime and shutdown watchdog timers will open and use. Defaults to /dev/watchdog0. This setting has no effect if a hardware watchdog is not available.

CapabilityBoundingSet=

Controls which capabilities to include in the capability bounding set for PID 1 and its children. See capabilities(7) for details.

Takes a whitespace-separated list of capability names as read by cap_from_name(3). Capabilities listed will be included in the bounding set, all others are removed. If the list of capabilities is prefixed with ~, all but the listed capabilities will be included, the effect of the assignment inverted. Note that this option also affects the respective capabilities in the effective,

permitted and inheritable capability sets. The capability bounding set may also be individually configured for units using the CapabilityBoundingSet= directive for units, but note that capabilities dropped for PID 1 cannot be regained in individual units, they are lost for good.

NoNewPrivileges=

Takes a boolean argument. If true, ensures that PID 1 and all its children can never gain new privileges through execve(2) (e.g. via setuid or setgid bits, or filesystem capabilities). Defaults to false. General purpose distributions commonly rely on executables with setuid or setgid bits and will thus not function properly with this option enabled. Individual units cannot disable this option. Also see No New Privileges Flag[1].

SystemCallArchitectures=

Takes a space-separated list of architecture identifiers. Selects from which architectures system calls may be invoked on this system. This may be used as an effective way to disable invocation of non-native binaries system-wide, for example to prohibit execution of 32-bit x86 binaries on 64-bit x86-64 systems. This option operates system-wide, and acts similar to the SystemCallArchitectures= setting of unit files, see systemd.exec(5) for details. This setting defaults to the empty list, in which case no filtering of system calls based on architecture is applied. Known architecture identifiers are "x86", "x86-64", "x32", "arm" and the special identifier "native". The latter implicitly maps to the native architecture of the system (or more specifically, the architecture the system manager was compiled for). Set this setting to "native" to prohibit execution of any non-native binaries. When a binary executes a system call of an architecture that is not listed in this setting, it will be immediately terminated with the SIGSYS signal.

TimerSlackNSec=

Sets the timer slack in nanoseconds for PID 1, which is inherited

by all executed processes, unless overridden individually, for example with the TimerSlackNSec= setting in service units (for details see systemd.exec(5)). The timer slack controls the accuracy of wake-ups triggered by system timers. See prctl(2) for more information. Note that in contrast to most other time span definitions this parameter takes an integer value in nano-seconds if no unit is specified. The usual time units are understood too.

StatusUnitFormat=

Takes name, description or combined as the value. If name, the system manager will use unit names in status messages (e.g. "systemd-journald.service"), instead of the longer and more informative descriptions set with Description= (e.g. "Journal Logging Service"). If combined, the system manager will use both unit names and descriptions in status messages (e.g. "systemd-journald.service - Journal Logging Service").

See systemd.unit(5) for details about unit names and Description=.

DefaultTimerAccuracySec=

Sets the default accuracy of timer units. This controls the global default for the AccuracySec= setting of timer units, see systemd.timer(5) for details. AccuracySec= set in individual units override the global default for the specific unit. Defaults to 1min. Note that the accuracy of timer units is also affected by the configured timer slack for PID 1, see TimerSlackNSec= above.

DefaultTimeoutStartSec=, DefaultTimeoutStopSec=, DefaultTimeoutAbortSec=, DefaultRestartSec=

Configures the default timeouts for starting, stopping and aborting of units, as well as the default time to sleep between automatic restarts of units, as configured per-unit in TimeoutStartSec=, TimeoutStopSec=, TimeoutAbortSec= and RestartSec= (for services, see systemd.service(5) for details on the per-unit settings).

Disabled by default, when service with Type=oneshot is used. For non-service units, DefaultTimeoutStartSec= sets the default TimeoutSec= value. DefaultTimeoutStartSec= and

DefaultTimeoutStopSec= default to 90s. DefaultTimeoutAbortSec= is not set by default so that all units fall back to TimeoutStopSec=. DefaultRestartSec= defaults to 100ms.

DefaultDeviceTimeoutSec=

Configures the default timeout for waiting for devices. It can be changed per device via the x-systemd.device-timeout= option in /etc/fstab and /etc/crypttab (see systemd.mount(5), crypttab(5)). Defaults to 90s.

DefaultStartLimitIntervalSec=, DefaultStartLimitBurst=

Configure the default unit start rate limiting, as configured per-service by StartLimitIntervalSec= and StartLimitBurst=. See systemd.service(5) for details on the per-service settings. DefaultStartLimitIntervalSec= defaults to 10s. DefaultStartLimitBurst= defaults to 5.

DefaultEnvironment=

Configures environment variables passed to all executed processes. Takes a space-separated list of variable assignments. See environ(7) for details about environment variables. Simple "%"-specifier expansion is supported, see below for a list of supported specifiers.

Example:

DefaultEnvironment="VAR1=word1 word2" VAR2=word3 "VAR3=word 5 6"

Sets three variables "VAR1", "VAR2", "VAR3".

ManagerEnvironment=

Takes the same arguments as DefaultEnvironment=, see above. Sets environment variables just for the manager process itself. In contrast to user managers, these variables are not inherited by processes spawned by the system manager, use DefaultEnvironment= for that. Note that these variables are merged into the existing environment block. In particular, in case of the system manager, this includes variables set by the kernel based on the kernel command line.

Setting environment variables for the manager process may be useful

to modify its behaviour. See ENVIRONMENT[2] for a descriptions of some variables understood by systemd.

Simple "%"-specifier expansion is supported, see below for a list of supported specifiers.

DefaultCPUAccounting=, DefaultMemoryAccounting=, DefaultTasksAccounting=, DefaultIOAccounting=, DefaultIPAccounting=

Configure the default resource accounting settings, as configured per-unit by CPUAccounting=, MemoryAccounting=, TasksAccounting=, IOAccounting= and IPAccounting=. See systemd.resource-control(5) for details on the per-unit settings. DefaultTasksAccounting= defaults to yes, DefaultMemoryAccounting= to yes. DefaultCPUAccounting= defaults to yes if enabling CPU accounting doesn't require the CPU controller to be enabled (Linux 4.15+ using the unified hierarchy for resource control), otherwise it defaults to no. The other three settings default to no.

DefaultTasksMax=

Configure the default value for the per-unit TasksMax= setting. See systemd.resource-control(5) for details. This setting applies to all unit types that support resource control settings, with the exception of slice units. Defaults to 80% of the minimum of kernel.pid_max=, kernel.threads-max= and root cgroup pids.max. Kernel has a default value for kernel.pid_max= and an algorithm of counting in case of more than 32 cores. For example with the default kernel.pid_max=, DefaultTasksMax= defaults to 26214, but might be greater in other systems or smaller in OS containers.

DefaultLimitCPU=, DefaultLimitFSIZE=, DefaultLimitDATA=, DefaultLimitSTACK=, DefaultLimitCORE=, DefaultLimitRSS=, DefaultLimitNOFILE=, DefaultLimitAS=, DefaultLimitNPROC=, DefaultLimitMEMLOCK=, DefaultLimitLOCKS=, DefaultLimitSIGPENDING=, DefaultLimitMSGQUEUE=, DefaultLimitNICE=, DefaultLimitRTPRIO=, DefaultLimitRTTIME=

These settings control various default resource limits for processes executed by units. See setrlimit(2) for details. These

settings may be overridden in individual units using the corresponding LimitXXX= directives and they accept the same parameter syntax, see systemd.exec(5) for details. Note that these resource limits are only defaults for units, they are not applied to the service manager process (i.e. PID 1) itself.

Most of these settings are unset, which means the resource limits are inherited from the kernel or, if invoked in a container, from the container manager. However, the following have defaults:

- DefaultLimitNOFILE= defaults to 1024:524288.
- DefaultLimitMEMLOCK= defaults to 8M.
- DefaultLimitCORE= does not have a default but it is worth mentioning that RLIMIT_CORE is set to "infinity" by PID 1 which is inherited by its children.

Note that the service manager internally in PID 1 bumps RLIMIT_NOFILE and RLIMIT_MEMLOCK to higher values, however the limit is reverted to the mentioned defaults for all child processes forked off.

DefaultOOMPolicy=

Configure the default policy for reacting to processes being killed by the Linux Out-Of-Memory (OOM) killer or systemd-oomd. This may be used to pick a global default for the per-unit OOMPolicy= setting. See systemd.service(5) for details. Note that this default is not used for services that have Delegate= turned on.

DefaultOOMScoreAdjust=

Configures the default OOM score adjustments of processes run by the service manager. This defaults to unset (meaning the forked off processes inherit the service manager's OOM score adjustment value), except if the service manager is run for an unprivileged user, in which case this defaults to the service manager's OOM adjustment value plus 100 (this makes service processes slightly more likely to be killed under memory pressure than the manager itself). This may be used to pick a global default for the per-unit OOMScoreAdjust= setting. See systemd.exec(5) for details. Note that

this setting has no effect on the OOM score adjustment value of the service manager process itself, it retains the original value set during its invocation.

DefaultSmackProcessLabel=

Takes a SMACK64 security label as the argument. The process executed by a unit will be started under this label if SmackProcessLabel= is not set in the unit. See systemd.exec(5) for the details.

If the value is "/", only labels specified with SmackProcessLabel= are assigned and the compile-time default is ignored.

SPECIFIERS

Specifiers may be used in the DefaultEnvironment= and ManagerEnvironment= settings. The following expansions are understood:

Table 1. Specifiers available

Specifier	Meaning	Details
"%a"	Architecture	A short string identifying the architecture of the local system. A string such as x86, x86-64 or arm64. See the architectures defined for ConditionArchitecture= in systemd.unit(5) for a full list.
"%A"	Operating system image version	The operating system image version identifier of the

? ? ? running system, as ?

? ? ? read from the ?

? ? ? IMAGE_VERSION= field ?

? ? ? of /etc/os-release. If ?

? ? ? not set, resolves to ?

? ? ? an empty string. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%b" ? Boot ID ? The boot ID of the ?

? ? ? running system, ?

? ? ? formatted as string. ?

? ? ? See random(4) for more ?

? ? ? information. ?

??

?"%B" ? Operating system ? The operating system ?

? ? build ID ? build identifier of ?

? ? ? the running system, as ?

? ? ? read from the ?

? ? ? BUILD_ID= field of ?

? ? ? /etc/os-release. If ?

? ? ? not set, resolves to ?

? ? ? an empty string. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%H" ? Host name ? The hostname of the ?

? ? ? running system. ?

??

?"%l" ? Short host name ? The hostname of the ?

? ? ? running system, ?

? ? ? truncated at the first ?

? ? ? dot to remove any ?

? ? ? domain component. ?

??

?"%m" ? Machine ID ? The machine ID of the ?

? ? ? running system, ?

? ? ? formatted as string. ?

? ? ? See machine-id(5) for ?

? ? ? more information. ?

??

?"%M" ? Operating system ? The operating system ?

? ? image identifier ? image identifier of ?

? ? ? the running system, as ?

? ? ? read from the ?

? ? ? IMAGE_ID= field of ?

? ? ? /etc/os-release. If ?

? ? ? not set, resolves to ?

? ? ? an empty string. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%o" ? Operating system ID ? The operating system ?

? ? ? identifier of the ?

? ? ? running system, as ?

? ? ? read from the ID= ?

? ? ? field of ?

? ? ? /etc/os-release. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%v" ? Kernel release ? Identical to uname -r ?

? ? ? output. ?

??

?"%w" ? Operating system ? The operating system ?

? ? version ID ? version identifier of ?

? ? ? the running system, as ?

? ? ? read from the ?

? ? ? VERSION_ID= field of ?

? ? ? /etc/os-release. If ?

? ? ? not set, resolves to ?

? ? ? an empty string. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%W" ? Operating system ? The operating system ?

? ? variant ID ? variant identifier of ?

? ? ? the running system, as ?

? ? ? read from the ?

? ? ? VARIANT_ID= field of ?

? ? ? /etc/os-release. If ?

? ? ? not set, resolves to ?

? ? ? an empty string. See ?

? ? ? os-release(5) for more ?

? ? ? information. ?

??

?"%T" ? Directory for ? This is either /tmp or ?

? ? temporary files ? the path "$TMPDIR", ?

? ? ? "$TEMP" or "$TMP" are ?

? ? ? set to. (Note that the ?

? ? ? directory may be ?

? ? ? specified without a ?

? ? ? trailing slash.) ?

??

?"%V" ? Directory for ? This is either ?

? ? larger and ? /var/tmp or the path ?

? ? persistent ? "$TMPDIR", "$TEMP" or ?

? ? temporary files ? "$TMP" are set to. ?

? ? ? (Note that the ?

? ? ? directory may be ?

? ? ? specified without a ?

? ? ? trailing slash.) ?

??

?"%%" ? Single percent sign ? Use "%%" in place of ?

? ? ? "%" to specify a ?

? ? ? single percent sign. ?

??

HISTORY

systemd 252

Option DefaultBlockIOAccounting= was deprecated. Please switch to the unified cgroup hierarchy.

SEE ALSO

systemd(1), systemd.directives(7), systemd.exec(5), systemd.service(5), environ(7), capabilities(7)

NOTES

1. No New Privileges Flag

https://docs.kernel.org/userspace-api/no_new_privs.html

2. ENVIRONMENT

https://systemd.io/ENVIRONMENT