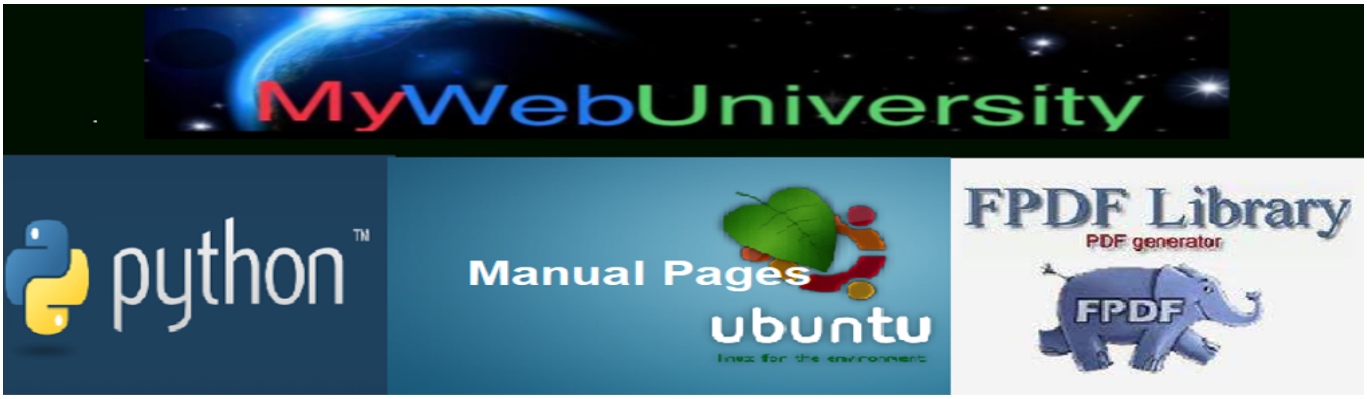

Rocky Enterprise Linux 9.2 Manual Pages on command 'systemd.kill.5'

$ man systemd.kill.5

SYSTEMD.KILL(5) systemd.kill SYSTEMD.KILL(5)

NAME

systemd.kill - Process killing procedure configuration

SYNOPSIS

service.service, socket.socket, mount.mount, swap.swap, scope.scope

DESCRIPTION

Unit configuration files for services, sockets, mount points, swap devices and scopes share a subset of configuration options which define the killing procedure of processes belonging to the unit.

This man page lists the configuration options shared by these five unit types. See systemd.unit(5) for the common options shared by all unit configuration files, and systemd.service(5), systemd.socket(5), systemd.swap(5), systemd.mount(5) and systemd.scope(5) for more information on the configuration file options specific to each unit type.

The kill procedure configuration options are configured in the [Service], [Socket], [Mount] or [Swap] section, depending on the unit type.

OPTIONS

KillMode=

Specifies how processes of this unit shall be killed. One of control-group, mixed, process, none.

If set to control-group, all remaining processes in the control group of this unit will be killed on unit stop (for services: after the stop command is executed, as configured with ExecStop=). If set to mixed, the SIGTERM signal (see below) is sent to

the main process while the subsequent SIGKILL signal (see below) is sent to all remaining processes of the unit's control group. If set to process, only the main process itself is killed (not recommended!). If set to none, no process is killed (strongly recommended against!). In this case, only the stop command will be executed on unit stop, but no process will be killed otherwise. Processes remaining alive after stop are left in their control group and the control group continues to exist after stop unless empty.

Note that it is not recommended to set KillMode= to process or even none, as this allows processes to escape the service manager's lifecycle and resource management, and to remain running even while their service is considered stopped and is assumed to not consume any resources.

Processes will first be terminated via SIGTERM (unless the signal to send is changed via KillSignal= or RestartKillSignal=). Optionally, this is immediately followed by a SIGHUP (if enabled with SendSIGHUP=). If processes still remain after the main process of a unit has exited or the delay configured via the TimeoutStopSec= has passed, the termination request is repeated with the SIGKILL signal or the signal specified via FinalKillSignal= (unless this is disabled via the SendSIGKILL= option). See kill(2) for more information.

Defaults to control-group.

KillSignal=

Specifies which signal to use when stopping a service. This controls the signal that is sent as first step of shutting down a unit (see above), and is usually followed by SIGKILL (see above and below). For a list of valid signals, see signal(7). Defaults to SIGTERM.

Note that, right after sending the signal specified in this setting, systemd will always send SIGCONT, to ensure that even suspended tasks can be terminated cleanly.

RestartKillSignal=

Specifies which signal to use when restarting a service. The same as KillSignal= described above, with the exception that this setting is used in a restart job. Not set by default, and the value of KillSignal= is used.

SendSIGHUP=

Specifies whether to send SIGHUP to remaining processes immediately after sending the signal configured with KillSignal=. This is useful to indicate to shells and

shell-like programs that their connection has been severed. Takes a boolean value. Defaults to "no".

SendSIGKILL=

Specifies whether to send SIGKILL (or the signal specified by FinalKillSignal=) to remaining processes after a timeout, if the normal shutdown procedure left processes of the service around. When disabled, a KillMode= of control-group or mixed service will not restart if processes from prior services exist within the control group. Takes a boolean value. Defaults to "yes".

FinalKillSignal=

Specifies which signal to send to remaining processes after a timeout if SendSIGKILL= is enabled. The signal configured here should be one that is not typically caught and processed by services (SIGTERM is not suitable). Developers can find it useful to use this to generate a coredump to troubleshoot why a service did not terminate upon receiving the initial SIGTERM signal. This can be achieved by configuring LimitCORE= and setting FinalKillSignal= to either SIGQUIT or SIGABRT. Defaults to SIGKILL.

WatchdogSignal=

Specifies which signal to use to terminate the service when the watchdog timeout expires (enabled through WatchdogSec=). Defaults to SIGABRT.

SEE ALSO

systemd(1), systemctl(1), journalctl(1), systemd.unit(5), systemd.service(5), systemd.socket(5), systemd.swap(5), systemd.mount(5), systemd.exec(5), systemd.directives(7), kill(2), signal(7)