## NAME

logind.conf, logind.conf.d − Login manager configuration files

## SYNOPSIS

/etc/systemd/logind.conf

/etc/systemd/logind.conf.d/*.conf

/run/systemd/logind.conf.d/*.conf

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

## DESCRIPTION

These files configure various parameters of the systemd login manager, **systemd-logind.service**(8). See **systemd.syntax**(7) for a general description of the syntax.

## CONFIGURATION DIRECTORIES AND PRECEDENCE

The default configuration is defined during compilation, so a configuration file is only needed when it is necessary to deviate from those defaults. By default, the configuration file in /etc/systemd/ contains commented out entries showing the defaults as a guide to the administrator. This file can be edited to create local overrides.

When packages need to customize the configuration, they can install configuration snippets in /usr/lib/systemd/*.conf.d/ or /usr/local/lib/systemd/*.conf.d/. The main configuration file is read before any of the configuration directories, and has the lowest precedence; entries in a file in any configuration directory override entries in the single 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 with the lexicographically latest name takes precedence. For options which accept a list of values, entries are collected as they occur in files sorted lexicographically.

Files in /etc/ are reserved for the local administrator, who may use this logic to override the configuration files installed by vendor packages. 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 "[Login]" section:

*NAutoVTs=*

Takes a positive integer. Configures how many virtual terminals (VTs) to allocate by default that, when switched to and are previously unused, "autovt" services are automatically spawned on. These services are instantiated from the template unit autovt@.service for the respective VT TTY name, for example, autovt@tty4.service. By default, autovt@.service is linked to getty@.service. In other words, login prompts are started dynamically as the user switches to unused virtual terminals. Hence, this parameter controls how many login "gettys" are available on the VTs. If a VT is already used by some other subsystem (for example, a graphical login), this kind of activation will not be attempted. Note that the VT configured in *ReserveVT=* is always subject to this kind of activation, even if it is not one of the VTs configured with the *NAutoVTs=* directive. Defaults to 6. When set to 0, automatic spawning of "autovt" services is disabled.

*ReserveVT=*

Takes a positive integer. Identifies one virtual terminal that shall unconditionally be reserved for autovt@.service activation (see above). The VT selected with this option will be marked busy unconditionally, so that no other subsystem will allocate it. This functionality is useful to ensure that, regardless of how many VTs are allocated by other subsystems, one login "getty" is always available. Defaults to 6 (in other words, there will always be a "getty" available on Alt−F6.). When set to 0, VT reservation is disabled.

*KillUserProcesses=*

Takes a boolean argument. Configures whether the processes of a user should be killed when the user logs out. If true, the scope unit corresponding to the session and all processes inside that scope will be terminated. If false, the scope is "abandoned", see **systemd.scope**(5), and processes are not killed. Defaults to "no", but see the options *KillOnlyUsers*= and *KillExcludeUsers*= below.

In addition to session processes, user process may run under the user manager unit user@.service. Depending on the linger settings, this may allow users to run processes independent of their login sessions. See the description of **enable−linger** in **loginctl**(1).

Note that setting *KillUserProcesses*=*yes* will break tools like **screen**(1) and **tmux**(1), unless they are moved out of the session scope. See example in **systemd-run**(1).

*KillOnlyUsers*=, *KillExcludeUsers*=
: These settings take space−separated lists of usernames that override the *KillUserProcesses*= setting. A user name may be added to *KillExcludeUsers*= to exclude the processes in the session scopes of that user from being killed even if *KillUserProcesses*=*yes* is set. If *KillExcludeUsers*= is not set, the "root" user is excluded by default. *KillExcludeUsers*= may be set to an empty value to override this default. If a user is not excluded, *KillOnlyUsers*= is checked next. If this setting is specified, only the session scopes of those users will be killed. Otherwise, users are subject to the *KillUserProcesses*=*yes* setting.

*IdleAction*=
: Configures the action to take when the system is idle. Takes one of "ignore", "poweroff", "reboot", "halt", "kexec", "suspend", "hibernate", "hybrid−sleep", "suspend−then−hibernate", and "lock". Defaults to "ignore".

Note that this requires that user sessions correctly report the idle status to the system. The system will execute the action after all sessions report that they are idle, no idle inhibitor lock is active, and subsequently, the time configured with *IdleActionSec*= (see below) has expired.

*IdleActionSec*=
: Configures the delay after which the action configured in *IdleAction*= (see above) is taken after the system is idle.

*InhibitDelayMaxSec*=
: Specifies the maximum time a system shutdown or sleep request is delayed due to an inhibitor lock of type "delay" being active before the inhibitor is ignored and the operation executes anyway. Defaults to 5.

*UserStopDelaySec*=
: Specifies how long to keep the user record and per−user service user@.service around for a user after they logged out fully. If set to zero, the per−user service is terminated immediately when the last session of the user has ended. If this option is configured to non−zero rapid logout/login cycles are sped up, as the user's service manager is not constantly restarted. If set to "infinity" the per−user service for a user is never terminated again after first login, and continues to run until system shutdown. Defaults to 10s.

*HandlePowerKey*=, *HandleSuspendKey*=, *HandleHibernateKey*=, *HandleLidSwitch*=, *HandleLidSwitchExternalPower*=, *HandleLidSwitchDocked*=
: Controls how logind shall handle the system power and sleep keys and the lid switch to trigger actions such as system power−off or suspend. Can be one of "ignore", "poweroff", "reboot", "halt", "kexec", "suspend", "hibernate", "hybrid−sleep", "suspend−then−hibernate", and "lock". If "ignore", logind will never handle these keys. If "lock", all running sessions will be screen−locked; otherwise, the specified action will be taken in the respective event. Only input devices with the "power−switch" udev tag will be watched for key/lid switch events. *HandlePowerKey*= defaults to "poweroff". *HandleSuspendKey*= and *HandleLidSwitch*= default to "suspend". *HandleLidSwitchExternalPower*= is completely ignored by default (for backwards compatibility) — an explicit value must be set before it will be used to determine behaviour. *HandleLidSwitchDocked*= defaults to "ignore". *HandleHibernateKey*= defaults to "hibernate". If the system is inserted in a docking station, or if more

than one display is connected, the action specified by *HandleLidSwitchDocked*= occurs; if the system is on external power the action (if any) specified by *HandleLidSwitchExternalPower*= occurs; otherwise the *HandleLidSwitch*= action occurs.

A different application may disable logind's handling of system power and sleep keys and the lid switch by taking a low−level inhibitor lock ("handle−power−key", "handle−suspend−key", "handle−hibernate−key", "handle−lid−switch"). This is most commonly used by graphical desktop environments to take over suspend and hibernation handling, and to use their own configuration mechanisms. If a low−level inhibitor lock is taken, logind will not take any action when that key or switch is triggered and the *Handle**= settings are irrelevant.

*PowerKeyIgnoreInhibited*=, *SuspendKeyIgnoreInhibited*=, *HibernateKeyIgnoreInhibited*=, *LidSwitchIgnoreInhibited*=

Controls whether actions that **systemd−logind** takes when the power and sleep keys and the lid switch are triggered are subject to high−level inhibitor locks ("shutdown", "sleep", "idle"). Low level inhibitor locks ("handle−power−key", "handle−suspend−key", "handle−hibernate−key", "handle−lid−switch"), are always honored, irrespective of this setting.

These settings take boolean arguments. If "no", the inhibitor locks taken by applications are respected. If "yes", "shutdown", "sleep", and "idle" inhibitor locks are ignored. *PowerKeyIgnoreInhibited*=, *SuspendKeyIgnoreInhibited*=, and *HibernateKeyIgnoreInhibited*= default to "no". *LidSwitchIgnoreInhibited*= defaults to "yes". This means that when **systemd−logind** is handling events by itself (no low level inhibitor locks are taken by another application), the lid switch does not respect suspend blockers by default, but the power and sleep keys do.

*HoldoffTimeoutSec*=

Specifies the timeout after system startup or system resume in which systemd will hold off on reacting to lid events. This is required for the system to properly detect any hotplugged devices so systemd can ignore lid events if external monitors, or docks, are connected. If set to 0, systemd will always react immediately, possibly before the kernel fully probed all hotplugged devices. This is safe, as long as you do not care for systemd to account for devices that have been plugged or unplugged while the system was off. Defaults to 30s.

*RuntimeDirectorySize*=

Sets the size limit on the *$XDG_RUNTIME_DIR* runtime directory for each user who logs in. Takes a size in bytes, optionally suffixed with the usual K, G, M, and T suffixes, to the base 1024 (IEC). Alternatively, a numerical percentage suffixed by "%" may be specified, which sets the size limit relative to the amount of physical RAM. Defaults to 10%. Note that this size is a safety limit only. As each runtime directory is a tmpfs file system, it will only consume as much memory as is needed.

*InhibitorsMax*=

Controls the maximum number of concurrent inhibitors to permit. Defaults to 8192 (8K).

*SessionsMax*=

Controls the maximum number of concurrent user sessions to manage. Defaults to 8192 (8K). Depending on how the pam_systemd.so module is included in the PAM stack configuration, further login sessions will either be refused, or permitted but not tracked by systemd−logind.

*RemoveIPC*=

Controls whether System V and POSIX IPC objects belonging to the user shall be removed when the user fully logs out. Takes a boolean argument. If enabled, the user may not consume IPC resources after the last of the user's sessions terminated. This covers System V semaphores, shared memory and message queues, as well as POSIX shared memory and message queues. Note that IPC objects of the root user and other system users are excluded from the effect of this setting. Defaults to "yes".

## SEE ALSO

**systemd**(1), **systemd-logind.service**(8), **loginctl**(1), **systemd-system.conf**(5)