NAME

systemd.special − Special systemd units

SYNOPSIS

basic.target, bluetooth.target, cryptsetup−pre.target, cryptsetup.target, veritysetup−pre.target, veritysetup.target, ctrl−alt−del.target, blockdev@.target, boot−complete.target, default.target, emergency.target, exit.target, final.target, first−boot−complete.target, getty.target, getty−pre.target, graphical.target, halt.target, hibernate.target, hybrid−sleep.target, suspend−then−hibernate.target, initrd.target, initrd−fs.target, initrd−root−device.target, initrd−root−fs.target, initrd−usr−fs.target, kbrequest.target, kexec.target, local−fs−pre.target, local−fs.target, machines.target multi−user.target, network−online.target, network−pre.target, network.target, nss−lookup.target, nss−user−lookup.target, paths.target, poweroff.target, printer.target, reboot.target, remote−cryptsetup.target, remote−veritysetup.target, remote−fs−pre.target, remote−fs.target, rescue.target, rpcbind.target, runlevel2.target, runlevel3.target, runlevel4.target, runlevel5.target, shutdown.target, sigpwr.target, sleep.target, slices.target, smartcard.target, sockets.target, sound.target, suspend.target, swap.target, sysinit.target, system−update.target, system−update−pre.target, time−set.target, time−sync.target, timers.target, umount.target, usb−gadget.target, −.slice, system.slice, user.slice, machine.slice, −.mount, dbus.service, dbus.socket, display−manager.service, init.scope, syslog.socket, system−update−cleanup.service

DESCRIPTION

A few units are treated specially by systemd. Many of them have special internal semantics and cannot be renamed, while others simply have a standard meaning and should be present on all systems.

UNITS MANAGED BY THE SYSTEM SERVICE MANAGER

Special System Units

−.mount

The root mount point, i.e. the mount unit for the / path. This unit is unconditionally active, during the entire time the system is up, as this mount point is where the basic userspace is running from.

basic.target

A special target unit covering basic boot−up.

systemd automatically adds dependency of the type *After=* for this target unit to all services (except for those with *DefaultDependencies=no*).

Usually, this should pull−in all local mount points plus /var/, /tmp/ and /var/tmp/, swap devices, sockets, timers, path units and other basic initialization necessary for general purpose daemons. The mentioned mount points are special cased to allow them to be remote.

This target usually does not pull in any non−target units directly, but rather does so indirectly via other early boot targets. It is instead meant as a synchronization point for late boot services. Refer to **bootup**(7) for details on the targets involved.

boot−complete.target

This target is intended as generic synchronization point for services that shall determine or act on whether the boot process completed successfully. Order units that are required to succeed for a boot process to be considered successful before this unit, and add a *Requires=* dependency from the target unit to them. Order units that shall only run when the boot process is considered successful after the target unit and pull in the target from it, also with *Requires=*. Note that by default this target unit is not part of the initial boot transaction, but is supposed to be pulled in only if required by units that want to run only on successful boots.

See **systemd-boot-check-no-failures.service**(8) for a service that implements a generic system health check and orders itself before boot−complete.target.

See **systemd-bless-boot.service**(8) for a service that propagates boot success information to the boot

loader, and orders itself after boot−complete.target.

ctrl−alt−del.target
: systemd starts this target whenever Control+Alt+Del is pressed on the console. Usually, this should be aliased (symlinked) to reboot.target.

cryptsetup.target
: A target that pulls in setup services for all encrypted block devices.

veritysetup.target
: A target that pulls in setup services for all verity integrity protected block devices.

dbus.service
: A special unit for the D−Bus bus daemon. As soon as this service is fully started up systemd will connect to it and register its service.

dbus.socket
: A special unit for the D−Bus system bus socket. All units with *Type=dbus* automatically gain a dependency on this unit.

default.target
: The default unit systemd starts at bootup. Usually, this should be aliased (symlinked) to multi−user.target or graphical.target. See **bootup**(7) for more discussion.

  The default unit systemd starts at bootup can be overridden with the *systemd.unit=* kernel command line option, or more conveniently, with the short names like *single*, *rescue*, *1*, *3*, *5*, ...; see **systemd**(1).

display−manager.service
: The display manager service. Usually, this should be aliased (symlinked) to gdm.service or a similar display manager service.

emergency.target
: A special target unit that starts an emergency shell on the main console. This target does not pull in other services or mounts. It is the most minimal version of starting the system in order to acquire an interactive shell; the only processes running are usually just the system manager (PID 1) and the shell process. This unit may be used by specifying *emergency* on the kernel command line; it is also used when a file system check on a required file system fails and boot−up cannot continue. Compare with rescue.target, which serves a similar purpose, but also starts the most basic services and mounts all file systems.

  In many ways booting into emergency.target is similar to the effect of booting with "init=/bin/sh" on the kernel command line, except that emergency mode provides you with the full system and service manager, and allows starting individual units in order to continue the boot process in steps.

  Note that depending on how emergency.target is reached, the root file system might be mounted read−only or read−write (no remounting is done specially for this target). For example, the system may boot with root mounted read−only when *ro* is used on the kernel command line and remain this way for emergency.target, or the system may transition to emergency.target after the system has been partially booted and disks have already been remounted read−write.

exit.target
: A special service unit for shutting down the system or user service manager. It is equivalent to poweroff.target on non−container systems, and also works in containers.

  systemd will start this unit when it receives the **SIGTERM** or **SIGINT** signal when running as user service daemon.

  Normally, this (indirectly) pulls in shutdown.target, which in turn should be conflicted by all units that want to be scheduled for shutdown when the service manager starts to exit.

final.target
: A special target unit that is used during the shutdown logic and may be used to pull in late services after all normal services are already terminated and all mounts unmounted.

getty.target
: A special target unit that pulls in statically configured local TTY getty instances.

graphical.target
: A special target unit for setting up a graphical login screen. This pulls in multi−user.target.

  Units that are needed for graphical logins shall add *Wants=* dependencies for their unit to this unit (or multi−user.target) during installation. This is best configured via *WantedBy=graphical.target* in the unit's [Install] section.

hibernate.target
: A special target unit for hibernating the system. This pulls in sleep.target.

hybrid−sleep.target
: A special target unit for hibernating and suspending the system at the same time. This pulls in sleep.target.

suspend−then−hibernate.target
: A special target unit for suspending the system for a period of time, waking it and putting it into hibernate. This pulls in sleep.target.

halt.target
: A special target unit for shutting down and halting the system. Note that this target is distinct from poweroff.target in that it generally really just halts the system rather than powering it down.

  Applications wanting to halt the system should not start this unit directly, but should instead execute **systemctl halt** (possibly with the **−−no−block** option) or call **systemd**(1)'s **org.freedesktop.systemd1.Manager.Halt** D−Bus method directly.

init.scope
: This scope unit is where the system and service manager (PID 1) itself resides. It is active as long as the system is running.

initrd.target
: This is the default target in the initramfs, similar to default.target in the main system. It is used to mount the real root and transition to it. See **bootup**(7) for more discussion.

initrd−fs.target
: **systemd-fstab-generator**(3) automatically adds dependencies of type *Before=* to sysroot−usr.mount and all mount points found in /etc/fstab that have the **x−initrd.mount** mount option set and do not have the **noauto** mount option set. It is also indirectly ordered after sysroot.mount. Thus, once this target is reached the /sysroot/ hierarchy is fully set up, in preparation for the transition to the host OS.

initrd−root−device.target
: A special initrd target unit that is reached when the root filesystem device is available, but before it has been mounted. **systemd-fstab-generator**(3) and **systemd-gpt-auto-generator**(3) automatically setup the appropriate dependencies to make this happen.

initrd−root−fs.target
: **systemd-fstab-generator**(3) automatically adds dependencies of type *Before=* to the sysroot.mount unit, which is generated from the kernel command line's *root=* setting (or equivalent).

initrd−usr−fs.target
: **systemd-fstab-generator**(3) automatically adds dependencies of type *Before=* to the sysusr−usr.mount unit, which is generated from the kernel command line's *usr=* switch. Services may order themselves after this target unit in order to run once the /sysusr/ hierarchy becomes available, on systems that come up initially without a root file system, but with an initialized /usr/ and need to access that before setting up the root file system to ultimately switch to. On systems where *usr=* is not

used this target is ordered after sysroot.mount and thus mostly equivalent to initrd−root−fs.target. In effect on any system once this target is reached the file system backing /usr/ is mounted, though possibly at two different locations, either below the /sysusr/ or the /sysroot/ hierarchies.

kbrequest.target
: systemd starts this target whenever Alt+ArrowUp is pressed on the console. Note that any user with physical access to the machine will be able to do this, without authentication, so this should be used carefully.

kexec.target
: A special target unit for shutting down and rebooting the system via kexec.

  Applications wanting to reboot the system should not start this unit directly, but should instead execute **systemctl kexec** (possibly with the **−−no−block** option) or call **systemd**(1)'s **org.freedesktop.systemd1.Manager.KExec** D−Bus method directly.

local−fs.target
: **systemd-fstab-generator**(3) automatically adds dependencies of type *Before=* to all mount units that refer to local mount points for this target unit. In addition, it adds dependencies of type *Wants=* to this target unit for those mounts listed in /etc/fstab that have the **auto** mount option set.

machines.target
: A standard target unit for starting all the containers and other virtual machines. See systemd−nspawn@.service for an example.

multi−user.target
: A special target unit for setting up a multi−user system (non−graphical). This is pulled in by graphical.target.

  Units that are needed for a multi−user system shall add *Wants=* dependencies for their unit to this unit during installation. This is best configured via *WantedBy=multi−user.target* in the unit's [Install] section.

network−online.target
: Units that strictly require a configured network connection should pull in network−online.target (via a *Wants=* type dependency) and order themselves after it. This target unit is intended to pull in a service that delays further execution until the network is sufficiently set up. What precisely this requires is left to the implementation of the network managing service.

  Note the distinction between this unit and network.target. This unit is an active unit (i.e. pulled in by the consumer rather than the provider of this functionality) and pulls in a service which possibly adds substantial delays to further execution. In contrast, network.target is a passive unit (i.e. pulled in by the provider of the functionality, rather than the consumer) that usually does not delay execution much. Usually, network.target is part of the boot of most systems, while network−online.target is not, except when at least one unit requires it. Also see **Running Services After the Network is up**[1] for more information.

  All mount units for remote network file systems automatically pull in this unit, and order themselves after it. Note that networking daemons that simply *provide* functionality to other hosts (as opposed to *consume* functionality of other hosts) generally do not need to pull this in.

  systemd automatically adds dependencies of type *Wants=* and *After=* for this target unit to all SysV init script service units with an LSB header referring to the "$network" facility.

  Note that this unit is only useful during the original system start−up logic. After the system has completed booting up, it will not track the online state of the system anymore. Due to this it cannot be used as a network connection monitor concept, it is purely a one−time system start−up concept.

paths.target

A special target unit that sets up all path units (see **systemd.path**(5) for details) that shall be active after boot.

It is recommended that path units installed by applications get pulled in via *Wants=* dependencies from this unit. This is best configured via a *WantedBy=paths.target* in the path unit's [Install] section.

poweroff.target
A special target unit for shutting down and powering off the system.

Applications wanting to power off the system should not start this unit directly, but should instead execute **systemctl poweroff** (possibly with the **−−no−block** option) or call **systemd-logind**(8)'s **org.freedesktop.login1.Manager.PowerOff** D−Bus method directly.

runlevel0.target is an alias for this target unit, for compatibility with SysV.

reboot.target
A special target unit for shutting down and rebooting the system.

Applications wanting to reboot the system should not start this unit directly, but should instead execute **systemctl reboot** (possibly with the **−−no−block** option) or call **systemd-logind**(8)'s **org.freedesktop.login1.Manager.Reboot** D−Bus method directly.

runlevel6.target is an alias for this target unit, for compatibility with SysV.

remote−cryptsetup.target
Similar to cryptsetup.target, but for encrypted devices which are accessed over the network. It is used for **crypttab**(8) entries marked with **_netdev**.

remote−veritysetup.target
Similar to veritysetup.target, but for verity integrity protected devices which are accessed over the network. It is used for **veritytab**(8) entries marked with **_netdev**.

remote−fs.target
Similar to local−fs.target, but for remote mount points.

systemd automatically adds dependencies of type *After=* for this target unit to all SysV init script service units with an LSB header referring to the "$remote_fs" facility.

rescue.target
A special target unit that pulls in the base system (including system mounts) and spawns a rescue shell. Isolate to this target in order to administer the system in single−user mode with all file systems mounted but with no services running, except for the most basic. Compare with emergency.target, which is much more reduced and does not provide the file systems or most basic services. Compare with multi−user.target, this target could be seen as single−user.target.

runlevel1.target is an alias for this target unit, for compatibility with SysV.

Use the "systemd.unit=rescue.target" kernel command line option to boot into this mode. A short alias for this kernel command line option is "1", for compatibility with SysV.

runlevel2.target, runlevel3.target, runlevel4.target, runlevel5.target
These are targets that are called whenever the SysV compatibility code asks for runlevel 2, 3, 4, 5, respectively. It is a good idea to make this an alias for (i.e. symlink to) graphical.target (for runlevel 5) or multi−user.target (the others).

shutdown.target
A special target unit that terminates the services on system shutdown.

Services that shall be terminated on system shutdown shall add *Conflicts=* and *Before=* dependencies

to this unit for their service unit, which is implicitly done when *DefaultDependencies=yes* is set (the default).

sigpwr.target
: A special target that is started when systemd receives the SIGPWR process signal, which is normally sent by the kernel or UPS daemons when power fails.

sleep.target
: A special target unit that is pulled in by suspend.target, hibernate.target and hybrid−sleep.target and may be used to hook units into the sleep state logic.

slices.target
: A special target unit that sets up all slice units (see **systemd.slice**(5) for details) that shall always be active after boot. By default the generic system.slice slice unit as well as the root slice unit −.slice are pulled in and ordered before this unit (see below).

  Adding slice units to slices.target is generally not necessary. Instead, when some unit that uses *Slice=* is started, the specified slice will be started automatically. Adding *WantedBy=slices.target* lines to the [Install] section should only be done for units that need to be always active. In that case care needs to be taken to avoid creating a loop through the automatic dependencies on "parent" slices.

sockets.target
: A special target unit that sets up all socket units (see **systemd.socket**(5) for details) that shall be active after boot.

  Services that can be socket−activated shall add *Wants=* dependencies to this unit for their socket unit during installation. This is best configured via a *WantedBy=sockets.target* in the socket unit's [Install] section.

suspend.target
: A special target unit for suspending the system. This pulls in sleep.target.

swap.target
: Similar to local−fs.target, but for swap partitions and swap files.

sysinit.target
: systemd automatically adds dependencies of the types *Requires=* and *After=* for this target unit to all services (except for those with *DefaultDependencies=no*).

  This target pulls in the services required for system initialization. System services pulled in by this target should declare *DefaultDependencies=no* and specify all their dependencies manually, including access to anything more than a read only root filesystem. For details on the dependencies of this target, refer to **bootup**(7).

syslog.socket
: The socket unit syslog implementations should listen on. All userspace log messages will be made available on this socket. For more information about syslog integration, please consult the **Syslog Interface**[2] document.

system−update.target, system−update−pre.target, system−update−cleanup.service
: A special target unit that is used for offline system updates. **systemd-system-update-generator**(8) will redirect the boot process to this target if /system−update exists. For more information see **systemd.offline-updates**(7).

  Updates should happen before the system−update.target is reached, and the services which implement them should cause the machine to reboot. The main units executing the update should order themselves after system−update−pre.target but not pull it in. Services which want to run during system updates only, but before the actual system update is executed should order themselves before this unit and pull it in. As a safety measure, if this does not happen, and /system−update still exists after system−update.target is reached, system−update−cleanup.service will remove this symlink and reboot

the machine.

timers.target
: A special target unit that sets up all timer units (see **systemd.timer**(5) for details) that shall be active after boot.

  It is recommended that timer units installed by applications get pulled in via *Wants=* dependencies from this unit. This is best configured via *WantedBy=timers.target* in the timer unit's [Install] section.

umount.target
: A special target unit that unmounts all mount and automount points on system shutdown.

  Mounts that shall be unmounted on system shutdown shall add Conflicts dependencies to this unit for their mount unit, which is implicitly done when *DefaultDependencies=yes* is set (the default).

Special System Units for Devices

Some target units are automatically pulled in as devices of certain kinds show up in the system. These may be used to automatically activate various services based on the specific type of the available hardware.

bluetooth.target
: This target is started automatically as soon as a Bluetooth controller is plugged in or becomes available at boot.

  This may be used to pull in Bluetooth management daemons dynamically when Bluetooth hardware is found.

printer.target
: This target is started automatically as soon as a printer is plugged in or becomes available at boot.

  This may be used to pull in printer management daemons dynamically when printer hardware is found.

smartcard.target
: This target is started automatically as soon as a smartcard controller is plugged in or becomes available at boot.

  This may be used to pull in smartcard management daemons dynamically when smartcard hardware is found.

sound.target
: This target is started automatically as soon as a sound card is plugged in or becomes available at boot.

  This may be used to pull in audio management daemons dynamically when audio hardware is found.

usb−gadget.target
: This target is started automatically as soon as a USB Device Controller becomes available at boot.

  This may be used to pull in usb gadget dynamically when UDC hardware is found.

Special Passive System Units

A number of special system targets are defined that can be used to properly order boot−up of optional services. These targets are generally not part of the initial boot transaction, unless they are explicitly pulled in by one of the implementing services. Note specifically that these *passive* target units are generally not pulled in by the consumer of a service, but by the provider of the service. This means: a consuming service should order itself after these targets (as appropriate), but not pull it in. A providing service should order itself before these targets (as appropriate) and pull it in (via a *Wants=* type dependency).

Note that these passive units cannot be started manually, i.e. "systemctl start time−sync.target" will fail with an error. They can only be pulled in by dependency. This is enforced since they exist for ordering purposes only and thus are not useful as only unit within a transaction.

blockdev@.target
: This template unit is used to order mount units and other consumers of block devices after services that synthesize these block devices. In particular, this is intended to be used with storage services (such as **systemd-cryptsetup@.service**(5)/ **systemd-veritysetup@.service**(5)) that allocate and manage a virtual block device. Storage services are ordered before an instance of blockdev@.target, and the consumer units after it. The ordering is particularly relevant during shutdown, as it ensures that the mount is deactivated first and the service backing the mount later. The blockdev@.target instance should be pulled in via a **Wants=** dependency of the storage daemon and thus generally not be part of any transaction unless a storage daemon is used. The instance name for instances of this template unit must be a properly escaped block device node path, e.g. blockdev@dev−mapper−foobar.target for the storage device /dev/mapper/foobar.

cryptsetup−pre.target
: This passive target unit may be pulled in by services that want to run before any encrypted block device is set up. All encrypted block devices are set up after this target has been reached. Since the shutdown order is implicitly the reverse start−up order between units, this target is particularly useful to ensure that a service is shut down only after all encrypted block devices are fully stopped.

veritysetup−pre.target
: This passive target unit may be pulled in by services that want to run before any verity integrity protected block device is set up. All verity integrity protected block devices are set up after this target has been reached. Since the shutdown order is implicitly the reverse start−up order between units, this target is particularly useful to ensure that a service is shut down only after all verity integrity protected block devices are fully stopped.

first−boot−complete.target
: This passive target is intended as a synchronization point for units that need to run once during the first boot. Only after all units ordered before this target have finished, will the **machine-id**(5) be committed to disk, marking the first boot as completed. If the boot is aborted at any time before that, the next boot will re−run any units with *ConditionFirstBoot=yes*.

getty−pre.target
: A special passive target unit. Users of this target are expected to pull it in the boot transaction via a dependency (e.g. *Wants=*). Order your unit before this unit if you want to make use of the console just before getty is started.

local−fs−pre.target
: This target unit is automatically ordered before all local mount points marked with **auto** (see above). It can be used to execute certain units before all local mounts.

network.target
: This unit is supposed to indicate when network functionality is available, but it is only very weakly defined what that is supposed to mean. However, the following should apply at minimum:

  - At start−up, any configured synthetic network devices (i.e. not physical ones that require hardware to show up and be probed, but virtual ones like bridge devices and similar which are created programmatically) that do not depend on any underlying hardware should be allocated by the time this target is reached. It is not necessary for these interfaces to also have completed IP level configuration by the time network.target is reached.
  - At shutdown, a unit that is ordered after network.target will be stopped before the network — to whatever level it might be set up by then — is shut down. It is hence useful when writing service files that require network access on shutdown, which should order themselves after this target, but not pull it in. Also see **Running Services After the Network is up**[1] for more information.

  It must emphasized that at start−up there's no guarantee that hardware−based devices have shown up by the time this target is reached, or even acquired complete IP configuration. For that purpose use network−online.target as described above.

network−pre.target
: This passive target unit may be pulled in by services that want to run before any network is set up, for example for the purpose of setting up a firewall. All network management software orders itself after this target, but does not pull it in.

nss−lookup.target
: A target that should be used as synchronization point for all host/network name service lookups. Note that this is independent of UNIX user/group name lookups for which nss−user−lookup.target should be used. All services for which the availability of full host/network name resolution is essential should be ordered after this target, but not pull it in. systemd automatically adds dependencies of type *After=* for this target unit to all SysV init script service units with an LSB header referring to the "$named" facility.

nss−user−lookup.target
: A target that should be used as synchronization point for all regular UNIX user/group name service lookups. Note that this is independent of host/network name lookups for which nss−lookup.target should be used. All services for which the availability of the full user/group database is essential should be ordered after this target, but not pull it in. All services which provide parts of the user/group database should be ordered before this target, and pull it in. Note that this unit is only relevant for regular users and groups — system users and groups are required to be resolvable during earliest boot already, and hence do not need any special ordering against this target.

remote−fs−pre.target
: This target unit is automatically ordered before all mount point units (see above) and cryptsetup/veritysetup devices marked with the **_netdev**. It can be used to run certain units before remote encrypted devices and mounts are established. Note that this unit is generally not part of the initial transaction, unless the unit that wants to be ordered before all remote mounts pulls it in via a *Wants=* type dependency. If the unit wants to be pulled in by the first remote mount showing up, it should use network−online.target (see above).

rpcbind.target
: The portmapper/rpcbind pulls in this target and orders itself before it, to indicate its availability. systemd automatically adds dependencies of type *After=* for this target unit to all SysV init script service units with an LSB header referring to the "$portmap" facility.

time−set.target
: Services responsible for setting the system clock (**CLOCK_REALTIME**) from a local source (such as a maintained timestamp file or imprecise real−time clock) should pull in this target and order themselves before it. Services where approximate, roughly monotonic time is desired should be ordered after this unit, but not pull it in.

  This target does not provide the accuracy guarantees of time−sync.target (see below), however does not depend on remote clock sources to be reachable, i.e. the target is typically not delayed by network problems and similar. Use of this target is recommended for services where approximate clock accuracy and rough monotonicity is desired but activation shall not be delayed for possibly unreliable network communication.

  The service manager automatically adds dependencies of type *After=* for this target unit to all timer units with at least one *OnCalendar=* directive.

  The **systemd-timesyncd.service**(8) service is a simple daemon that pulls in this target and orders itself before it. Besides implementing the SNTP network protocol it maintains a timestamp file on disk whose modification time is regularlary updated. At service start−up the local system clock is set from that modification time, ensuring it increases roughly monotonically.

  Note that ordering a unit after time−set.target only has effect if there's actually a service ordered before it that delays it until the clock is adjusted for rough monotonicity. Otherwise, this target might

get reached before the clock is adjusted to be roughly monotonic. Enable **systemd-timesyncd.service**(8), or an alternative NTP implementation to delay the target.

time−sync.target
: Services indicating completed synchronization of the system clock (**CLOCK_REALTIME**) to a remote source should pull in this target and order themselves before it. Services where accurate time is essential should be ordered after this unit, but not pull it in.

  The service manager automatically adds dependencies of type *After*= for this target unit to all SysV init script service units with an LSB header referring to the "$time" facility, as well to all timer units with at least one *OnCalendar*= directive.

  This target provides stricter clock accuracy guarantees than time−set.target (see above), but likely requires network communication and thus introduces unpredictable delays. Services that require clock accuracy and where network communication delays are acceptable should use this target. Services that require a less accurate clock, and only approximate and roughly monotonic clock behaviour should use time−set.target instead.

  Note that ordering a unit after time−sync.target only has effect if there's actually a service ordered before it that delays it until clock synchronization is reached. Otherwise, this target might get reached before the clock is synchronized to any remote accurate reference clock. When using **systemd-timesyncd.service**(8), enable **systemd-time-wait-sync.service**(8) to delay the target; or use an equivalent service for other NTP implementations.

Table 1. Comparison

time−set.target	**time−sync.target**
"quick" to reach	"slow" to reach
typically uses local clock sources, boot process not affected by availability of external resources	typically uses remote clock sources, inserts dependencies on remote resources into boot process
reliable, because local	unreliable, because typically network involved
typically guarantees an approximate and roughly monotonic clock only	typically guarantees an accurate clock
implemented by systemd−timesyncd.service	implemented by systemd−time−wait−sync.service

Special Slice Units

There are four ".slice" units which form the basis of the hierarchy for assignment of resources for services, users, and virtual machines or containers. See **systemd.slice**(7) for details about slice units.

−.slice
: The root slice is the root of the slice hierarchy. It usually does not contain units directly, but may be used to set defaults for the whole tree.

system.slice
: By default, all system services started by **systemd** are found in this slice.

user.slice
: By default, all user processes and services started on behalf of the user, including the per−user systemd instance are found in this slice. This is pulled in by systemd−logind.service.

machine.slice
: By default, all virtual machines and containers registered with **systemd−machined** are found in this slice. This is pulled in by systemd−machined.service.

UNITS MANAGED BY THE USER SERVICE MANAGER

Special User Units

When systemd runs as a user instance, the following special units are available:

default.target
: This is the main target of the user session, started by default. Various services that compose the normal user session should be pulled into this target. In this regard, default.target is similar to multi−user.target in the system instance, but it is a real unit, not an alias.

In addition, the following units are available which have definitions similar to their system counterparts: exit.target, shutdown.target, sockets.target, timers.target, paths.target, bluetooth.target, printer.target, smartcard.target, sound.target.

Special Passive User Units

graphical−session.target
: This target is active whenever any graphical session is running. It is used to stop user services which only apply to a graphical (X, Wayland, etc.) session when the session is terminated. Such services should have "PartOf=graphical−session.target" in their [Unit] section. A target for a particular session (e. g. gnome−session.target) starts and stops "graphical−session.target" with "BindsTo=graphical−session.target".

  Which services are started by a session target is determined by the "Wants=" and "Requires=" dependencies. For services that can be enabled independently, symlinks in ".wants/" and ".requires/" should be used, see **systemd.unit**(5). Those symlinks should either be shipped in packages, or should be added dynamically after installation, for example using "systemctl add−wants", see **systemctl**(1).

  Example 1. Nautilus as part of a GNOME session "gnome−session.target" pulls in Nautilus as top−level service:

```
[Unit]
Description=User systemd services for GNOME graphical session
Wants=nautilus.service
BindsTo=graphical−session.target
```

  "nautilus.service" gets stopped when the session stops:

```
[Unit]
Description=Render the desktop icons with Nautilus
PartOf=graphical−session.target

[Service]
...
```

graphical−session−pre.target
: This target contains services which set up the environment or global configuration of a graphical session, such as SSH/GPG agents (which need to export an environment variable into all desktop processes) or migration of obsolete d−conf keys after an OS upgrade (which needs to happen before starting any process that might use them). This target must be started before starting a graphical session like gnome−session.target.

xdg−desktop−autostart.target
: The XDG specification defines a way to autostart applications using XDG desktop files. systemd ships **systemd-xdg-autostart-generator**(8) for the XDG desktop files in autostart directories. Desktop Environments can opt−in to use this service by adding a *Wants=* dependency on xdg−desktop−autostart.target.

Special User Slice Units

There are four ".slice" units which form the basis of the user hierarchy for assignment of resources for user applications and services. See **systemd.slice**(7) for details about slice units and the documentation about

Desktop Environments[3] for further information.

−.slice
: The root slice is the root of the user's slice hierarchy. It usually does not contain units directly, but may be used to set defaults for the whole tree.

app.slice
: By default, all user services and applications managed by **systemd** are found in this slice. All interactively launched applications like web browsers and text editors as well as non−critical services should be placed into this slice.

session.slice
: All essential services and applications required for the session should use this slice. These are services that either cannot be restarted easily or where latency issues may affect the interactivity of the system and applications. This includes the display server, screen readers and other services such as DBus or XDG portals. Such services should be configured to be part of this slice by adding *Slice=session.slice* to their unit files.

background.slice
: All services running low−priority background tasks should use this slice. This permits resources to be preferentially assigned to the other slices. Examples include non−interactive tasks like file indexing or backup operations where latency is not important.

SEE ALSO

systemd(1), **systemd.unit**(5), **systemd.service**(5), **systemd.socket**(5), **systemd.target**(5), **systemd.slice**(5), **bootup**(7), **systemd-fstab-generator**(8), **user@.service**(5)

NOTES

1. Running Services After the Network is up
   https://www.freedesktop.org/wiki/Software/systemd/NetworkTarget
2. Syslog Interface
   https://www.freedesktop.org/wiki/Software/systemd/syslog
3. Desktop Environments
   https://systemd.io/DESKTOP_ENVIRONMENTS