systemd.mount

- Description
- Automatic Dependencies
  - Implicit Dependencies
  - Default Dependencies
- fstab
- Options
- See Also

systemd

systemd.mount

5

systemd.mount

Mount unit configuration

`mount.mount`

Description

A unit configuration file whose name ends in `.mount` encodes information about a file system mount point controlled and supervised by systemd.

This man page lists the configuration options specific to this unit type. See `systemd.unit(5)` for the common options of all unit configuration files. The common configuration items are configured in the generic [Unit] and [Install] sections. The mount specific configuration options are configured in the [Mount] section.

Additional options are listed in `systemd.exec(5)`, which define the execution environment the `mount(8)` program is executed in, and in `systemd.kill(5)`, which define the way the processes are terminated, and in `systemd.resource-control(5)`, which configure resource control settings for the

processes of the service.

Note that the options `User=` and `Group=` are not useful for mount units. systemd passes two parameters to `mount(8)`; the values of `What=` and `Where=`. When invoked in this way, `mount(8)` does not read any options from `/etc/fstab`, and must be run as UID 0.

Mount units must be named after the mount point directories they control. Example: the mount point `/home/lennart` must be configured in a unit file `home-lennart.mount`. For details about the escaping logic used to convert a file system path to a unit name, see `systemd.unit(5)`. Note that mount units cannot be templated, nor is possible to add multiple names to a mount unit by creating symlinks to its unit file.

Optionally, a mount unit may be accompanied by an automount unit, to allow on-demand or parallelized mounting. See `systemd.automount(5)`.

Mount points created at runtime (independently of unit files or `/etc/fstab`) will be monitored by systemd and appear like any other mount unit in systemd. See `/proc/self/mountinfo` description in `proc(5)`.

Some file systems have special semantics as API file systems for kernel-to-userspace and userspace-to-userspace interfaces. Some of them may not be changed via mount units, and cannot be disabled. For a longer discussion see API File Systems.

The `systemd-mount(1)` command allows creating `.mount` and `.automount` units dynamically and transiently from the command line.

Automatic Dependencies

Implicit Dependencies

The following dependencies are implicitly added:

- If a mount unit is beneath another mount unit in the file system hierarchy, both a requirement dependency and an ordering dependency between both units are created automatically.
- Block device backed file systems automatically gain `Requires=`, `StopPropagatedFrom=`, and `After=` type dependencies on the device unit encapsulating the block device (see `x-systemd.device-bound=` for details).
- If traditional file system quota is enabled for a mount unit, automatic `Wants=` and `Before=` dependencies on `systemd-quotacheck@.service` and `quotaon@.service` are added.

- Additional implicit dependencies may be added as result of execution and resource control parameters as documented in `systemd.exec(5)` and `systemd.resource-control(5)`.

Default Dependencies

The following dependencies are added unless `DefaultDependencies=no` is set:

- All mount units acquire automatic `Before=` and `Conflicts=` on `umount.target` in order to be stopped during shutdown.
- Mount units referring to local file systems automatically gain an `After=` dependency on `local-fs-pre.target`, and a `Before=` dependency on `local-fs.target` unless one or more mount options among `nofail`, `x-systemd.wanted-by=`, and `x-systemd.required-by=` is set. See below for detailed information.
  Additionally, an `After=` dependency on `swap.target` is added when the file system type is `tmpfs`.
- Network mount units automatically acquire `After=` dependencies on `remote-fs-pre.target`, `network.target`, plus `After=` and `Wants=` dependencies on `network-online.target`, and a `Before=` dependency on `remote-fs.target`, unless one or more mount options among `nofail`, `x-systemd.wanted-by=`, and `x-systemd.required-by=` is set.

Mount units referring to local and network file systems are distinguished by their file system type specification. In some cases this is not sufficient (for example network block device based mounts, such as iSCSI), in which case `_netdev` may be added to the mount option string of the unit, which forces systemd to consider the mount unit a network mount.

`fstab`

Mount units may either be configured via unit files, or via `/etc/fstab` (see `fstab(5)` for details). Mounts listed in `/etc/fstab` will be converted into native units dynamically at boot and when the configuration of the system manager is reloaded. In general, configuring mount points through `/etc/fstab` is the preferred approach to manage mounts for humans. For tooling, writing mount units should be preferred over editing `/etc/fstab`. See `systemd-fstab-generator(8)` for details about the conversion from `/etc/fstab` to mount units.

The NFS mount option `bg` for NFS background mounts as documented in `nfs(5)` is detected by `systemd-fstab-generator` and the options are transformed so that systemd fulfills the job-control implications of that option. Specifically `systemd-fstab-generator` acts as though `x-systemd.mount-timeout=infinity,retry=10000` was prepended to the option list, and `fg,nofail` was appended. Depending on specific requirements, it may be appropriate to provide some of these options explicitly, or to make use of the `x-systemd.automount` option described below instead of using `bg`.

When reading `/etc/fstab` a few special mount options are understood by systemd which influence how dependencies are created for mount points. systemd will create a dependency of type `Wants=` or `Requires=` (see option `nofail` below), from either `local-fs.target` or `remote-fs.target`, depending whether the file system is local or remote.

`x-systemd.requires=`
Configures a `Requires=` and an `After=` dependency between the created mount unit and another systemd unit, such as a device or mount unit. The argument should be a unit name, or an absolute path to a device node or mount point. This option may be specified more than once. This option is particularly useful for mount point declarations that need an additional device to be around (such as an external journal device for journal file systems) or an additional mount to be in place (such as an overlay file system that merges multiple mount points). See `After=` and `Requires=` in `systemd.unit(5)` for details.

Note that this option always applies to the created mount unit only regardless whether `x-systemd.automount` has been specified.

`x-systemd.wants=`
Configures a `Wants=` and an `After=` dependency between the created mount unit and another systemd unit, similar to the `x-systemd.requires=` option.

`x-systemd.before=`; `x-systemd.after=`
In the created mount unit, configures a `Before=` or `After=` dependency on another systemd unit, such as a mount unit. The argument should be a unit name or an absolute path to a mount point. This option may be specified more than once. This option is particularly useful for mount point declarations with `nofail` option that are mounted asynchronously but need to be mounted before or after some unit start, for example, before `local-fs.target` unit. See `Before=` and `After=` in `systemd.unit(5)` for details.

Note that these options always apply to the created mount unit only regardless whether `x-systemd.automount` has been specified.

`x-systemd.wanted-by=`; `x-systemd.required-by=`
In the created mount unit, configures a `WantedBy=` or `RequiredBy=` dependency on another unit. This option may be specified more than once. If this is specified, the default dependencies (see above) other than `umount.target` on the created mount unit, e.g. `local-fs.target`, are not automatically created. Hence it is likely that some ordering dependencies need to be set up manually through `x-systemd.before=` and `x-systemd.after=`. See `WantedBy=` and `RequiredBy=` in `systemd.unit(5)` for details.

`x-systemd.wants-mounts-for=`; `x-systemd.requires-mounts-for=`
Configures a `RequiresMountsFor=` or `WantsMountsFor=` dependency between the created mount unit and other mount units. The argument must be an absolute path. This option may be specified more than once. See `RequiresMountsFor=` or `WantsMountsFor=` in `systemd.unit(5)` for details.

`x-systemd.onfailure=`
In the created mount unit, configures an `OnFailure=` dependency. The argument is a unit name,

see `x-systemd.requires=` above. This option may be specified more than once. See `OnFailure=` in `systemd.unit(5)` for details.

Note that this option always applies to the created mount unit only, regardless of whether `x-systemd.automount` has been specified.

Note that this option is silently ignored on swap entries.

`x-systemd.graceful-option=`
Additional mount option that shall be appended if supported by the kernel. This may be used to configure mount options that are optional and only enabled on kernels that support them. Note that this is supported only for native kernel mount options (i.e. explicitly not for mount options implemented in userspace, such as those processed by `/usr/bin/mount` itself, by FUSE or by mount helpers such as `mount.nfs`). This option may be specified more than once.

`x-systemd.device-bound=`
Takes a boolean argument. If true or no argument, a `BindsTo=` dependency on the backing device is set. If false, the mount unit is not stopped no matter whether the backing device is still present. This is useful when the file system is backed by volume managers. If not set, and the mount comes from unit fragments, i.e. generated from `/etc/fstab` by `systemd-fstab-generator(8)` or loaded from a manually configured mount unit, a combination of `Requires=` and `StopPropagatedFrom=` dependencies is set on the backing device, otherwise only `Requires=` is used.

`x-systemd.automount`
An automount unit will be created for the file system. See `systemd.automount(5)` for details.

`x-systemd.idle-timeout=`
Configures the idle timeout of the automount unit. See `TimeoutIdleSec=` in `systemd.automount(5)` for details.

`x-systemd.device-timeout=`
Configure how long systemd should wait for a device to show up before giving up on an entry from `/etc/fstab`. Specify a time in seconds or explicitly append a unit such as `s`, `min`, `h`, `ms`.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file.

`x-systemd.mount-timeout=`
Configure how long systemd should wait for the mount command to finish before giving up on an entry from `/etc/fstab`. Specify a time in seconds or explicitly append a unit such as `s`, `min`, `h`, `ms`.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file.

See `TimeoutSec=` below for details.

`x-systemd.makefs`
The file system will be initialized on the device. If the device is not "empty", i.e. it contains any signature, the operation will be skipped. It is hence expected that this option remains set even after the device has been initialized.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file.

See `systemd-makefs@.service(8)`.

`wipefs(8)` may be used to remove any signatures from a block device to force `x-systemd.makefs` to reinitialize the device.

`x-systemd.growfs`
The file system will be grown to occupy the full block device. If the file system is already at maximum size, no action will be performed. It is hence expected that this option remains set even after the file system has been grown. Only certain file system types are supported, see `systemd-makefs@.service(8)` for details.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file.

`x-systemd.pcrfs`
Measures file system identity information (mount point, type, label, UUID, partition label, partition UUID) into PCR 15 after the file system has been mounted. This ensures the `systemd-pcrfs@.service(8)` or `systemd-pcrfs-root.service` services are pulled in by the mount unit.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file. It is also implied for the root and `/usr/` partitions discovered by `systemd-gpt-auto-generator(8)`.

`x-systemd.validatefs`
Validates mount constraint metadata of the mounted file system after mounting it. This ensures the `systemd-validatefs@.service(8)` service is pulled in by the mount unit.

Note that this option can only be used in `/etc/fstab`, and will be ignored when part of the `Options=` setting in a unit file. It is also implied for all partitions discovered by `systemd-gpt-auto-generator(8)`.

`x-systemd.rw-only`
If a mount operation fails to mount the file system read-write, it normally tries mounting the file system read-only instead. This option disables that behaviour, and causes the mount to fail immediately instead. This option is translated into the `ReadWriteOnly=` setting in a unit file.

`_netdev`
Normally the file system type is used to determine if a mount is a "network mount", i.e. if it should only be started after the network is available. Using this option overrides this detection and specifies that the mount requires network.

Network mount units are ordered between `remote-fs-pre.target` and `remote-fs.target`, instead of `local-fs-pre.target` and `local-fs.target`. They also pull in `network-online.target` and are ordered after it and `network.target`.

`noauto`; `auto`
With `noauto`, the mount unit will not be added as a dependency for `local-fs.target` or `remote-fs.target`. This means that it will not be mounted automatically during boot, unless it is pulled in by some other unit. The `auto` option has the opposite meaning and is the default.

Note that if `x-systemd.automount` (see above) is used, neither `auto` nor `noauto` have any effect. The matching automount unit will be added as a dependency to the appropriate target.

`nofail`
With `nofail`, this mount will be only wanted, not required, by `local-fs.target` or `remote-fs.target`. Moreover, the mount unit is not ordered before these target units. This means that the boot will continue without waiting for the mount unit and regardless whether the mount point can be mounted successfully.

`x-initrd.mount`
An additional filesystem to be mounted in the initrd. See `initrd-fs.target` description in `systemd.special(7)`. This is both an indicator to the initrd to mount this partition early and an indicator to the host to leave the partition mounted until final shutdown. Or in other words, if this flag is set it is assumed the mount shall be active during the entire regular runtime of the system, i.e. established before the initrd transitions into the host all the way until the host transitions to the final shutdown phase.

If a mount point is configured in both `/etc/fstab` and a unit file that is stored below `/usr/`, the former will take precedence. If the unit file is stored below `/etc/`, it will take precedence. This means: native unit files take precedence over traditional configuration files, but this is superseded by the rule that configuration in `/etc/` will always take precedence over configuration in `/usr/`.

Options

Mount unit files may include [Unit] and [Install] sections, which are described in `systemd.unit(5)`.

Mount unit files must include a [Mount] section, which carries information about the file system mount points it supervises. A number of options that may be used in this section are shared with other unit types. These options are documented in `systemd.exec(5)`, `systemd.kill(5)` and `systemd.resource-control(5)`. The options specific to the [Mount] section of mount units are the following:

`What=`
Takes an absolute path or a fstab-style identifier of a device node, file or other resource to mount. See `mount(8)` for details. If this refers to a device node, a dependency on the respective device unit

is automatically created. (See `systemd.device(5)` for more information.) This option is mandatory. Note that the usual specifier expansion is applied to this setting, literal percent characters should hence be written as `%%`. If this mount is a bind mount and the specified path does not exist yet it is created as directory.

`Where=`
Takes an absolute path of a file or directory for the mount point; in particular, the destination cannot be a symbolic link. If the mount point does not exist at the time of mounting, it is created as either a directory or a file. The former is the usual case; the latter is done only if this mount is a bind mount and the source (`What=`) is not a directory. This string must be reflected in the unit filename. (See above.) This option is mandatory.

`Type=`
Takes a string for the file system type. See `mount(8)` for details. This setting is optional.

If the type is `overlay`, and `upperdir=` or `workdir=` are specified as options and the directories do not exist, they will be created.

`Options=`
Mount options to use when mounting. This takes a comma-separated list of options. This setting is optional. Note that the usual specifier expansion is applied to this setting, literal percent characters should hence be written as `%%`.

`SloppyOptions=`
Takes a boolean argument. If true, parsing of the options specified in `Options=` is relaxed, and unknown mount options are tolerated. This corresponds with `mount(8)`'s `-s` switch. Defaults to off.

`LazyUnmount=`
Takes a boolean argument. If true, detach the filesystem from the filesystem hierarchy at time of the unmount operation, and clean up all references to the filesystem as soon as they are not busy anymore. This corresponds with `umount(8)`'s `-l` switch. Defaults to off.

`ReadWriteOnly=`
Takes a boolean argument. If false, a mount point that shall be mounted read-write but cannot be mounted so is retried to be mounted read-only. If true the operation will fail immediately after the read-write mount attempt did not succeed. This corresponds with `mount(8)`'s `-w` switch. Defaults to off.

`ForceUnmount=`
Takes a boolean argument. If true, force an unmount (in case of an unreachable NFS system). This corresponds with `umount(8)`'s `-f` switch. Defaults to off.

`DirectoryMode=`
Directories of mount points (and any parent directories) are automatically created if needed. This option specifies the file system access mode used when creating these directories. Takes an access mode in octal notation. Defaults to 0755.

`TimeoutSec=`
Configures the time to wait for the mount command to finish. If a command does not exit within the configured time, the mount will be considered failed and be shut down again. All commands still running will be terminated forcibly via `SIGTERM`, and after another delay of this time with `SIGKILL`. (See `KillMode=` in `systemd.kill(5)`.) Takes a unit-less value in seconds, or a time span value such as "5min 20s". Pass 0 to disable the timeout logic. The default value is set from `DefaultTimeoutStartSec=` option in `systemd-system.conf(5)`.

See Also

`systemd(1)`, `systemctl(1)`, `systemd-system.conf(5)`, `systemd.unit(5)`, `systemd.exec(5)`, `systemd.kill(5)`, `systemd.resource-control(5)`, `systemd.service(5)`, `systemd.device(5)`, `proc(5)`, `mount(8)`, `systemd-fstab-generator(8)`, `systemd.directives(7)`, `systemd-mount(1)`

□□□ #2
□ □□□ □□□ 2026-10-31 15:06:26 UTC
□ □□□ □□□ 2026-10-31 15:06:27 UTC