NAME

systemd-boot, sd-boot − A simple UEFI boot manager

DESCRIPTION

systemd−boot (short: **sd−boot**) is a simple UEFI boot manager. It provides a graphical menu to select the entry to boot and an editor for the kernel command line. **systemd−boot** supports systems with UEFI firmware only.

systemd−boot loads boot entry information from the EFI system partition (ESP), usually mounted at /efi/, /boot/, or /boot/efi/ during OS runtime, as well as from the Extended Boot Loader partition if it exists (usually mounted to /boot/). Configuration file fragments, kernels, initrds and other EFI images to boot generally need to reside on the ESP or the Extended Boot Loader partition. Linux kernels must be built with **CONFIG_EFI_STUB** to be able to be directly executed as an EFI image. During boot **systemd−boot** automatically assembles a list of boot entries from the following sources:

- Boot entries defined with **Boot Loader Specification**[1] description files located in /loader/entries/ on the ESP and the Extended Boot Loader Partition. These usually describe Linux kernel images with associated initrd images, but alternatively may also describe arbitrary other EFI executables.
- Unified kernel images following the **Boot Loader Specification**[1], as executable EFI binaries in /EFI/Linux/ on the ESP and the Extended Boot Loader Partition.
- The Microsoft Windows EFI boot manager, if installed
- The Apple macOS boot manager, if installed
- The EFI Shell binary, if installed
- A reboot into the UEFI firmware setup option, if supported by the firmware

systemd−boot supports the following features:

- Basic boot manager configuration changes (such as timeout configuration, default boot entry selection, ...) may be made directly from the boot loader UI at boot−time, as well as during system runtime with EFI variables.
- The boot manager integrates with the **systemctl** command to implement features such as **systemctl reboot −−boot−loader−entry=...** (for rebooting into a specific boot menu entry, i.e. "reboot into Windows") and **systemctl reboot −−boot−loader−menu=...** (for rebooting into the boot loader menu), by implementing the **Boot Loader Interface**[2]. See **systemctl**(1) for details.
- An EFI variable set by the boot loader informs the OS about the EFI System Partition used during boot. This is then used to automatically mount the correct EFI System Partition to /efi/ or /boot/ during OS runtime. See **systemd-gpt-auto-generator**(8) for details.
- The boot manager provides information about the boot time spent in UEFI firmware using the **Boot Loader Interface**[2]. This information can be displayed using **systemd-analyze**(1).
- The boot manager implements boot counting and automatic fallback to older, working boot entries on failure. See **Automatic Boot Assessment**[3].
- The boot manager optionally reads a random seed from the ESP partition, combines it with a 'system token' stored in a persistent EFI variable and derives a random seed to use by the OS as entropy pool initialization, providing a full entropy pool during early boot.

bootctl(1) may be used from a running system to locate the ESP and the Extended Boot Loader Partition, list available entries, and install **systemd−boot** itself.

kernel-install(8) may be used to copy kernel images onto the ESP or the Extended Boot Loader Partition and to generate description files compliant with the Boot Loader Specification.

KEY BINDINGS

The following keys may be used in the boot menu:

↑ (Up), ↓ (Down), j, k, PageUp, PageDown, Home, End
Navigate up/down in the entry list

↵ (Enter), → (Right)
: Boot selected entry

d
: Make selected entry the default

e
: Edit the kernel command line for selected entry

+, t
: Increase the timeout before default entry is booted

−, T
: Decrease the timeout

v
: Show systemd−boot, UEFI, and firmware versions

P
: Print status

Q
: Quit

h, ?, F1
: Show a help screen

Ctrl+l
: Reprint the screen

The following keys may be pressed during bootup or in the boot menu to directly boot a specific entry:

l
: Linux

w
: Windows

a
: macOS

s
: EFI shell

1, 2, 3, 4, 5, 6, 7, 8, 9
: Boot entry number 1 ... 9

The boot menu is shown when a non−zero menu timeout has been configured. If the menu timeout has been set to zero, it is sufficient to press any key — before the boot loader initializes — to bring up the boot menu, except for the keys listed immediately above as they directly boot into the selected boot menu item. Note that depending on the firmware implementation the time window where key presses are accepted before the boot loader initializes might be short. If the window is missed, reboot and try again, possibly pressing a suitable key (e.g. the space bar) continuously; on most systems it should be possible to hit the time window after a few attempts. To avoid this problem, consider setting a non−zero timeout, thus showing the boot menu unconditionally. Some desktop environments might offer an option to directly boot into the boot menu, to avoid the problem altogether. Alternatively, use the command line **systemctl reboot −−boot−loader−menu=0** from the shell.

In the editor, most keys simply insert themselves, but the following keys may be used to perform additional actions:

← (Left), → (Right), Home, End
: Navigate left/right

Esc
: Abort the edit and quit the editor

Ctrl+k
: Clear the command line

Ctrl+w, Alt+Backspace
: Delete word backwards

Alt+d
: Delete word forwards

↵ (Enter)
: Boot entry with the edited command line

Note that unless configured otherwise in the UEFI firmware, systemd−boot will use the US keyboard layout, so key labels might not match for keys like +/−.

FILES

The files **systemd−boot** processes generally reside on the UEFI ESP which is usually mounted to /efi/, /boot/ or /boot/efi/ during OS runtime. It also processes files on the Extended Boot Loader partition which is typically mounted to /boot/, if it exists. **systemd−boot** reads runtime configuration such as the boot timeout and default entry from /loader/loader.conf on the ESP (in combination with data read from EFI variables). See **loader.conf**(5). Boot entry description files following the **Boot Loader Specification**[1] are read from /loader/entries/ on the ESP and the Extended Boot Loader partition. Unified kernel boot entries following the **Boot Loader Specification**[1] are read from /EFI/Linux/ on the ESP and the Extended Boot Loader partition. Optionally, a random seed for early boot entropy pool provisioning is stored in /loader/random−seed in the ESP.

EFI VARIABLES

The following EFI variables are defined, set and read by **systemd−boot**, under the vendor UUID "4a67b082−0a4c−41cf−b6c7−440b29bb8c4f", for communication between the OS and the boot loader:

LoaderBootCountPath
: If boot counting is enabled, contains the path to the file in whose name the boot counters are encoded. Set by the boot loader. **systemd-bless-boot.service**(8) uses this information to mark a boot as successful as determined by the successful activation of the boot−complete.target target unit.

LoaderConfigTimeout, *LoaderConfigTimeoutOneShot*
: The menu timeout in seconds. Read by the boot loader. *LoaderConfigTimeout* is maintained persistently, while *LoaderConfigTimeoutOneShot* is a one−time override which is read once (in which case it takes precedence over *LoaderConfigTimeout*) and then removed. *LoaderConfigTimeout* may be manipulated with the t/T keys, see above.

LoaderDevicePartUUID
: Contains the partition UUID of the EFI System Partition the boot loader was run from. Set by the boot loader. **systemd-gpt-auto-generator**(8) uses this information to automatically find the disk booted from, in order to discover various other partitions on the same disk automatically.

LoaderEntries
: A list of the identifiers of all discovered boot loader entries. Set by the boot loader.

LoaderEntryDefault, *LoaderEntryOneShot*
: The identifier of the default boot loader entry. Set primarily by the OS and read by the boot loader. *LoaderEntryOneShot* sets the default entry for the next boot only, while *LoaderEntryDefault* sets it persistently for all future boots. **bootctl**(1)'s **set−default** and **set−oneshot** commands make use of these variables. The boot loader modifies *LoaderEntryDefault* on request, when the d key is used, see above.

LoaderEntrySelected
: The identifier of the boot loader entry currently being booted. Set by the boot loader.

LoaderFeatures
: A set of flags indicating the features the boot loader supports. Set by the boot loader. Use **bootctl**(1) to view this data.

LoaderFirmwareInfo, *LoaderFirmwareType*
Brief firmware information. Set by the boot loader. Use **bootctl**(1) to view this data.

LoaderImageIdentifier
The path of executable of the boot loader used for the current boot, relative to the EFI System Partition's root directory. Set by the boot loader. Use **bootctl**(1) to view this data.

LoaderInfo
Brief information about the boot loader. Set by the boot loader. Use **bootctl**(1) to view this data.

LoaderTimeExecUSec, *LoaderTimeInitUSec*, *LoaderTimeMenuUsec*
Information about the time spent in various parts of the boot loader. Set by the boot loader. Use **systemd-analyze**(1) to view this data.

LoaderRandomSeed
A binary random seed **systemd−boot** may optionally pass to the OS. This is a volatile EFI variable that is hashed at boot from the combination of a random seed stored in the ESP (in /loader/random−seed) and a "system token" persistently stored in the EFI variable *LoaderSystemToken* (see below). During early OS boot the system manager reads this variable and passes it to the OS kernel's random pool, crediting the full entropy it contains. This is an efficient way to ensure the system starts up with a fully initialized kernel random pool — as early as the initial RAM disk phase. **systemd−boot** reads the random seed from the ESP, combines it with the "system token", and both derives a new random seed to update in−place the seed stored in the ESP, and the random seed to pass to the OS from it via SHA256 hashing in counter mode. This ensures that different physical systems that boot the same "golden" OS image — i.e. containing the same random seed file in the ESP — will still pass a different random seed to the OS. It is made sure the random seed stored in the ESP is fully overwritten before the OS is booted, to ensure different random seed data is used between subsequent boots.

See **Random Seeds**[4] for further information.

LoaderSystemToken
A binary random data field, that is used for generating the random seed to pass to the OS (see above). Note that this random data is generally only generated once, during OS installation, and is then never updated again.

Many of these variables are defined by the **Boot Loader Interface**[2].

BOOT COUNTING

systemd−boot implements a simple boot counting mechanism on top of the **Boot Loader Specification**[1], for automatic and unattended fallback to older kernel versions/boot loader entries when a specific entry continuously fails. Any boot loader entry file and unified kernel image file that contains a "+" followed by one or two numbers (if two they need to be separated by a "−"), before the .conf or .efi suffix is subject to boot counting: the first of the two numbers ('tries left') is decreased by one on every boot attempt, the second of the two numbers ('tries done') is increased by one (if 'tries done' is absent it is considered equivalent to 0). Depending on the current value of these two counters the boot entry is considered to be in one of three states:

1. If the 'tries left' counter of an entry is greater than zero the entry is considered to be in 'indeterminate' state. This means the entry has not completed booting successfully yet, but also hasn't been determined not to work.
2. If the 'tries left' counter of an entry is zero it is considered to be in 'bad' state. This means no further attempts to boot this item will be made (that is, unless all other boot entries are also in 'bad' state), as all attempts to boot this entry have not completed successfully.
3. If the 'tries left' and 'tries done' counters of an entry are absent it is considered to be in 'good' state. This means further boot counting for the entry is turned off, as it successfully booted at least once. The **systemd-bless-boot.service**(8) service moves the currently booted entry from 'indeterminate' into 'good' state when a boot attempt completed successfully.

Generally, when new entries are added to the boot loader, they first start out in 'indeterminate' state, i.e. with a 'tries left' counter greater than zero. The boot entry remains in this state until either it managed to complete a full boot successfully at least once (in which case it will be in 'good' state) — or the 'tries left' counter reaches zero (in which case it will be in 'bad' state).

Example: let's say a boot loader entry file foo.conf is set up for 3 boot tries. The installer will hence create it under the name foo+3.conf. On first boot, the boot loader will rename it to foo+2−1.conf. If that boot does not complete successfully, the boot loader will rename it to foo+1−2.conf on the following boot. If that fails too, it will finally be renamed foo+0−3.conf by the boot loader on next boot, after which it will be considered 'bad'. If the boot succeeds however the entry file will be renamed to foo.conf by the OS, so that it is considered 'good' from then on.

The boot menu takes the 'tries left' counter into account when sorting the menu entries: entries in 'bad' state are ordered at the beginning of the list, and entries in 'good' or 'indeterminate' at the end. The user can freely choose to boot any entry of the menu, including those already marked 'bad'. If the menu entry to boot is automatically determined, this means that 'good' or 'indeterminate' entries are generally preferred (as the bottom item of the menu is the one booted by default), and 'bad' entries will only be considered if there are no 'good' or 'indeterminate' entries left.

The **kernel-install**(8) kernel install framework optionally sets the initial 'tries left' counter to the value specified in /etc/kernel/tries when a boot loader entry is first created.

SEE ALSO

bootctl(1), **loader.conf**(5), **systemd-bless-boot.service**(8), **systemd-boot-system-token.service**(8), **kernel-install**(8), **Boot Loader Specification**[1], **Boot Loader Interface**[2]

NOTES

1. Boot Loader Specification
   https://systemd.io/BOOT_LOADER_SPECIFICATION
2. Boot Loader Interface
   https://systemd.io/BOOT_LOADER_INTERFACE
3. Automatic Boot Assessment
   https://systemd.io/AUTOMATIC_BOOT_ASSESSMENT
4. Random Seeds
   https://systemd.io/RANDOM_SEEDS