bootctl

- Description
- Generic EFI Firmware/Boot Loader Commands
- Boot Loader Specification Commands
- Boot Loader Interface Commands
- systemd-boot Commands
- Kernel Image Commands
- Options
- Signed .efi files
- Exit status
- Environment
- Examples
- Files
- See Also

systemd

bootctl

1

bootctl

Control EFI firmware boot settings and manage boot loader

bootctl

OPTIONS

COMMAND

Description

`bootctl` can check the EFI firmware and boot loader status, list and manage available boot loaders and boot loader entries, and install, update, or remove the `systemd-boot(7)` boot loader on the current system.

Generic EFI Firmware/Boot Loader Commands

These commands are available on any EFI system, regardless of the boot loader used.

`status`
Shows brief information about the system firmware, the boot loader that was used to boot the system, the boot loaders currently available in the ESP, the boot loaders listed in the firmware's list of boot loaders and the current default boot loader entry. If no command is specified, this is the implied default.

See the example below for details of the output.

`reboot-to-firmware` [<BOOL>]
Query or set the "Reboot-Into-Firmware-Setup" flag of the EFI firmware. Takes a boolean argument which controls whether to show the firmware setup on next system reboot. If the argument is omitted shows the current status of the flag, or whether the flag is supported. This controls the same flag as `systemctl reboot --firmware-setup`, but is more low-level and allows setting the flag independently from actually requesting a reboot.

Hint: use `systemctl reboot --firmware-setup` to reboot into firmware setup once. See `systemctl(1)` for details.

Boot Loader Specification Commands

These commands are available for all boot loaders that implement the UAPI.1 Boot Loader Specification, such as `systemd-boot`.

`list`
Shows all available boot loader entries implementing the UAPI.1 Boot Loader Specification, as well

as any other entries discovered or automatically generated by a boot loader implementing the Boot Loader Interface. JSON output may be requested with `--json=`.

See the example below for details of the output.

`unlink` [<ID>]
Removes a boot loader entry including the files it refers to. Takes an optional boot loader entry ID string or a glob pattern as argument. Referenced files such as the kernel, initrds, system extensions (sysexts), configuration extensions (confexts) or credential files are only removed if no other entry refers to them.

If no <ID> argument is specified, the `--oldest` option must be specified, in which case the boot loader entry with the lowest version is removed (for robustness reasons the currently booted menu entry is never removed, nor is the last existing boot loader entry).

`link` <KERNEL>
Creates one or more Type #1 boot loader entries pointing to the specified UKI. Takes the path to a Unified Kernel Image (UKI) as argument. The UKI is copied into the ESP (or XBOOTLDR partition if present) below the configured entry token directory, and one or more Boot Loader Specification Type #1 entries are generated referring to it (one per UKI profile, if multiple profiles are embedded).

The title, version, commit number and initial try counter of the generated entries may be overridden with `--entry-title=`, `--entry-version=`, `--entry-commit=` and `--tries-left=`. Additional sidecar resources (system extension images, configuration extension images, credential files) to pass to the UKI at boot may be specified with `--extra=`.

If the ESP/XBOOTLDR do not have enough free space for the new boot loader entry and its referenced resources the oldest existing boot loader entry matching the selected entry token is removed (along with any resources referenced by it that are no longer referenced by any other entry). This step is repeated until the new boot loader entry fits. For robustness reasons the currently booted boot loader entry is never removed, nor is the last existing boot loader entry.

By default, the operation refuses to proceed if the resulting ESP/XBOOTLDR free space would drop below a safety threshold after automatic removal of older entries completes; use `--keep-free=` to adjust.

`link-auto`
Like `link`, but instead of taking the kernel and sidecar resources as arguments, discovers them automatically. The following directories are searched, in decreasing order of priority: `/etc/systemd/uki/`, `/run/systemd/uki/`, `/var/lib/systemd/uki/` (where `systemd-sysupdate(8)` stages downloaded resources), `/usr/local/lib/systemd/uki/` and `/usr/lib/systemd/uki/`. In each directory the UKI is taken from `kernel.efi`, or — if absent — the newest version in the versioned directory `kernel.efi.v/` (see `systemd.v(7)`); the UKI from the highest-priority directory that has one is used. Sidecar resources are picked up from the `extras.d/` subdirectory of each, matching the suffixes `.sysext.raw`, `.confext.raw` and `.cred`, each either as a plain file or as a versioned `.v/` directory;

resources are combined across all directories, with the same resource in a higher-priority directory taking precedence. Any files specified with `--extra=` are linked in addition to the discovered ones.

This is used by `systemd-sysupdate(8)` when it completes an update, so a freshly downloaded kernel and its sidecar resources are linked into place without specifying any paths.

`cleanup`
Removes files from the ESP and XBOOTLDR partitions that belong to the entry token but are not referenced in any boot loader entries.

Boot Loader Interface Commands

These commands are available for all boot loaders that implement the UAPI.1 Boot Loader Specification and the Boot Loader Interface, such as `systemd-boot`.

`set-preferred` <ID>; `set-default` <ID>; `set-oneshot` <ID>; `set-sysfail` <ID>
Sets the default boot loader entry. Takes a single boot loader entry ID string or a glob pattern as argument. The `set-oneshot` command will set the default entry only for the next boot, `set-default` will set it persistently for all future boots, `set-preferred` is like `set-default`, but is aware of boot assessment and will skip boot entries that have their tries-left counter set to zero. The `set-sysfail` command will set the boot loader entry to be used in case of a system failure. System failure (SysFail) boot entries can optionally modify the automatic selection order in the event of a failure, such as a boot firmware update failure with the failure status recorded in the EFI system table.

`bootctl list` can be used to list available boot loader entries and their IDs.

In addition, the boot loader entry ID may be specified as one of: `@default`, `@oneshot` or `@current`, which correspond to the current default boot loader entry for all future boots, the current default boot loader entry for the next boot, and the currently booted boot loader entry. These special IDs are resolved to the current values of the EFI variables `LoaderEntryDefault`, `LoaderEntrySysFail`, `LoaderEntryOneShot` and `LoaderEntrySelected`, see UAPI.1 Boot Loader Specification for details. These special IDs are primarily useful as a quick way to persistently make the currently booted boot loader entry the default choice, or to upgrade the default boot loader entry for the next boot to the default boot loader entry for all future boots, but may be used for other operations too.

If set to `@saved` the chosen entry will be saved as an EFI variable on every boot and automatically selected the next time the boot loader starts.

When an empty string ("") is specified as the ID, then the corresponding EFI variable will be unset.

Hint: use `systemctl reboot --boot-loader-entry=ID` to reboot into a specific boot entry and `systemctl reboot --boot-loader-menu=timeout` to reboot into the boot loader menu once. See `systemctl(1)` for details.

`set-timeout` <TIMEOUT>; `set-timeout-oneshot` <TIMEOUT>
Sets the boot loader menu timeout in seconds. The `set-timeout-oneshot` command will set the timeout only for the next boot. See `systemd.time(7)` for details about the syntax of time spans.

If this is set to `menu-disabled` or `menu-hidden` or `0`, no menu is shown and the default entry will be booted immediately, while setting this to `menu-force` disables the timeout while always showing the menu. When an empty string ("") is specified the bootloader will revert to its default menu timeout.

`systemd-boot` Commands

These commands manage the `systemd-boot` EFI boot loader, and do not work in conjunction with other boot loaders.

`install`
Installs `systemd-boot` into the EFI system partition. A copy of `systemd-boot` will be stored as the EFI default/fallback loader at `ESP/EFI/BOOT/BOOT*.EFI`. The boot loader is then added to the top of the firmware's boot loader list.

`update`
Updates all installed versions of `systemd-boot(7)`, if the available version is newer than the version installed in the EFI system partition. This also includes the EFI default/fallback loader at `ESP/EFI/BOOT/BOOT*.EFI`. The boot loader is then added to end of the firmware's boot loader list if missing.

`remove`
Removes all installed versions of `systemd-boot` from the EFI system partition and the firmware's boot loader list.

`is-installed`
Checks whether `systemd-boot` is installed in the ESP. Note that a single ESP might host multiple boot loaders; this hence checks whether `systemd-boot` is one (of possibly many) installed boot loaders — and neither whether it is the default nor whether it is registered in any EFI variables.

`random-seed`
Generates a random seed and stores it in the EFI System Partition (ESP), for use by the `systemd-boot` boot loader. If a random seed already exists in the ESP it is refreshed. Also generates a random 'system token' and stores it persistently as an EFI variable, if one has not been set before. If the boot loader finds the random seed in the ESP and the system token in the EFI variable it will derive a random seed to pass to the OS and a new seed to store in the ESP from the combination of both. The random seed passed to the OS is credited to the kernel's entropy pool by the system

manager during early boot, and permits userspace to boot up with an entropy pool fully initialized very early on. Also see `systemd-boot-random-seed.service(8)`.

See Random Seeds for further information.

Kernel Image Commands

`kernel-identify` <KERNEL-IMAGE>
Takes a kernel image as argument. Checks what kind of kernel the image is. Returns one of `uki`, `addon`, `pe`, and `unknown`.

`kernel-inspect` <KERNEL-IMAGE>
Takes a kernel image as argument. Prints details about the image.

Options

The following options are understood:

`--root=root`
Takes a directory path as an argument. All paths will be prefixed with the given alternate <root> path, including config search paths.

`--image=image`
Takes a path to a disk image file or block device node. If specified, all operations are applied to file system in the indicated disk image. This option is similar to `--root=`, but operates on file systems stored in disk images or block devices. The disk image should either contain just a file system or a set of file systems within a GPT partition table, following the UAPI.2 Discoverable Partitions Specification. For further information on supported disk images, see `systemd-nspawn(1)`'s switch of the same name.

`--install-source=`
When installing binaries with `--root=` or `--image=`, selects where to source them from. Takes one of `auto` (the default), `image` or `host`. With `auto` binaries will be picked from the specified directory or image, and if not found they will be picked from the host. With `image` or `host` no fallback search will be performed if the binaries are not found in the selected source.

`-p`; `--print-esp-path`
This option modifies the behaviour of `status`. Only prints the path to the EFI System Partition (ESP) to standard output and exits.

`-x`; `--print-boot-path`
This option modifies the behaviour of `status`. Only prints the path to the Extended Boot Loader partition if it exists, and the path to the ESP otherwise to standard output and exit. This command is useful to determine where to place boot loader entries, as they are preferably placed in the Extended Boot Loader partition if it exists and in the ESP otherwise.

Boot Loader Specification Type #1 entries should generally be placed in the directory `$(bootctl -x)/loader/entries/`. Existence of that directory may also be used as indication that boot loader entry support is available on the system. Similarly, Boot Loader Specification Type #2 entries should be placed in the directory `$(bootctl -x)/EFI/Linux/`.

Note that this option (similarly to the `--print-esp-path` option mentioned above), is available independently from the boot loader used, i.e. also without `systemd-boot` being installed.

`--print-loader-path`
This option modifies the behaviour of `status`: it shows the absolute path to the boot loader EFI binary used for the current boot if this information is available. Note that no attempt is made to verify whether the binary still exists.

`--print-stub-path`
This option modifies the behaviour of `status`: it shows the absolute path to the UKI/stub EFI binary used for the current boot if this information is available. Note that no attempt is made to verify whether the binary still exists.

`-R`; `--print-root-device`
Print the path to the block device node backing the root file system of the local OS. This prints a path such as `/dev/nvme0n1p5`. If the root file system is backed by dm-crypt/LUKS or dm-verity the underlying block device is returned. If the root file system is backed by multiple block devices (as supported by btrfs) the operation will fail. If the switch is specified twice (i.e. `-RR`) and the discovered block device is a partition device the "whole" block device it belongs to is determined and printed (e.g. `/dev/nvme0n1`). If the root file system is `tmpfs` (or a similar in-memory file system), the block device backing `/usr/` is returned if applicable. If the root file system is a network file system (e.g. NFS, CIFS) the operation will fail.

`--print-efi-architecture`
Print the EFI architecture string of the local firmware. This is useful to generically format filenames such as `bootx64.efi` that include the local firmware architecture in the name.

`--variables=yes|no|auto`
Controls whether to touch the firmware's boot loader list stored in EFI variables, and other EFI variables. If not specified or set to `auto`, EFI variables will not be modified when execution in a container runtime is detected. `yes` may be used to explicit override the check.

`--random-seed=yes|no`
By default, the `install` command initializes a random seed file in the ESP. When creating an image it may be desirable to disable that in order to avoid having the same seed in all instances.

`--graceful`
Ignore failure when the EFI System Partition cannot be found, when EFI variables cannot be written, or a different or newer boot loader is already installed. Currently only applies to `is-installed`, `update`, `random-seed` and `install` verbs. Since version v258, it is implicitly enabled when running inside a chroot.

`-q`; `--quiet`
Suppress printing of the results of various commands and also the hints about ESP being unavailable.

`--make-entry-directory=yes|no`
Controls creation and deletion of the UAPI.1 Boot Loader Specification Type #1 entry directory on the file system containing resources such as kernel and initrd images during `install` and `remove`, respectively. The directory is named after the entry token, as specified with `--entry-token=` parameter described below, and is placed immediately below the `$BOOT` root directory (i.e. beneath the file system returned by the `--print-boot-path` option, see above). Defaults to `no`.

`--entry-token=`
Controls how to name and identify boot loader entries for this OS installation. Accepted during `install`, and takes one of `auto`, `machine-id`, `os-id`, `os-image-id` or an arbitrary string prefixed by `literal:` as argument.

If set to `machine-id` the entries are named after the machine ID of the running system (e.g. `b0e793a9baf14b5fa13ecbe84ff637ac`). See `machine-id(5)` for details about the machine ID concept and file.

If set to `os-id` the entries are named after the OS ID of the running system, i.e. the `ID=` field of `os-release(5)` (e.g. `fedora`). Similarly, if set to `os-image-id` the entries are named after the OS image ID of the running system, i.e. the `IMAGE_ID=` field of `os-release` (e.g. `vendorx-cashier-system`).

If set to `auto` (the default), the `/etc/kernel/entry-token` file will be read if it exists, and the stored value used. Otherwise, if the local machine ID is initialized it is used. Otherwise, `IMAGE_ID=` from `os-release` will be used, if set. Otherwise, `ID=` from `os-release` will be used, if set.

Unless set to `machine-id`, or when `--make-entry-directory=yes` is used the selected token string is written to a file `/etc/kernel/entry-token`, to ensure it will be used for future entries. This file is also read by `kernel-install(8)`, in order to identify under which name to generate boot loader entries for newly installed kernels, or to determine the entry names for removing old ones.

Using the machine ID for naming the entries is generally preferable, however there are cases where using the other identifiers is a good option. Specifically: if the identification data that the machine ID entails shall not be stored on the (unencrypted) `$BOOT` partition, or if the ID shall be generated on first boot and is not known when the entries are prepared. Note that using the machine ID has the benefit that multiple parallel installations of the same OS can coexist on the same medium, and they can update their boot loader entries independently. When using another

identifier (such as the OS ID or the OS image ID), parallel installations of the same OS would try to use the same entry name. To support parallel installations, the installer must use a different entry token when adding a second installation.

`--all-architectures`
Install binaries for all supported EFI architectures (this implies `--variables=no`).

`--efi-boot-option-description=`
Description of the entry added to the firmware's boot option list. Defaults to `Linux Boot Manager`.

Using the default entry name `Linux Boot Manager` is generally preferable as only one bootloader installed to a single ESP partition should be used to boot any number of OS installations found on the various disks installed in the system. Specifically distributions should not use this flag to install a branded entry in the boot option list. However, in situations with multiple disks, each with their own ESP partition, it can be beneficial to make it easier to identify the bootloader being used in the firmware's boot option menu.

`--efi-boot-option-description-with-device=`
Takes a boolean, defaults to false. Controls whether to append disk model information to the firmware boot option item description (as configured with `--efi-boot-option-description=` above). This is useful when installing multiple operating systems on separate disks on the same system, as it ensures the firmware boot options are discernible and give a hint which disk is booted into. Note that this uses hardware model information, and hence might not be too useful in case multiple disks of an identical model are used.

`-n`; `--dry-run`
Dry run for `unlink` and `cleanup`.

In dry run mode, the unlink and cleanup operations only print the files that would get deleted without actually deleting them. The short option `-n` was added in version 262.

`--oldest`
When used with `unlink`, selects the oldest installed boot loader entry matching the boot entry token for removal (rather than passing an explicit entry ID). This is useful for pruning older installed boot loader entries. Note that the currently booted entry is never removed, nor is the last remaining one.

`--keep-free=BYTES`
When used with `link`, controls the minimum amount of free space (in bytes) that must remain on the target partition (ESP or XBOOTLDR) after the new entry has been materialized. The operation fails if installing the entry would drop the free space below this threshold. Accepts the usual size suffixes (K, M, G, …). If empty, the built-in default is restored. If set to zero no minimum amount of free space is kept.

`--entry-title=TITLE`
When used with `link`, specifies the title of the generated boot loader entry (the `title` field of the Type #1 entry). If not specified, a title is derived from the UKI's embedded metadata.

`--entry-version=VERSION`
When used with `link`, specifies the version string of the generated boot loader entry (the `version` field of the Type #1 entry). If not specified, the version is derived from the UKI's embedded metadata. Used by the boot loader to sort and select entries.

`--entry-commit=NR`
When used with `link`, specifies the commit number for the generated boot loader entry.

`--extra=PATH`; `-X PATH`
When used with `link`, registers an additional sidecar resource file that shall be passed to the UKI at boot. This may be a system extension image (`*.sysext.raw`), configuration extension image (`*.confext.raw`), or credential file (`*.cred`). The file is copied into the ESP/XBOOTLDR alongside the UKI and the boot loader will load and pass it to the kernel via initrd. This option may be used multiple times to register more than one extra resource. If passed an empty argument, all previously specified extras are cleared.

`--tries-left=NR`
When used with `link`, initializes the boot counting `tries-left` counter for the generated entry. If set, the resulting boot entry file is named according to the boot counting scheme described in the Automatic Boot Assessment documentation, so that the boot loader decreases the counter on each attempted boot and eventually marks the entry as bad. If not specified, boot counting is not enabled for the generated entry.

`--secure-boot-auto-enroll=yes|no`; `--private-key=PATH/URI`; `--private-key-source=TYPE[:NAME]`; `--certificate=PATH`; `--certificate-source=TYPE[:NAME]`
Configure the ESP for secure boot auto-enrollment when invoking the `install` command. Takes a boolean argument. Disabled by default. Enabling this option will make `bootctl` populate the ESP with signed `PK`, `KEK` and `db` signature databases, each containing the given certificate in `DER` format as their only entry. These secure boot signature databases will be picked up and enrolled by `systemd-boot` if secure boot is in setup mode and secure boot auto-enrollment is enabled.

When specifying this option, a certificate and private key have to be provided as well using the `--certificate=` and `--private-key=` options. The `--certificate=` option takes a path to a PEM encoded X.509 certificate or a URI that's passed to the OpenSSL provider configured with `--certificate-source` which takes one of `file` or `provider`, with the latter being followed by a specific provider identifier, separated with a colon, e.g. `provider:pkcs11`. The `--private-key=` option can take a path or a URI that will be passed to the OpenSSL engine or provider, as specified by `--private-key-source=` as a `type:name` tuple, such as `engine:pkcs11`. The specified OpenSSL signing engine or provider will be used to sign the EFI signature lists.

Signed .efi files

`bootctl` `install` and `update` will look for a `systemd-boot` file ending with the `.efi.signed` suffix first, and copy that instead of the normal `.efi` file. This allows distributions or end-users to provide signed images for UEFI SecureBoot.

Exit status

On success, 0 is returned, a non-zero failure code otherwise. `bootctl --print-root-device` returns exit status 80 in case the root file system is not backed by single block device, and other non-zero exit statuses on other errors.

Environment

If `$SYSTEMD_RELAX_ESP_CHECKS=1` is set the validation checks for the ESP are relaxed, and the path specified with `--esp-path=` may refer to any kind of file system on any kind of partition.

Similarly, `$SYSTEMD_RELAX_XBOOTLDR_CHECKS=1` turns off some validation checks for the Extended Boot Loader partition.

Examples

Output from `status` and `list`

```
$ bootctl status
System:
     Firmware: UEFI 2.40 (firmware-version)  ← firmware vendor and version
  Secure Boot: disabled (setup)              ← Secure Boot status
 TPM2 Support: yes
 Boot into FW: supported                     ← does the firmware support booting into itself

Current Boot Loader:                         ← details about systemd-boot or another boot
loader
      Product: systemd-boot version            implementing the Boot Loader Interface
     Features: ✓ Boot counting
               ✓ Menu timeout control
               ✓ One-shot menu timeout control
```

```
               ✓ Default entry control
               ✓ One-shot entry control
               ✓ Support for XBOOTLDR partition
               ✓ Support for passing random seed to OS
               ✓ Load drop-in drivers
               ✓ Boot loader sets ESP information
               ✓ Menu can be disabled
          ESP: /dev/disk/by-partuuid/01234567-89ab-cdef-dead-beef00000000
         File: └─/EFI/systemd/systemd-bootx64.efi

Random Seed:                                ← random seed used for entropy in early boot
 Passed to OS: yes
 System Token: set
       Exists: yes

Available Boot Loaders on ESP:
          ESP: /boot/efi (/dev/disk/by-partuuid/01234567-89ab-cdef-dead-beef00000000)
         File: └─/EFI/systemd/systemd-bootx64.efi (systemd-boot 251
         File: └─/EFI/BOOT/BOOTX64.EFI (systemd-boot 251

Boot Loaders Listed in EFI Variables:
        Title: Linux Boot Manager
           ID: 0x0001
       Status: active, boot-order
    Partition: /dev/disk/by-partuuid/…
         File: └─/EFI/systemd/systemd-bootx64.efi

        Title: Fedora
           ID: 0x0000
       Status: active, boot-order
    Partition: /dev/disk/by-partuuid/…
         File: └─/EFI/fedora/shimx64.efi

        Title: Linux-Firmware-Updater
           ID: 0x0002
       Status: active, boot-order
    Partition: /dev/disk/by-partuuid/…
         File: └─/EFI/fedora/fwupdx64.efi
```

```
Boot Loader Entries:
        $BOOT: /boot/efi (/dev/disk/by-partuuid/01234567-89ab-cdef-dead-beef00000000)

Default Boot Loader Entry:
         type: Boot Loader Specification Type #1 (.conf)
        title: Fedora Linux 36 (Workstation Edition)
           id: …
       source: /boot/efi/loader/entries/entry-token-kernel-version.conf
      version: kernel-version
   machine-id: …
        linux: /entry-token/kernel-version/linux
       initrd: /entry-token/kernel-version/initrd
      options: root=…

$ bootctl list
Boot Loader Entries:
         type: Boot Loader Specification Type #1 (.conf)
        title: Fedora Linux 36 (Workstation Edition) (default) (selected)
           id: …
       source: /boot/efi/loader/entries/entry-token-kernel-version.conf
      version: kernel-version
   machine-id: …
        linux: /entry-token/kernel-version/linux
       initrd: /entry-token/kernel-version/initrd
      options: root=…

         type: Boot Loader Specification Type #2 (.efi)
        title: Fedora Linux 35 (Workstation Edition)
           id: …
       source: /boot/efi/EFI/Linux/fedora-kernel-version.efi
      version: kernel-version
   machine-id: …
        linux: /EFI/Linux/fedora-kernel-version.efi
      options: root=…

         type: Automatic
        title: Reboot Into Firmware Interface
           id: auto-reboot-to-firmware-setup
       source: /sys/firmware/efi/efivars/LoaderEntries-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
```

In the listing, `(default)` specifies the entry that will be used by default, and `(selected)` specifies the entry that was selected the last time (i.e. is currently running).

Files

`/etc/systemd/uki/`; `/run/systemd/uki/`; `/var/lib/systemd/uki/`; `/usr/local/lib/systemd/uki/`; `/usr/lib/systemd/uki/`
The `link-auto` command discovers the UKI to install from these directories (listed in decreasing order of priority). It uses `kernel.efi` or the versioned `kernel.efi.v/` directory from the highest-priority directory that contains one. Additional sidecar resources (system extension images, configuration extension images, credential files) are picked up from the `extras.d/` subdirectory of each, combined across all directories. `/var/lib/systemd/uki/` is the directory where `systemd-sysupdate(8)` is supposed to place downloaded kernels, system extensions, configuration extensions and credentials.

See Also

`systemd-boot(7)`, UAPI.1 Boot Loader Specification, Boot Loader Interface, `systemd-boot-random-seed.service(8)`

□□□ #1
□ □□□ □□□ 2026-10-31 14:45:53 UTC
□ □□□ □□□ 2026-10-31 14:45:53 UTC