kernel-install

- Description
- Commands
- Compatibility with the kernel build system
- The $BOOT partition
- Options
- Environment variables
  - Environment variables exported for plugins
  - Environment variables understood by kernel-install
- Exit status
- Files
- See Also

systemd

kernel-install

8

kernel-install

Add and remove kernel and initrd images to and from the boot partition

kernel-install

OPTIONS

add

KERNEL-VERSION

KERNEL-IMAGE

INITRD-FILE

kernel-install

OPTIONS

add-all

kernel-install

OPTIONS

remove

KERNEL-VERSION

kernel-install

OPTIONS

inspect

KERNEL-VERSION

KERNEL-IMAGE

INITRD-FILE

kernel-install

OPTIONS

list

Description

`kernel-install` is used to install and remove kernel and initrd images [^1] to and from the boot partition, referred to as `$BOOT` here. It will usually be one of `/boot/`, `/efi/`, or `/boot/efi/`, see below.

`kernel-install` will run the executable files ("plugins") located in the directory `/usr/lib/kernel/install.d/` and the local administration directory `/etc/kernel/install.d/`. All files are collectively sorted and executed in lexicographic order, regardless of the directory in which they live. However, files with identical filenames replace each other. Files in `/etc/kernel/install.d/` take precedence over files with the same name in `/usr/lib/kernel/install.d/`. This can be used to override a system-supplied executables with a local file if needed; a symbolic link in `/etc/kernel/install.d/` with the same name as an executable in `/usr/lib/kernel/install.d/`, pointing to `/dev/null`, disables the executable entirely. Executables must have the extension `.install`; other extensions are ignored.

An executable placed in these directories should return `0` on success. It may also return `77` to cause the whole operation to terminate (executables later in lexicographic order will be skipped).

Commands

The following commands are understood:

`add [[[KERNEL-VERSION] KERNEL-IMAGE] [INITRD-FILE ...]]`
This command takes a kernel version string and a path to a kernel image file as arguments. If the former is omitted, specified as an empty string or as "-" it defaults to the current kernel version, i.e. the same string `uname -r` returns. If the latter is omitted, specified as an empty string or as "-" defaults to `/usr/lib/modules/KERNEL_VERSION/vmlinuz`. Optionally, one or more initrd images may be specified as well (note that plugins might generate additional ones).

The executable files from `/usr/lib/kernel/install.d/*.install` and `/etc/kernel/install.d/*.install` (i.e. the plugins) are called with the following arguments:

```
add KERNEL-VERSION $BOOT/ENTRY-TOKEN/KERNEL-VERSION/ KERNEL-IMAGE [INITRD-FILE ...]
```

The third argument directly refers to the path where to place kernel images, initrd images and other resources for UAPI.1 Boot Loader Specification Type #1 entries (the "entry directory"). If other boot loader schemes are used the parameter may be ignored.

The <ENTRY-TOKEN> string is typically the machine ID and is supposed to identify the local installation on the system. For details see below.

`kernel-install` creates `$BOOT/ENTRY-TOKEN/KERNEL-VERSION`, if enabled (see `$KERNEL_INSTALL_LAYOUT`).

Three default plugins execute the following operations in this case:

`50-depmod.install`
The plugin runs `depmod(8)` for the <KERNEL-VERSION>.

`90-loaderentry.install`
The plugin copies <KERNEL-IMAGE> to `$BOOT/ENTRY-TOKEN/KERNEL-VERSION/linux`. If <INITRD-FILE>s are provided, it also copies them to `$BOOT/ENTRY-TOKEN/KERNEL_VERSION/INITRD-FILE`. This can also be used to prepend microcode before the actual initrd. It also creates a boot loader entry according to the UAPI.1 Boot Loader Specification (Type #1) in `$BOOT/loader/entries/ENTRY-TOKEN-KERNEL-VERSION.conf`. The title of the entry is the <PRETTY_NAME> parameter specified in `/etc/os-release` or `/usr/lib/os-release` (if the former is missing), or "Linux <KERNEL-VERSION>", if unset.

If `$KERNEL_INSTALL_LAYOUT` is not "bls", this plugin does nothing.

`90-uki-copy.install`
If `$KERNEL_INSTALL_STAGING_AREA/uki.efi` exists, the plugin copies it to `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi`. Also, files in `$KERNEL_INSTALL_STAGING_AREA/uki.efi.extra.d"/` are copied to `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi.extra.d/`.

If `$KERNEL_INSTALL_STAGING_AREA/uki.efi` does not exist, and the <KERNEL-IMAGE> argument has a `.efi` extension, the plugin copies the image to `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi`. Also, files in `KERNEL-IMAGE.extra.d/` and `$KERNEL_INSTALL_STAGING_AREA/uki.efi.extra.d"/` are copied to `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi.extra.d/`.

If `$KERNEL_INSTALL_LAYOUT` is not "uki", this plugin does nothing.

`add-all`
This is the same as `add` (see above), but invokes the operation iteratively for every installed kernel in `/usr/lib/modules/`. This operation is only supported on systems where the kernel image is installed in `/usr/lib/modules/KERNEL-VERSION/vmlinuz`.

`remove KERNEL-VERSION`
This command expects a kernel version string as single argument.

The executable files from `/usr/lib/kernel/install.d/*.install` and `/etc/kernel/install.d/*.install` (i.e. the plugins) are called with the following arguments:

```
remove KERNEL-VERSION $BOOT/ENTRY-TOKEN/KERNEL-VERSION/
```

Afterwards, `kernel-install` removes the entry directory `$BOOT/ENTRY-TOKEN/KERNEL-VERSION/` and its contents, if it exists.

Three default plugins execute the following operations in this case:

`50-depmod.install`
The plugin removes the files generated by `depmod` for this kernel again.

`90-loaderentry.install`
The plugin removes the file `$BOOT/loader/entries/ENTRY-TOKEN-KERNEL-VERSION.conf`.

`90-uki-copy.install`
The plugin removes the file `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi` and `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION.efi.extra.d/` directory including its contents.

`inspect [[[KERNEL-VERSION] KERNEL-IMAGE] [INITRD-FILE ...]]`
Takes the same parameters as `add`.

Shows the various paths and parameters configured or auto-detected. In particular shows the values of the various `$KERNEL_INSTALL_*` environment variables listed below, as they would be passed to plugins. The `--json` option can be used to get the output of this verb as a JSON object.

`list`
Shows the various installed kernels. This enumerates the subdirectories of `/usr/lib/modules/`, and shows whether a kernel image is installed there.

Compatibility with the kernel build system

installkernel

OPTIONS

VERSION

VMLINUZ

MAP

INSTALLATION-DIR

When invoked as `installkernel`, this program accepts arguments as specified by the kernel build system's `make install` command. The `VERSION` and `VMLINUZ` parameters specify the kernel version and the kernel binary. The other two parameters (`MAP` and `INSTALLATION-DIR`) are currently ignored.

The `$BOOT` partition

The partition where the kernels and UAPI.1 Boot Loader Specification snippets are located is called `$BOOT`. `kernel-install` determines the location of this partition by checking `/efi/`, `/boot/`, and `/boot/efi/` in turn. The first location where `$BOOT/loader/entries/` or `$BOOT/ENTRY-TOKEN/` exists is used.

Options

The following options are understood:

`--make-entry-directory=yes|no|auto`
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 `add` and `remove`, respectively. The directory is named after the entry token, and is placed immediately below the boot root directory. When `auto`, the directory is created or removed only when the install layout is `bls`. Defaults to `auto`.

`--entry-type=type1|type2|all`
Controls the type of entries handled by the command. This is typically useful when multiple types of boot entries with the same kernel version are installed, and only one should be removed. When `type1` or `type2` is specified, each plugin is invoked with `$KERNEL_INSTALL_BOOT_ENTRY_TYPE` environment variable with the specified value. When `all` is specified, the environment variable will not be set. Defaults to `all`.

`--entry-token=`
Controls how to name and identify boot loader entries for this kernel installation or deletion. 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` (or `$KERNEL_INSTALL_CONF_ROOT/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. Otherwise, a randomly generated machine ID is used.

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_ROOT` 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.

`-v`; `--verbose`
Output additional information about operations being performed.

`--root=root`
Takes a directory path as an argument. All paths will be prefixed with the given alternate <root> path, including config search paths. This is useful to operate on a system image mounted to the

specified directory instead of the host system itself.

`--image=image`
Takes a path to a disk image file or block device node. If specified, all operations are applied to the 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.

Environment variables

Environment variables exported for plugins

If `--verbose` is used, `$KERNEL_INSTALL_VERBOSE=1` will be exported for plugins. They may output additional logs in this case.

`$KERNEL_INSTALL_IMAGE_TYPE=uki|pe|unknown` is set for the plugins to specify the type of the kernel image.

uki
Unified kernel image.

pe
PE binary.

unknown
Unknown type.

`$KERNEL_INSTALL_MACHINE_ID` is set for the plugins to the desired machine-id to use. It's always a 128-bit ID. Normally it is read from `/etc/machine-id`, but it can also be overridden via `$MACHINE_ID` (see below). If not specified via these methods, a fallback value will generated by `kernel-install` and used only for a single invocation.

`$KERNEL_INSTALL_ENTRY_TOKEN` is set for the plugins to the desired entry "token" to use. It's an identifier that shall be used to identify the local installation, and is often the machine ID, i.e. same as `$KERNEL_INSTALL_MACHINE_ID`, but might also be a different type of identifier, for example a fixed string or the `ID=`, `IMAGE_ID=` values from `/etc/os-release`. The string passed here will be used to

name Boot Loader Specification entries, or the directories the kernel image and initial RAM disk images are placed into.

Note that while `$KERNEL_INSTALL_ENTRY_TOKEN` and `$KERNEL_INSTALL_MACHINE_ID` are often set to the same value, the latter is guaranteed to be a valid 32 character ID in lowercase hexadecimals while the former can be any short string. The entry token to use is read from `/etc/kernel/entry-token`, if it exists. Otherwise, a few possible candidates below `$BOOT` are checked for Boot Loader Specification Type 1 entry directories, and if found the entry token is derived from that. If that is not successful, `$KERNEL_INSTALL_MACHINE_ID` is used as fallback.

`$KERNEL_INSTALL_BOOT_ROOT` is set for the plugins to the absolute path of the root directory (mount point, usually) of the hierarchy where boot loader entries, kernel images, and associated resources should be placed. This usually is the path where the XBOOTLDR partition or the ESP (EFI System Partition) are mounted, and also conceptually referred to as `$BOOT`. Can be overridden by setting `$BOOT_ROOT` (see below).

`$KERNEL_INSTALL_LAYOUT=auto|bls|uki|other|...` is set for the plugins to specify the installation layout. Additional layout names may be defined by convention. If a plugin uses a special layout, it is encouraged to declare its own layout name and configure `layout=` in `install.conf` upon initial installation. The following values are currently understood:

bls

Standard UAPI.1 Boot Loader Specification Type #1 layout, compatible with `systemd-boot(7)`: entries in `$BOOT/loader/entries/ENTRY-TOKEN-KERNEL-VERSION[+TRIES].conf`, kernel and initrds under `$BOOT/ENTRY-TOKEN/KERNEL-VERSION/`

Implemented by `90-loaderentry.install`.

uki

Standard UAPI.1 Boot Loader Specification Type #2 layout, compatible with `systemd-boot(7)`: unified kernel images under `$BOOT/EFI/Linux` as `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION[+TRIES].efi`.

Implemented by `90-uki-copy.install`.

other
Some other layout not understood natively by `kernel-install`.

auto
Pick the layout automatically. If the kernel is a UKI set layout to `uki`. If not default to `bls` if `$BOOT/loader/entries.srel` with content `type1` or `$BOOT/ENTRY-TOKEN` exists, or `other` otherwise.

Leaving layout blank has the same effect. This is the default.

`$KERNEL_INSTALL_INITRD_GENERATOR` and `$KERNEL_INSTALL_UKI_GENERATOR` are set for plugins to select the initrd and/or UKI generator. This may be configured as `initrd_generator=` and `uki_generator=` in

`install.conf`, see below.

`$KERNEL_INSTALL_STAGING_AREA` is set for plugins to a path to a directory. Plugins may drop files in that directory, and they will be installed as part of the loader entry, based on the file name and extension: Files named `initrd*` will be installed as <INITRD-FILE>s, and files named `microcode*` will be prepended before <INITRD-FILE>s.

`$KERNEL_INSTALL_ENTRY_NAME` is set for the plugins to the resolved entry name, derived by expanding specifiers in the entry name format string. This is the base name (without file extension) used for UKI files, loader entry configuration files, and similar. By default this is `ENTRY-TOKEN-KERNEL-VERSION`, matching the traditional naming. See `$KERNEL_INSTALL_ENTRY_NAME_FORMAT` below for details on the available specifiers.

Environment variables understood by `kernel-install`

`$KERNEL_INSTALL_CONF_ROOT` can be set to override the location of the configuration files read by `kernel-install`. When set, `install.conf`, `entry-token`, and other files will be read from this directory only. Note that this path is relative to the host, and in particular *symlinks in this directory are resolved relative to the host*, even if `--root=root` is used. This means that it is generally *not* correct to use this variable to specify a directory underneath <root> if symlinks are used there.

`$KERNEL_INSTALL_PLUGINS` can be set to override the list of plugins executed by `kernel-install`. The argument is a whitespace-separated list of paths. `KERNEL_INSTALL_PLUGINS=:` may be used to prevent any plugins from running.

`$MACHINE_ID` can be set for `kernel-install` to override `$KERNEL_INSTALL_MACHINE_ID`, the machine ID.

`$BOOT_ROOT` can be set for `kernel-install` to override `$KERNEL_INSTALL_BOOT_ROOT`, the installation location for boot entries.

The last two variables may also be set in `install.conf`. Variables set in the environment take precedence over the values specified in the config file.

`$KERNEL_INSTALL_ENTRY_NAME_FORMAT` can be set to configure the format string used to derive the entry name for boot loader entries and UKI files. The format string is expanded using specifier expansion. May also be configured as `entry_name_format=` in `install.conf`. The following specifiers are understood:

Specifier	Meaning	Details
`%e`	Entry token	The entry token, same as `$KERNEL_INSTALL_ENTRY_TOKEN`.

Specifier	Meaning	Details
`%v`	Kernel version	The kernel version being installed. Note that unlike the common specifier `%v` which expands to the currently running kernel version (i.e. `uname -r`), here it refers to the target kernel version passed to `kernel-install`.

Specifiers available in `$KERNEL_INSTALL_ENTRY_NAME_FORMAT` {.specifiers}

The resolved entry name must be a valid filename (no path separators). Defaults to `%e-%v`, which produces the traditional `ENTRY-TOKEN-KERNEL-VERSION` naming.

For example, setting `entry_name_format=%M_%A_%a` on a system with `IMAGE_ID=myos`, `IMAGE_VERSION=20250830`, and architecture `x86-64` would produce entry names like `myos_20250830_x86-64`.

Note that the entry name is resolved at the time of each `kernel-install` invocation. If the format string uses specifiers that depend on mutable system state (e.g. `%M`, `%A`, `%H`), and that state changes between installing and removing a kernel, the `remove` operation may compute a different entry name than `add` did, leaving stale files on the boot partition. The `entry_name_format=` setting should therefore remain stable across a kernel's install/remove lifecycle. Additionally, the format should produce a unique name for each installed kernel, e.g. by including `%v` (kernel version) or `%A` (`IMAGE_VERSION`); otherwise installing multiple kernels will overwrite each other's boot entries.

Exit status

If every executable returns 0 or 77, 0 is returned, and a non-zero failure code otherwise.

Files

`/etc/kernel/install.d/*.install`; `/usr/lib/kernel/install.d/*.install`
Drop-in files which are executed by `kernel-install`.

`/etc/kernel/cmdline`; `/usr/lib/kernel/cmdline`; `/proc/cmdline`
Specifies the kernel command line to use. The first of the files that is found will be used. Lines starting with the `#` character are ignored. When running in a container, `/proc/cmdline` is ignored. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details.

`/etc/kernel/devicetree`; `/usr/lib/kernel/devicetree`
Specifies the partial path to the file containing the device tree blob to install with the kernel and

use at boot. The first of the files that is found will be used. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details.

The `devicetree` file contains a path, and this path specifies a location relative to the kernel install tree. A set of locations is checked, including in particular `/usr/lib/modules/KERNEL_VERSION/dtb/`, which is the recommended location to place the dtb files under. For example, with `broadcom/bcm2711-rpi-4-b.dtb` in the `devicetree` file, the device tree blob for the Raspberry Pi 4 Model B would be installed, and the actual file would be `/usr/lib/modules/KERNEL_VERSION/dtb/broadcom/bcm2711-rpi-4-b.dtb`.

`/etc/kernel/tries`; `/run/kernel/tries`; `/usr/lib/kernel/tries`
Read by `90-loaderentry.install` and `90-uki-copy.install`. A numeric value is read from the first file found, and the generated entry file or UKI is named to include it as `$BOOT/loader/entries/ENTRY-TOKEN-KERNEL-VERSION+TRIES.conf` or `$BOOT/EFI/Linux/ENTRY-TOKEN-KERNEL-VERSION+TRIES.efi`, respectively. This is useful for boot loaders such as `systemd-boot(7)` which implement boot attempt counting with a counter embedded in the entry file name. A symlink to `/dev/null` disables boot counting, overriding files lower in the search order. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details.

`/etc/kernel/entry-token`
If this file exists it is read and used as "entry token" for this system, i.e. is used for naming Boot Loader Specification entries. See `$KERNEL_INSTALL_ENTRY_TOKEN` above for details. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details.

`/etc/machine-id`
The content of this file specifies the machine identification <MACHINE-ID>.

`/etc/os-release`; `/usr/lib/os-release`
Read by `90-loaderentry.install`. If available, `PRETTY_NAME=` is read from these files and used as the title of the boot menu entry. Otherwise, `Linux KERNEL-VERSION` will be used.

`/etc/kernel/install.conf`; `/run/kernel/install.conf`; `/usr/local/lib/kernel/install.conf`; `/usr/lib/kernel/install.conf`; `/etc/kernel/install.conf.d/*.conf`; `/run/kernel/install.conf.d/*.conf`; `/usr/local/lib/kernel/install.conf.d/*.conf`; `/usr/lib/kernel/install.conf.d/*.conf`
Configuration file with options for `kernel-install`, as a series of `KEY=`<VALUE> assignments, compatible with shell syntax, following the same rules as described in `os-release(5)`. The first of the files that is found will be used. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details. Drop-in files may also be used to extend the configuration with overrides, see `systemd.unit(5)`.

Currently, the following keys are supported: `MACHINE_ID=`, `BOOT_ROOT=`, `layout=`, `initrd_generator=`, `uki_generator=`, `entry_name_format=`. See the Environment variables section above for details.

`/etc/kernel/uki.conf`
Ini-style configuration file for `ukify(1)` which is only effective when `$KERNEL_INSTALL_LAYOUT` or `layout=` in `install.conf` is set to `uki` and `$KERNEL_INSTALL_UKI_GENERATOR` or `uki_generator=` in

`install.conf` is set to `ukify`, or is unset. `$KERNEL_INSTALL_CONF_ROOT` may be used to override the search path; see below for details.

`/usr/lib/modules/KERNEL-VERSION/`
Location for installed kernel modules and other kernel related resources. For each locally installed kernel a directory named after the kernel version (`uname -r`) is kept.

`/usr/lib/modules/KERNEL-VERSION/vmlinuz`
Location for installed kernel images. This is the recommended location for OS package managers to install kernel images into (as applicable), from which `kernel-install add` then copies it into the final boot partition.

For various cases listed above, if the `$KERNEL_INSTALL_CONF_ROOT` environment variable is set, it will override the search path. The files will be loaded *only* from the directory specified by the environment variable. When the variable is not set, the listed paths are tried in turn, and the first file that exists is used.

See Also

`machine-id(5)`, `os-release(5)`, `depmod(8)`, `systemd-boot(7)`, `ukify(1)`, UAPI.1 Boot Loader Specification

[^1]: Nowadays actually CPIO archives used as an "initramfs", rather than "initrd". See `bootup(7)` for an explanation.

□□□ #1
□ □□□ □□□ 2026-10-31 14:46:42 UTC
□ □□□ □□□ 2026-10-31 14:46:42 UTC