

veritytab



- [Description](#)
 - [Examples](#)
 - [See Also](#)
-

systemd

veritytab

5

veritytab

Configuration for verity block devices

`/etc/veritytab`

Description

The `/etc/veritytab` file describes verity protected block devices that are set up during system boot.

Empty lines and lines starting with the `#` character are ignored. Each of the remaining lines describes one verity protected block device. Fields are delimited by white space.

Each line is in the form

```
volume-name data-device hash-device roothash options
```

The first four fields are mandatory, the remaining one is optional.

The first field contains the name of the resulting verity volume; its block device is set up below `/dev/mapper/`.

The second field contains a path to the underlying block data device, or a specification of a block device via `UUID=` followed by the `<UUID>`.

The third field contains a path to the underlying block hash device, or a specification of a block device via `UUID=` followed by the `<UUID>`.

The fourth field is the `<roothash>` in hexadecimal. If this field is specified as dash, it is attempted to read the root hash from the udev property `ID_DISSECT_PART_R00THASH=` (encoded in hexadecimal) of the data device.

The fifth field, if present, is a comma-delimited list of options. The following options are recognized:

`superblock=B00L`

Use dm-verity with or without permanent on-disk superblock.

`format=NUMBER`

Specifies the hash version type. Format type `0` is original Chrome OS version. Format type `1` is modern version.

`data-block-size=BYTES`

Used block size for the data device. (Note kernel supports only page-size as maximum here; Multiples of 512 bytes.)

`hash-block-size=BYTES`

Used block size for the hash device. (Note kernel supports only page-size as maximum here; Multiples of 512 bytes.)

`data-blocks=BLOCKS`

Number of blocks of data device used in verification. If not specified, the whole device is used.

`hash-offset=BYTES`

Offset of hash area/superblock on `hash-device`. (Multiples of 512 bytes.)

`salt=HEX`

Salt used for format or verification. Format is a hexadecimal string; 256 bytes long maximum; `-` is the special value for empty.

`uuid=UUID`

Use the provided `<UUID>` instead of generating new one. The `<UUID>` must be provided in standard UUID format, e.g. `12345678-1234-1234-1234-123456789abc`.

`ignore-corruption`; `restart-on-corruption`; `panic-on-corruption`

Defines what to do if a data verity problem is detected (data corruption). Without these options kernel fails the IO operation with I/O error. With `--ignore-corruption` option the corruption is only logged. With `--restart-on-corruption` or `--panic-on-corruption` the kernel is restarted (panicked) immediately. (You have to provide way how to avoid restart loops.)

`ignore-zero-blocks`

Instruct kernel to not verify blocks that are expected to contain zeroes and always directly return zeroes instead.

“ [!WARNING] Use this option only in very specific cases. This option is available since Linux kernel version 4.5.

`check-at-most-once`

Instruct kernel to verify blocks only the first time they are read from the data device, rather than every time.

“ [!WARNING] It provides a reduced level of security because only offline tampering of the data device's content will be detected, not online tampering. This option is available since Linux kernel version 4.17.

`hash=HASH`

Hash algorithm for dm-verity. This should be the name of the algorithm, like "sha1". For default see `veritysetup --help`.

`fec-device=PATH`

Use forward error correction (FEC) to recover from corruption if hash verification fails. Use encoding data from the specified device. The fec device argument can be block device or file image. If fec device path does not exist, it will be created as file. Note: block sizes for data and hash devices must match. Also, if the verity data_device is encrypted the fec_device should be too.

`fec-offset=BYTES`

This is the offset, in bytes, from the start of the FEC device to the beginning of the encoding data. (Aligned on 512 bytes.)

`fec-roots=NUM`

Number of generator roots. This equals to the number of parity bytes in the encoding data. In RS(M, N) encoding, the number of roots is M-N. M is 255 and M-N is between 2 and 24 (including).

`root-hash-signature=PATH|base64:BASE64|auto`

A Base64 string encoding the root hash signature prefixed by `base64:`, or an absolute path to a root hash signature file used to verify the root hash (in kernel). If the special string `auto` is specified, the root hash signature is attempted to be read from the udev property

`ID_DISSECT_PART_R00THASH_SIG=` (in Base64 format) of the data device. This feature requires Linux kernel version 5.4 or more recent.

`_netdev`

Marks this veritysetup device as requiring network. It will be started after the network is available,

similarly to `systemd.mount(5)` units marked with `_netdev`. The service unit to set up this device will be ordered between `remote-fs-pre.target` and `remote-veritysetup.target`, instead of `veritysetup-pre.target` and `veritysetup.target`.

Hint: if this device is used for a mount point that is specified in `fstab(5)`, the `_netdev` option should also be used for the mount point. Otherwise, a dependency loop might be created where the mount point will be pulled in by `local-fs.target`, while the service to configure the network is usually only started *after* the local file system has been mounted.

`noauto`

This device will not be added to `veritysetup.target`. This means that it will not be automatically enabled on boot, unless something else pulls it in. In particular, if the device is used for a mount point, it'll be enabled automatically during boot, unless the mount point itself is also disabled with `noauto`.

`nofail`

This device will not be a hard dependency of `veritysetup.target`. It'll still be pulled in and started, but the system will not wait for the device to show up and be enabled, and boot will not fail if this is unsuccessful. Note that other units that depend on the enabled device may still fail. In particular, if the device is used for a mount point, the mount point itself also needs to have the `nofail` option, or the boot will fail if the device is not enabled successfully.

`x-initrd.attach`

Setup this verity protected block device in the initrd, similarly to `systemd.mount(5)` units marked with `x-initrd.mount`.

Although it is not necessary to mark the mount entry for the root file system with `x-initrd.mount`, `x-initrd.attach` is still recommended with the verity protected block device containing the root file system as otherwise systemd will attempt to detach the device during the regular system shutdown while it is still in use. With this option the device will still be detached but later after the root file system is unmounted.

All other verity protected block devices that contain file systems mounted in the initrd should use this option.

`tpm2-measure-nvpcr=`

Takes a boolean argument, or an NvPCR name as argument. May be used to enable automatic measurement of the volume name and Verity root hash, as well as the serials and issuers of the certificates used to generate the provided signatures (if any) will be measured. Passing false disables the mechanism. Passing true enables measurement into the `verity` NvPCR. If any other string is specified this selects the NvPCR to measure into by name.

At early boot and when the system manager configuration is reloaded, this file is translated into native systemd units by `systemd-veritysetup-generator(8)`.

Examples

/etc/veritytab example

Set up two verity protected block devices. One using device blocks, another using files.

```
usr PARTUUID=783e45ae-7aa3-484a-beef-a80ff9c19cbb PARTUUID=21dc1dfe-4c33-8b48-98a9-918a22eb3e37 36e3f740ad502e2c25e2a23d9c7c17bf0fdad2300b7580842d4b7ec1fb0fa263 auto
data /etc/data /etc/hash a5ee4b42f70ae1f46a08a7c92c2e0a20672ad2f514792730f5d49d7606ab8fdf auto
```

See Also

`systemd(1)`, `systemd-veritysetup@.service(8)`, `systemd-veritysetup-generator(8)`, `fstab(5)`, `veritysetup(8)`

<div><div></div><div></div></div>	<div>#1</div>	
<div><div></div><div></div></div>	<div></div>	<div>2026-10-31 15:08:43 UTC</div>
<div><div></div><div></div></div>	<div></div>	<div>2026-10-31 15:08:43 UTC</div>