bionic (5) bilibop.conf.5.gz

Provided by: bilibop-common_0.5.4_amd64

NAME

```
       bilibop.conf - configuration file of bilibop packages
```

SYNOPSIS

```
       /etc/bilibop/bilibop.conf
```

DESCRIPTION

bilibop.conf is the configuration file of **bilibop-*** packages, which are intended to be used on systems running from an external and writable media (USB, FireWire, MMC, eSATA). It is composed of **VARIABLE**=**VALUE** pairs, where **VARIABLE** is a string beginning by 'BILIBOP_', and **VALUE** must be inserted between quotes if it contains blank characters (spaces or tabulations). Spaces around the equal sign (=) are not allowed. Empty lines or lines beginning by a hash sign (#) are ignored.

Valid configuration options can be divided in 'common' and 'specific' sections, as follows:

COMMON VARIABLES

BILIBOP_COMMON_BASENAME

This variable defines the basename of bilibop subdirectories (or symlink) that will be created in /dev and /run at boot time, from into the initramfs environment or from into the running system. If it is empty or unset, the value will fallback to 'bilibop'. If you modify it to anything else, you have to take care that some symlinks or custom settings of your system reflect the new location. At least, if **bilibop-rules** is installed, you should execute the helper scripts provided by this package, either by hand or with '**dpkg-reconfigure bilibop-rules**'. So, maybe it is not a good idea to modify it.

Default is unset.

BILIBOP-LOCKFS SPECIFIC VARIABLES

BILIBOP_LOCKFS

This variable defines the main behaviour of the bilibop-lockfs initramfs script. It is a boolean:

If set to false, the system will boot normally, and other BILIBOP_LOCKFS_* variables will be ignored, except **BILIBOP_LOCKFS_NOTIFY_POLICY**. If set to true, the initramfs script will move the normal root filesystem to another mountpoint used as the lower/readonly branch of an **aufs**(5) or **overlay** mountpoint (depending on the version of your kernel) used itself as the actual root filesystem. After what the temporary and writable /etc/fstab file will be modified to prepare other filesystems — if not whitelisted — to be mounted as readonly branches of aufs or overlay too.

If empty, unset, or set to anything else, then a heuristic will be used to set it to true or false, depending on the removable flag of the disk in the sysfs attributes, knowing that generally USB keys are seen as removable devices (true), and USB HDDs are seen as non removable devices (false).

In all cases, the value of this variable can be overridden from the boot commandline by adding '**lockfs**' or '**nolockfs**' to the line of kernel parameters. However, if the drive is physically locked by a switch, this will be detected and all previous settings will be overridden to set **BILIBOP_LOCKFS** as true.

Default is unset.

BILIBOP_LOCKFS_POLICY

This variable defines an additional 'lock' level to be enabled or not. Its value can be overridden from the boot commandline with '**lockfs=hard**' or '**lockfs=soft**'.

- soft

The readonly branches of **aufs**(5) or **overlay** mountpoints will be set to '**ro**' (readonly). Later, these readonly filesystems can be remounted manually as writable to allow root to save some changes on them. This kind of action is highly discouraged here, but is possible if you REALLY know what you do; otherwise it can lead to unexpected results, including of course data loss.

- hard

  The readonly branches of aufs mountpoints will be set to '**rr**' (real readonly); this is used by aufs to optimize some internal operations. Additionally, the corresponding block devices will be set as readonly too by using the **read_only_volume_list** variable in **lvm.conf**(5) for Logical Volumes, or **blockdev**(8) for other block devices, avoiding low-level write access to them (even by root) and avoiding the filesystems to be remounted later as writable.

If empty, unset, or set to anything else, the value will fallback to hard. However, if the drive is physically locked, previous settings will be overridden and the hard policy will be automatically applied.

Default is unset.

BILIBOP_LOCKFS_PATH_PREFIX

This variable defines the main directory under which all readonly and writable branches are set. It may be any arbitrary string, as long as it is a valid name and the directory does not exist yet.

If empty or unset, it defaults to the name of the module currently in use (i.e. aufs or overlay).

Default is unset.

BILIBOP_LOCKFS_PATH_SCHEME

This variable defines the structure of each set of ro and rw branches, regarding the others. It exists only because unlike aufs, overlay does not allow one to easily nest mount points (especially the **upperdirs**).

- isolated
  Each set of readonly and writable branches is created into a dedicated directory. For example, to set up an aufs or overlay for /usr/local, **/$union/usr/local**/ro and **/$union/usr/local**/rw are used. So with this scheme, branches related to a specific mount point are clearly identified, but symlinks crossing filesystem boundaries are broken (on the branches, not on their union mount).

- nested
  All readonly branches are set under the same reaonly sub-tree, and all writable branches are set under the same writable sub-tree. So branches are easily browsable, and symlinks are preserved. For example, to set an aufs for /usr/local, **/aufs/ro**/usr/local and **/aufs/rw**/usr/local are used. This scheme is not avalaibable with **overlay**, and is the default with **aufs**, for backward compatibility with versions of bilibop until 0.4.23.

- hybrid
  Readonly branches are nested, and writable branches are isolated.

If empty, unset, or set to anything else, the value will fallback to nested with aufs, and isolated with overlay.

Default is unset.

BILIBOP_LOCKFS_WHITELIST

One time the root filesystem is locked as the readonly branch of an aufs or overlay filesystem, the /etc/fstab file is modified on the writable branch to lock all other local filesystems as readonly branches of aufs or overlay mountpoints. This variable gives the ability to avoid the **lockfs** mechanism for some mountpoints: this is a whitespace separated list of mountpoints or device names (as known in **fstab**(5)) or tokens of the form **UUID**=**fsuuid**, **LABEL**=**fslabel** or **TYPE**=**fstype**. If the LABEL of a device contains spaces, replace them by underscores (_), as given by the output of '**udevadm** info --query property --name DEVICE' or '**blkid** -o udev -p DEVICE' for ID_FS_UUID, ID_FS_LABEL and ID_FS_TYPE variables. Note that whitelist a mountpoint, a device name or any token matching the corresponding fstab entry makes the device is whitelisted by the initramfs script, that is faster. Otherwise, **mount.lockfs**(8) will query metadata about the device to check if it must skip it or not.

Also note that it is possible to override (and blank) the value of this variable by adding '**lockfs=all**' on the boot commandline. This is also automatically done when the drive is physically locked. On the contrary, to append mountpoints to this whitelist from the boot commandline, it is also possible to use an option of the form '**lockfs=-/foobar**', where /foobar is the mountpoint to not lock; not that it is preceded by a minus sign (**-**).

Default is unset.

BILIBOP_LOCKFS_SIZE

By default, **bilibop-lockfs** allocates half of RAM size (or TMPFS_SIZE if set in /etc/default/tpmfs) for each aufs or overlay writable branch of a locked filesystem. It is possible to override this value for some mountpoints in a whitespace separated list of **mountpoint**=**size** pairs. Sizes can be absolute (suffixed with k, K, m, M, g or G), or relative to the total amount of RAM (and suffixed with %). The size allocated to the root filesystem can be fixed here too, but can be overridden from the boot commandline with a '**lockfs**=**size**' kernel parameter.

Default is unset.

BILIBOP_LOCKFS_SWAP_POLICY

This variable defines what to do with swap devices listed in /etc/fstab (and optionally in /etc/crypttab). Generally, there is no sense to setup a swap device on a flash memory stick, but this can be done on USB, FireWire or eSATA HDDs. Five policies are available:

- soft
  Nothing is changed: lines in **fstab**(5) and **crypttab**(5) are kept as is.

- hard
  Swap entries in fstab and crypttab are disabled (commented).

- noauto
  The 'noauto' keyword is appended to the list of options of swap entries in fstab and crypttab. This means swap devices can be enabled manually with **swapon**(8).

- crypt
  Entries about encrypted swap devices are kept as is, others are disabled. **ATTENTION:** this option makes no difference between swap devices encrypted with a random key (and whose the content is unrecoverable after system halt) and those whose the content is written in clear on a Logical Volume being itself included in an encrypted Volume Group.

- random
  Entries about swap devices encrypted with a random key are kept as is, others are disabled.

If BILIBOP_LOCKFS_SWAP_POLICY is not set to a known value, crypt or hard are the fallbacks, depending on the removable flag of the disk in the sysfs attributes: for devices seen as removable (USB sticks), the policy is to not use swap devices at all (_hard_ policy). Note that in all cases, swap usage can be disabled from the boot commandline with the _noswap_ kernel parameter, which is not a **bilibop**(7) specific boot option, but leads to set BILIBOP_LOCKFS_SWAP_POLICY to _hard_. This is also the case if the script detects that the drive is physically locked.

Default is unset.

BILIBOP_LOCKFS_NOTIFY_POLICY

This variable defines when to notify the user that filesystems are locked or not. Such notifications can be sent at system boot (needs **plymouth** package installed to work) as well as desktop session startup (needs **libnotify-bin** package installed to work). What follows describes desktop notifications; **plymouth**(8) messages are less verbose. There are four available policies:

- _always_
  This is the fallback when the variable is unset or set to something else than _never_, _lockfs_ or _nolockfs_. If the **bilibop-lockfs** feature is disabled, then a notification will be send to say that all information of the session can be written on the disk. If the feature is enabled, a notification will be send to say that all changes under the (listed) aufs or overlay mountpoints will be lost at shutdown. If some mountpoints have been whitelisted, a second notification will be sent to say that all changes under them will be kept at shutdown.

- _never_
  Never send notification about filesystems status.

- _lockfs_

If the **bilibop-lockfs** feature is enabled, then a notification will be send to say that all changes under aufs or overlay mountpoints will be lost at shutdown.

- *nolockfs*
  If the **bilibop-lockfs** feature is disabled, does the same thing as for *always*. If the feature is enabled and some mountpoints have been whitelisted, then a notification will be send to say that all changes under them will be kept at shutdown.

In all cases, any user can (for its own desktop session) override the admin settings by copying *lockfs-notify.desktop* (normally in */etc/xdg/autostart*) in its own *.config/autostart* directory and by modifying the lines beginning by **Exec=** or **Hidden=**. See **lockfs-notify**(1) for details.

Default is unset.

BILIBOP-RULES SPECIFIC VARIABLES

Unlike the previous variables whose modifications take effect only after the system has been rebooted, most of the following BILIBOP_RULES_* variables — except the first one — can be modified, and the changes applied during a same session by running '**lsbilibop -c**'. See **lsbilibop**(8).

BILIBOP_RULES_FAKE_DEVICE_MAP

By default, **bilibop**(7) rules build a */boot/grub/device.map* style-file named *grub-device.map* in the bilibop subdirectory in */run* (defined by the BILIBOP_COMMON_BASENAME variable). The goal is to map the removable device hosting the running system as **(hd0)**, i.e. as the first disk in the BIOS boot sequence. To make this faked map usable by **update-grub**(8), the file */boot/grub/device.map* must be replaced by a symlink to it. If it is the case, but you don't want to build this map, and then use a real map built on the fly by **grub-mkdevicemap**(8), explicitly set this to _false_ (all other values have no effect, i.e. have the same effect than _true_).

Default is unset.

BILIBOP_RULES_SYSTEM_INTERNAL

By default, bilibop rules use **udisks** (both versions **1.x** and **2.x**) facilities to override the usual bus type detection of whether a device is considered 'system internal'. This means root privileges will be needed to manage devices hosted by the same disk than the root filesystem. If you don't need this global behaviour, explicitly set this to _false_ (all other values have no effect, i.e. have the same effect than _true_).

Default is unset.

BILIBOP_RULES_SYSTEM_INTERNAL_WHITELIST

If BILIBOP_RULES_SYSTEM_INTERNAL is not 'false', all partitions hosted on the same disk than the root filesystem will be considered as 'system internal'. To disable this behaviour for only some devices — for example if you want a partition mountable/unmountable without needs of root privileges — you can list them here, separated by spaces. For each device or group of devices, you must specify at least one token of the form **UUID**=**fsuuid**, **LABEL**=**fslabel**, **TYPE**=**fstype** or **USAGE**=**fsusage**. If the LABEL of a device contains spaces, replace them by underscores (_), as given by the output of '**udevadm** info --query property --name _DEVICE_' or '**blkid** -o udev -p _DEVICE_' for _ID_FS_UUID_, _ID_FS_LABEL_, _ID_FS_TYPE_ and _ID_FS_USAGE_ variables.

Default is unset.

BILIBOP_RULES_PRESENTATION_HIDE

By default, bilibop rules hide (if possible) the filesystems contained on the same physical hard disk or memory stick than the root filesystem. This applies to desktop applications based on **udisks** (both versions **1.x** and **2.x**). If you don't want to hide the bilibop volumes, explicitly set this to _false_ (all other values have no effect, i.e. have the same effect than _true_).

Default is unset.

BILIBOP_RULES_PRESENTATION_HIDE_WHITELIST

If BILIBOP_RULES_PRESENTATION_HIDE is not 'false', all volumes hosted on the same disk than the root filesystem will be hidden to the user. To disable this behaviour for only some devices, you can list them here, separated by spaces. For each device or group of devices, you must specify at least one token of the form **UUID**=**fsuuid**, **LABEL**=**fslabel**, **TYPE**=**fstype** or **USAGE**=**fsusage**. If the LABEL of a device contains spaces, replace them by underscores (_), as given by the output of '**udevadm** info --query property --name _DEVICE_' or '**blkid** -o udev -p _DEVICE_' for _ID_FS_UUID_, _ID_FS_LABEL_, _ID_FS_TYPE_ and _ID_FS_USAGE_ variables.

Default is unset.

BILIBOP_RULES_PRESENTATION_ICON

If a device is not hidden, it can be shown to the user with another icon than the default one. For each device or group of devices you want to change the default icon, you must specify at least one token of the form **UUID**=**fsuuid:icon**, **LABEL**=**fslabel:icon**, **TYPE**=**fstype:icon** or **USAGE**=**fsusage:icon**. The icon name must follow the freedesktop.org icon theme specification. If the LABEL of a device contains spaces, replace them by underscores (_), as given by the output of '**udevadm** info --query property --name _DEVICE_' or '**blkid** -o udev -p _DEVICE_' for _ID_FS_UUID_, _ID_FS_LABEL_, _ID_FS_TYPE_ and _ID_FS_USAGE_ variables.

Default is unset.

BILIBOP_RULES_PRESENTATION_NAME

If a device is not hidden, it can be shown to the user with another name than the default one (generally the label of the filesystem). For each device or group of devices you want to change the default name, you must specify at least one token of the form **UUID**=**fsuuid**:**name**, **LABEL**=**fslabel**:**name**, **TYPE**=**fstype**:**name** or **USAGE**=**fsusage**:**name**. If the LABEL of a device contains spaces, replace them by underscores (_), as given by the output of '**udevadm** info --query property --name <u>DEVICE</u>' or '**blkid** -o udev -p <u>DEVICE</u>' for <u>ID_FS_UUID</u>, <u>ID_FS_LABEL</u>, <u>ID_FS_TYPE</u> and <u>ID_FS_USAGE</u> variables.

Default is unset.

FILES

```
/etc/bilibop/bilibop.conf
/usr/share/doc/bilibop-common/examples/bilibop.conf
/usr/share/doc/bilibop-lockfs/examples/bilibop.conf
/usr/share/doc/bilibop-rules/examples/bilibop.conf
```

SEE ALSO

aufs(5), **bilibop**(7), **blkid**(8), **crypttab**(5), **fstab**(5), **lockfs-notify**(1), **lsbilibop**(8), **mount**(8), **mount.lockfs**(8), **notify-send**(1), **plymouth**(8), **proc**(5), **udev**(7), **udevadm**(8),

```
        udisks(7), udisks(8)
```

AUTHOR

```
        This manual page has been written by Bilibop Project <quidame@poivron.org>.
```