NAME

bilibop - run Debian GNU/Linux from an external media

DESCRIPTION

A lot of GNU/Linux distributions - at least the most popular of them - provide freely downloadable *.iso* or *.img* disk images that can be copied on a USB memory stick (sometimes with just cat(1) or dd(1), sometimes in a more complicated way) and immediatly usable 'as is'.

But such operating systems are not designed to be modified; they are read-only, and even when they provide a 'persistent' feature, it is limited. Additionally, they are currently unmaintainable, in the sense that rebuild the complete image of the root filesystem is the only way to update the system or modify its settings in depth. This is often a hard or heavy task that cannot be done from the system itself: this needs a dedicated work space, outside of the running system, and this often needs another operating system to replace the disk image by the new one; and some of these tasks can be done only by experienced users. Others have to wait for the next release, if it comes a day.

Bilibop stands for '**B**ilibop **I**s **L**ive **I**nstall **B**oot **O**n **P**endrive'. This recursive acronym is now obsolete, but the name has been kept. The **bilibop project** is born as an alternative to the **LiveUSB** systems.

By performing a standard installation of Debian directly on a removable media — currently a USB key or an external HDD — it is possible to use it as a LiveUSB system, with the big difference that it behaves like any installed Debian OS: it can be maintained, modified, updated, or even broken by the root user at any time. In fact, without specific settings, it can be broken by an unprivileged user at any time; but this is also the case of LiveUSB systems.

So, **bilibop** is a collection of scripts using or used by other programs (**initramfs-tools**(7), udev(7), aufs(5), or **GRUB2**) to help admins to maintain a **Debian GNU/Linux** operating system installed on a removable and writable media, even if some of these scripts may also be used in other contexts. One of its main goals is to fix security issues or harden standard rules and policies, to make the system more robust in this particular situation. Instead of yet another new, living fast and dying young, Debian based distribution, bilibop has been designed as a set of few debian packages. **bilibop-lockfs** may also be installed on a laptop or on a public computer as an alternative to **fsprotect** or **overlayroot**, and **bilibop-udev** (or **bilibop-rules**) *should* also be installed on a LiveUSB.

BILIBOP PACKAGES

bilibop This is a meta package, depending on several other binary packages from the same **bilibop** source package.

bilibop-common
It mainly provides shell functions and documentation. See *README.Debian* in the documentation of the package for details about these functions. It also includes the drivemap(1) command.

bilibop-rules
This package provides udev rules and helper scripts. Its main purpose is to fix the external drive hosting the running system, and all its partitions, as owned by the '*disk*' group instead of '*floppy*', as done by the common udev rules applied to removable media. This is a workaround of the bug **#645466**. The udev rules provided by this package work even when the root filesystem is on a **LUKS** device, a **LVM** Logical Volume, a **loop** device or is an aufs(5) mountpoint. **bilibop-rules** also includes the lsbilibop(8) command, and some helper scripts in */usr/share/bilibop*, that can be executed manually or with '**dpkg-reconfigure bilibop-rules**'. See *README.Debian* in the documentation of the package for details.

bilibop-udev
This package is a kind of subset of **bilibop-rules**, and is more suited for LiveUSB systems. It just makes that the drive hosting the running system, and all its partitions, belong to the '**disk**' group instead of '**floppy**'. Its udev rules also create a symlink (*/dev/bilibop*) pointing to the drive name. See *README.Debian* in the documentation of the package for details.

bilibop-lockfs
By using an initramfs script and a mount(8) helper script, filesystems are mounted as readonly branches of aufs(5), the corresponding writable branches being on temporary filesystems. Additionally, block devices are set readonly too, avoiding low-level write access on them, even by root. All this makes the operating system unbreakable, unless with a hammer. See *README.Debian* in the documentation of the package for details.

INSTALLATION

Debian can be installed on a removable drive as it is on an internal one, except:

- It is highly recommended to install a full encrypted system. Otherwise, what can happen if the USB stick or the external HDD has been lost or forgotten somewhere, or even thieft ? Unfortunately (but there are evident security reasons), this can not be fully preseeded.

- Due to write-cycles limits on flash memory, it is not recommended to use a swap area on them: this can dramatically decrease the lifetime of the drive.

- Even if the **amd64** is now the most common architecture on modern **P**ersonal **C**omputers, installation of a **x86** system will make it more versatile and work both on amd64 and i386 architectures (and even on ia32, but this needs at least a specific partition scheme).

- Take care, near the end of the installation, that the bootloader will be installed on the MBR of the drive where the system has been freshly installed: choosing the default 'install on MBR' will install it on the Master Boot Record of the first disk !

- Taking previous recommendations into account, choose '**Expert Install**' or '**Expert Graphical Install**' in the installer boot menu. if you have to install Debian on several devices, don't perform an automated installation via the '**Auto Install**' option in the installer boot menu. If you really need to automate this process to win time, use a **preseed** file instead.

SETTINGS AND CONFIGURATION

The main advantage of a standard installation over a Live system is that the installed one can exactly answer your needs: if the needs change, the system can be easily modified. It can be installed and configured to be used as/for:

• daily usage (this is my case)

• router and/or firewall for a LAN

• ftp and/or http server (this is my case)

• forensics and rescue system (this is may case)

• embedded Debian repository (this is my case)

• testing system

• educational purposes

• others

Because an operating system running from an external device is generally used on different computers, with potentially different keyboards, architectures, monitors, and so on, it could need some special settings to be as versatile as possible. Maybe the field is too large to be covered into a single manual page: see */usr/share/doc/bilibop-common/misc/** for some tips and tricks, details and suggestions about possible settings.

FILES

/usr/share/bilibop-common/README.Debian

/usr/share/bilibop-common/examples/bilibop.conf

/usr/share/bilibop-common/misc/*

/usr/share/bilibop-lockfs/README.Debian

/usr/share/bilibop-lockfs/examples/bilibop.conf

/usr/share/bilibop-rules/README.Debian

/usr/share/bilibop-rules/examples/bilibop.conf

SEE ALSO

bilibop.conf(5), drivemap(1), lsbilibop(8)

AUTHOR

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

2013-10-26 bilibop