Operating principle

To create a live Slackware system, two Slackware systems are needed:

- the first one, with **"SysLinux"** (CSM (Compatibility Support Module) booting), **"eLiLo"** (UEFI (Unified Extended Firmware Interface) booting), **"Slackware-Live"**, **"Unionfs-FUSE"** (if You don't plan to use AUFS) and **"SquashFS-Tools"** will be used to build the live system;
- the second one is the system to make live; this system must have a **kernel** (Slackware stock huge SMP kernel is recommended), **modules** and the **"MkInitRD"** utility; the utilities **"SysLinux"**, **"LiLo"** and **"eLiLo"** are needed if You want the live system able to be copied to an USB key and/or installable on a disk partition.

Once the system has been setup, the building of the live system needs three commands:

- **"build-slackware-live.sh --init ..."** to create the initrd and copy the kernel;
- **"build-slackware-live.sh --module ..."** to create a SquashFS module of the system;
- **"build-slackware-live.sh --iso ..."** to create an ISO image of the live-system or **"build-slackware-live.sh --usb ..."** to copy it on USB key.

Example

- Setup your system:
  - From a Slackware Linux system — the build system —, install wanted packages into the **"/mnt/system"** — the live system root — directory:
    ```
    installpkg -root /mnt/system /mnt/cdrom/slackware/a/*.t?z
    ```
  - Run setup (recommended) and add a user (if needed):
    ```
    chroot /mnt/system #live system
    pkgtool #setup
    useradd -m -g users -s /bin/bash liveuser
    usermod -G floppy,cdrom,netdev,plugdev,scanner,lp,audio,video,power liveuser
    passwd liveuser
    exit #chroot
    ```
- Build the live system:
  ```
  build-slackware-live.sh --init /mnt/system /tmp/live
  build-slackware-live.sh --module /mnt/system /tmp/live 0-slackware-live
  build-slackware-live.sh --iso /tmp/live /tmp/slackware-live.iso
  ```

*It is possible to have a single system that transforms itself into a live system: just use **"/"** for the system root directory parameter (instead of **"/mnt/system"** in the example above); be carreful: all directories but **"/sys"**, **"/proc"**, **"/dev"** and **"/tmp"** are included in the live system; think to umount removable devices for example.*

System setup

Live system is just an ordinary Slackware

Install the live system like any other Slackware Linux (or derived distribution):

- make a regular install from the distribution install media, then, access this system by mounting its partition from the build system:
  ```
  mkdir /mnt/system
  mount /dev/installation_device /mnt/system
  ```
- or install packages into a subdirectory of the build system (the live system root directory) — with **"installpkg -root /mnt/system ..."** for example —.

Installing from a list of selected packages

The build system includes an option to install a list of selected packages into a subdirectory:

- usage: **"build-slackware-live.sh --add packages_dir root_dir pkg_list_file"**
- example: **"build-slackware-live.sh --add /mnt/cdrom /mnt/system packages-list.txt"**

packages-list.txt example

```
slackware/a/*
slackware/n/dhcpcd
slackware/n/iputils
slackware/n/net-tools
slackware/n/network-scripts
postinstall=ln -sf ifconfig usr/bin/ifcfg
postinstall=echo "live.slackware.org" > etc/HOSTNAME
```

Sysprep

Running Sysprep is recommanded before building the live system and is required if you want to speed up live system boot (see **"fastboot"** boot parameter); it consists on the following actions:

- run commands like **"mkfontdir"** (... — ordinary run by **"pkgtool"** during setup);
- run commands like **"fc-cache"**, **"update-pango-querymodules"** (... — run by stock Slackware **"rc.M"** startup script);
- run **"ldconfig"** and **"depmod"**;
- merge **"/etc/passwd"** and **"/etc/group"** system files (if the system is divided into multiple directories);
- set up root and users profile (from **"/etc/skel"**).

- usage: **"build-slackware-live.sh --sysprep dir_1(rw) dir_2(ro) dir_3(ro)..."**
  only the first directory parameter is mandatory; if the system is divided into multiple directories (one per module), list all of them to recompose the whole system; setup changes will be written into the first one
- example: **"build-slackware-live.sh --sysprep /mnt/system-gui /mnt/system-core"**

Live system build commands

Setup kernel and initrd

- usage: **"build-slackware-live.sh --init root_dir live_dir [modules_list]"**
  *the **"modules_list"** defaults to **"squashfs:fuse:loop"** (for Slackware stock huge kernel)*
  "modules_list"** example: **"squashfs:fuse:loop:isofs:nls_utf8:xhci-hcd:ehci-hcd:usb-storage"
- example: **"build-slackware-live.sh --init /mnt/system /tmp/live"**

*It is possible to have custom **"elilo.conf"** and **"syslinux.cfg"**: they are not overwritten by **"--init"** command; if **"elilo.conf"** is modified, run **"--init"** again to have it copied inside **"efi.img"** file.*

Create a SquashFS module for the system

- usage: **"build-slackware-live.sh --module root_dir live_dir module_file [-xz|-gzip] [-optional]"**
  *with the **"-xz"** or **"-gzip"** option, it is possible to choose SquashFS compression method: the live system will be about 20 % smaller with xz, but about 20% faster with gzip — the provided SquashFS package uses gzip by default*
  *with the **"-optional"** option, the module is stored in directory **"live_system_dir/boot/optional"** instead of **"live_system_dir/boot/modules"***
- example - for live system build (typically):
  "build-slackware-live.sh --module /mnt/system /tmp/live 0-slackware-live"
- example - to store changes made on live USB while running:
  "build-slackware-live.sh --module /live/changes /live/media 1-changes"
- remarq: **"-xz"** compression method needs at least a 2.6.38 kernel (or a patched one)

Warning, on FAT filesystem (if using EFI partition for example), file size is limited to 4GB; split the live system into multiple modules if needed.

Using multiple modules

The system can be divided into multiple modules (example: "core", "gui", "tools",...) that are loaded in alphabetical order; if a file is present into several modules, the one taken from the last loaded module is used; naming example:

- 1-an_application
- 1-an_other_application
- ...
- 2-gui
- 3-core
- 4-2012-01-01-updates
- 4-2012-02-01-updates
- ...

Copy live system on USB device

*Warning: if the destination is a whole disk or key (**"/dev/sdb"** for example), it is automaticaly repartitionned (and all data on it are wiped); on slackware 64, the partitionning scheme uses an hybrid MBR and a 32MB EFI partition is created.*

- usage: **"build-slackware-live.sh --usb live_dir device"**
- example - after initialization and module creation:
  "build-slackware-live.sh --usb /tmp/live /dev/sdx1"
- example - from a running live system:
  "build-slackware-live.sh --usb /live/media /dev/sdx1"

Create a live CD/DVD ISO from live system

- usage: **"build-slackware-live.sh --iso live_dir iso_file_name"**
- example - after initialization and module creation:
  "build-slackware-live.sh --iso /tmp/live /tmp/slackware-live.iso"
- example - from a running live system:
  "build-slackware-live.sh --iso /live/media /mnt/sdx1/slackware-live.iso"

Hybrid ISO / USB

- Convert ISO: **"isohybrid /path/to/iso"**
- Copy ISO on key (warning, **wipes everything** on key): **"dd if=/path/to/iso of=/dev/sd..."**

Boot parameters

System language and keymap layout

- **"locale"**: system language; example: **"locale=fr_FR.UTF-8"**
- **"keymap"**: keymap layout; example: **"keymap=fr"**; the first two caracters are used for Xorg keymap layout
- **"tz"**: timezone; example: **"tz=Europe/Paris"** (value must be a valid path from **"/usr/share/zoneinfo"**)
- **"hwc"**: hardware clock: **"UTC"** or **"localtime"**

Modules loading

- **"include=module1:module2:..."**: to load selected modules from **"/boot/optional"** directory (module names are the SquashFS file names)
- **"exclude=module1:module2:..."**: to specify the main modules (from **"/boot/modules"** directory) not to load; example: **"exclude=1-gui"**

Persistent home directory storage or system changes

- **"home=path_to_storage|NFS_resource|block_device"**: activates persistent home directory on USB key or over NFS (if not, it is stored in RAM);
  - the storage can be a directory if the USB stick is ext3 formatted; example: **"home=/home.dir"**;
  - a file (containing a Linux file system); append **"=size_in_MB"** to the file path to create the storage file (example: **"home=/home.fs=100"**);
  - a block device (**"/dev/sdb3"** for example);
  - or a NFS resource.
- **"changes=path_to_storage[=size_in_MB]|NFS_resource|block_device"**: activates persistent system changes on USB key or over NFS (if not, they are stored in RAM).

Misc

- **"runlevel=[1-5]"**: overrides default startup runlevel (cf **"/etc/inittab"**)
  on runlevel 5, the user with uid 1000 will be automatically logged in and X session started (if installed); if this user doesn't exists, root user is used instead
- **"copy2ram=yes"**: enables running the live system from memory
- **"useswap=yes"**: enables swap on detected swap partitions
- **"rootpw=*password*"**: defines a root password
- **"fastboot=yes"**: skip **"ldconfig"**, **"depmod"**, **"fc-cache"**, **"update-mime-database"**, **"gtk-update-icon-cache"**, **"update-gtk-immodules"**, **"update-gtk-immodules"** and **"update-pango-querymodules"**
- **"debug=seconds"** to have time to read the console messages if the initrd fails loading the live system

Live system installation

The live system can be installed into a hard disk partition; the result is the same as a clean installation.

Install live system

- usage: **"build-slackware-live.sh --install root_dir device [-auto]"**
  *the **"-auto"** option enables LiLo / eLiLo installation*
- example - from a running live system (typically):
  "build-slackware-live.sh --install /live/system /dev/sdx2 -auto"
- example - system cloning:
  "build-slackware-live.sh --install /mnt/system /dev/sdx2"

Live root over NFS setup

The live system can be booted over the network.

Prerequisites

- DHCP server (**"n/dhcp"**)
- TFTP server (**"n/inetd"**, **"n/tftp-hpa"**)
- NFS server (**"n/portmap"** + **"n/nfs-utils"**)

Enabling live root over NFS service

- usage: **"build-slackware-live.sh --share live_dir net_iface ip_range [modules_list/auto]"**
  ***"modules_list"**: network drivers to add to the initrd*
- example - after initialization and module creation:
  "build-slackware-live.sh --share /tmp/live eth0 1-253 forcedeth:tg3"
- example - from a running live system:
  "build-slackware-live.sh --share /live/media eth0 1-100 auto"
  *if **"auto"** is specified, currently loaded network drivers are added to initrd*
- disabling service: **"build-slackware-live.sh --unshare"**

Live system usage

Firefox cache and persistent home directory problem

When using Firefox with persistent home directory feature enabled on an USB flash disk, disk cache should be disabled to avoid continuous writings. This can be done by pointing your Web browser to **"about:config"** (or by editing **".mozilla/firefox/*PROFILE_NAME*/prefs.js"**) and set **"browser.cache.disk.enable"** property to **"false"**.