This page is being written and not yet complete

Get acquainted with Slint

Introduction

This document is intended to allow people who have already used another “Unix-like” system to get quickly acquainted with Slint. We assume that the reader:

- knows the main Unix commands and the associated concepts,
- knows how-to edit a text file (such as a configuration file)
- can read a simple shell script.

Installation, configuration, startup

Installation

The installation media can be obtained on Slint's repository (see above).

Slint provides a text installer, the document Slint-HOWTO presents the installation process.

The ISO images include a polyglot and accessible installers, which also completes the system configuration: choosing the language, the keyboard layout and creating ordinary users.

Configuration post-installation

The tasks of configuration and administration must be carried out as root. If you are logged in as a normal user, type `su` (to obtain root privileges) or `su -` (to become root). The `sudo command`, although available, is rarely used to administer Slackware.

- To create “ordinary” users, type `adduser`
- To change the font for the console type `setconsolefont`
- To change the console keymap edit /etc/rc.d/rc.keymap and make it executable
- To change the LANG edit /etc/profile.d/lang.sh and case occurring /etc/profile.d/lang.csh. Note: in Slackware /bin/sh is a symlink to /bin/bash.
- To modify the keyboard settings for X (in graphical mode), copy usr/share/X11/xorg.conf.d/90-keyboard-layout.conf to /etc/X11/xorg.conf.d, hen edit the copy. The file /etc/X11/xkb/rules/evdev.lst lists all known alues for XkbModel, XkbLayout, XkbVariant and XkbOptions under the respective headings `! model`, `! layout`, `! variant` and `! option`
- You can re-run the setup scripts (already used at the end of installation) using the command `pkgtool`, menu entry `Setup`.

Slint includes the main text editors such as pico, nano, elvis and vim and file managers mc and thunar and, of course, all common utilities and shells, useful to configure and administer the system.

In general, Slint does not prejudge how the system will be used. It is up to the administrator to customize it for its intended usage by editing the configuration files located in /etc or its subdirectories. It helps to read the comments in scripts for management of services located in /etc/rc.d as well as in the configuration files.

Startup

At first startup after installation your system will be in “console” mode, without a GUI. If you prefer to start in graphical mode, replace “id:3:initdefault:” with “id:4:initdefault:” (runlevel 4 instead of 3) in the file /etc/inittab

Otherwise, to switch from console to graphical mode, choose first your window manager or desktop by default through command `xwmconfig` either as root (for all) or as ordinary user (for that user). Then type `startx` to start this window manager or desktop from the console after system startup.

In Slint the run levels configured in /etc/inittab are:

- 0: shutdown
- 1: single user
- 3: multi user (default level)
- 4: idem 3, but with a sessions manager for X
- 7: reboot

At these levels correspond scripts in /etc/rc.d:

- rc.S runs at startup, it initializes the system, checks then mount the file systems.
- rc.M in multi user mode starts most services (daemons launched by scripts in /etc/rc.d if executable).
- rc.K goes into single user mode (run level 1 or S).
- rc.4 starts a session manager: gdm, kdm or xdm, others can be added editing that script.
- rc.0 shuts down the system (symlink to rc.6).
- rc.6 reboots or shuts down (if called as rc.0) the system cleanly.

Note. The administrator can add services scripts (daemons manager) in /etc/rc.d. Place the `start` of the demons in /etc/rc.d/rc.local and the `stop` in /etc/rc.d/rc.local_shutdown. /etc/rc.d/rc.local is run by /etc/rc.d/rc.M, /etc/rc.d/rc.local_shutdown by /etc/rc.d/rc.6

In addition Slackware includes a structure for managing services by run level in a `sysvinit` fashion for the software not included in the distribution (commercial ones, noticeably) that need it.

The management of services to be launched at startup can be done:

- manually making executable: (`chmod 755 <script name>`) or not: (`chmod 644 <script name>`) the corresponding script in /etc/rc/.d
- using the command `pkgtool` (choose `Setup` then `services` in the menu).

Note. As all administration tools, `pkgtool` should be run as root.

Management of software packages

Presentation of Slint software packages

Slint is distributed as a set of software packages containing applications and associated documents, and if necessary scripts executed during installation, noticeably for creating symbolic links or manage the configuration files and services' management scripts.

A Slint package consists in a file tree archived with `tar` and once compressed with `gzip`, nowadays with `xz` .

A shell script called “SlackBuild” compiles binaries to be shipped in the archive and installs them in the package's tree alongside other files. It usually ends with the execution of the `makepkg` command, which makes the archive from the package's file tree, including case occuring the appropriate installation scripts, cf. “man makepkg”.

The software packages are in the directories of installation media indicated above. Each of these directories contains following files:

- PACKAGES.TXT ⇒ Name, size, and description of each package.
- FILE_LIST ⇒ All files included in the directory.
- MANIFEST.bz2 ⇒ The characteristics of each file (to be) installed by each package (file compressed with `bzip2`)

For instance, the contents of the directory source/ap/tmux/ is:

```
.
|-- slack-desc          application's description
|-- tmux-1.8.tar.xz     source archive
`-- tmux.SlackBuild     should be (made) executable
```

This allows in particular to build a package with other compilation options and/or for a newer version by editing the SlackBuild, and case occuring by copying in the package source directory a different version of the software's source archive.

In our example to modernize `tmux`, just download the most recent (at time of writing) source archive tmux-1.9a.tar.gz from the tmux website, place it in source/ap/tmux/ and type as root from that directory 'VERSION=1.9a./tmux.SlackBuild' to build a/tmp/tmux-1.9a-i486-1.txz package that can replace the previous one with the command 'upgradepkg /tmp/tmux-1.9a-i486-1.txz'

The Slackware packages database

It consists of text files in the directory /var/log of installed system:

/var/log/packages/<package>: provides summary data about the package and lists installed files
/var/log/scripts/<package>: lists case occurring the commands run by installation scripts
/var/log/removed_packages/<package>: provide information about deleted package
/var/log/removed_scripts/<package>: lists the commands run by installation scripts of removed packages

As they are text files, you can read them with `less` or with a text editor. They are updated and used by package management programs. They allow for example to know the contents of a package, in which packages(s) is shipped an installed file or whether a file was modified or deleted since installation.

The files PACKAGES.TXT and FILE_LIST included in the package directories are also used for their management.

Installing, removing and updating software packages

These functions are provided respectively by the commands `installpkg`, `removepkg`, `upgradepkg`, cf. their man pages.

Caution: `ugradepkg` would be better named `replacepkg` as that command installs the specified package then removes the previously installed one, regardless of their respective versions.

The command `pkgtool`, menu driven, allows to install, remove, examine the content of packages and more generally administer the system: `pkgtool` is the “Swiss army knife” of Slackware. In addition to “man pkgtool”, get to know the tool browsing its menu.

The command `slackpkg`, build atop other Slackware package tools, provides in addition an access to a local or remote mirror of official Slackware packages. This allows e.g. to download and install a set of packages or keep the system up to date with a single command. Its operation is governed by the configuration files /etc/slackpkg/slackpkg.conf and /etc/slackpkg/mirrors, see “man slackpkg” and “man slackpkg.conf”.

The software package's database on Slackware's website

The Slackware Package Browser allows to make a search using following criteria:

- Package
- Label
- Description
- Content

in one or more packages directories, for all versions of Slackware from 8.1 up to “current”.
It also provides access to the files' tree of each of these versions.

Finally, some trusted third party offer packages already built. See especially those offered by Eric Hameleers aka Alien BOB, Robby Workman and Matteo Bernardini aka ponce, Slackware contributors.

The website http://www.slackware.org.uk/ also hosts packages or Slackbuilds such as those proposed by distributions derived from Slackware, as Salix maintained mainly by George Vlahavas, or desktops not included in Slackware, such as Mate proposed by Chess Griffin and Willy Sudiarto Raharjo, or provided by slacky.eu

Packages that can be installed on Slackware are also available elsewhere. Important warnings:

- Know who proposes these packages and preferably trust contributors of Slackware or at least of http://slackbuilds.org
- Carefully inspect the contents of packages before installation, including a check of the permissions of files and any changes to existing directories and files. Use `less`, `tar` and/or

  `explodepkg`, see “man explodepkg”.
- Never install a package whose sources and SlackBuild are not available.

s.

Build your own packages

This is the recommended method if no SlackBuild is available, to benefit of the Slackware package management tools and maintain a “clean” system.

The software `slacktrack`, included in Slackware, can help you, as “man makepkg”. Inspecting existing Slackbuilds and templates proposed at slackbuilds.org can inspire you. Browse SlackDocs with the key word “package” or “install” to learn more.

Maintainance of the system

To be informed of the availability of software packages that meet security breaches, email majordomo@slackware.com with the phrase “subscribe slackware-security” in the body.

For a stable version updates are few, because they are only intended to fill a security breach or to correct a major bug and not just to provide a more recent version of software included in the distribution. Users are expected to modernize their software themselves if they wish, see previous chapter for how to do that: Slackware is not a rolling releases, but publishes a succession of stable versions.

Updating or reinstalling a package can lead to reinstall a file (usually a configuration file) already installed. In this case if they don't match the system will install a new file next to the precedent with a .new extension and you get to decide what to do on a case by case basis: merge the two files, keep the former as is or replace it with the new one.

The practice of installing on a system a software package shipped in another version of Slackware is discouraged and often fails. Better try to rebuild a package for the installed version, running the SlackBuild in a copy of the source directory of that package, gathered from the other version. And of course, it is strongly unadvised to install packages intended for or included in another distribution, unless it be deemed 100% compatible with Slint's identical version.