A history of Slackware development

by Eric Hameleers
http://slackware.com/~alien/
alien@slackware.com

Presented at:
T-DOSE
Fontys University of Applied Science in Eindhoven
http://t-dose.org/

October 3–4, 2009

Overview of this presentation

A history of Slackware development

- Introduction
- A look back in time
  - History
  - Slackware philosophy
- Development
  - The team
  - Package management
  - Package format and support tools
- Installing additional software
  - 3rd party repositories
  - SlackBuild scripts
- Other resources

Overview of this presentation

A history of Slackware development

- Introduction
- A look back in time
  - History
  - Slackware philosophy
- Development
  - The team
  - Package management
  - Package format and support tools
- Installing additional software
  - 3rd party repositories
  - SlackBuild scripts
- Other resources

You will see how easy it is to use a Wii controller in Slackware ;-)

Introduction

Eric Hameleers

- Slackware user since 1996
- Slackware core team member since 2006

A look back in time

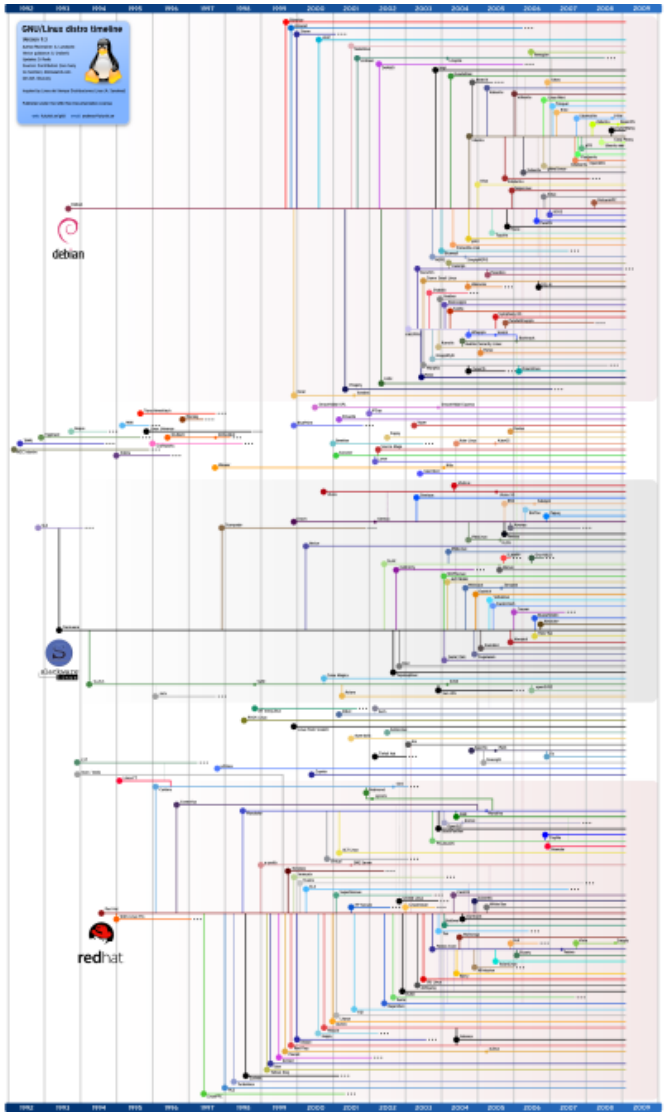
- Derived from SoftLandingSystems (SLS) — Slackware is probably the first "fork" distribution.
- Started as a series of fixes for various bugs in SLS that for whatever reason never went back into Peter MacDonald's SLS. Patrick made these available on his university's ftp server for others.
- This modified version of SLS moved forward while SLS development stalled, and by popular demand, it was released as "Slackware 1.0" on July 16th, 1993. In order to move away from SLS, the installer for the new "Slackware" was rewritten, using a new tool called "dialog" (which Patrick helped write). The dialog based installer made its debut in Slackware 1.1.0.

Slackware release timeline

Some highlights in Slackware release history

1.0	16/07/1993 (first release)
1.1.2	05/02/1994 (with dialog-based installer)
1.2.0.1	01/04/1994 (with Linux kernel 1.0)
3.3	11/07/1997 (earliest version still on most mirrors, used linux 2.0.30)
4.0.0	17/05/1999 (first release to ship with kde)
7.1	22/06/2000 (first release to ship with gnome)
10.1	02/02/2005 (gnome was dropped, 2.6 kernel appeared in /testing)
11.0	01/10/2006 (last with specialized 2.4 kernel boot disks, and a monolithic x.org)
12.1	02/05/2008 (http/ftp network installation and support for LVM/LUKS in the installer)
13.0	28/08/2009 (KDE4 replaces KDE3, 64-bit port)

Linux Distro Timeline - http://futurist.se/gldt/



Interesting graph of distro heritage

Slackware Philosophy

- Slackware is still very much a "traditional" linux distribution — this is expected due to its reputation as the most UNIX-like distribution of linux.
- Stability and ease of use are Slackware's primary goals
  - Attempt to ship unmodified upstream sources
  - Not bound to a predefined release schedule — it will be released when it's ready, but we try to do a new release at least once per year
  - Transparent configuration — well-commented configuration files
  - Each application is configured independently like the upstream developers intend — Slackware does not have a "global" configuration file or tool
  - No Slackware-specific "hidden" configuration — documentation provided by the upstream developers should be correct and complete

Slackware Philosophy (cont'd)

- Conservative development model — well-tested and functional software is not quickly replaced by newer, untested, and possibly less stable software without good reason

This does not mean that Slackware ships OLD software!

Slackware is generally just as "up to date" (if not **more** up to date) as any other distribution with the software that it includes

- Slackware is not intended to be a Windows clone or work-alike — there are some other linux distributions handling that quite well
- Slackware provides a system that is intuitive and easy to use for an experienced Unix administrator while also being relatively easy to learn for a new Unix user

Development

What is Slackware's development model?

- Commercial distribution, funded *only* by store sales. On the other hand, forever free to download.
- Non-open development
  - No bugzilla, no public code repository, no code contributors
  - No procedures to "become" a developer.
- "Benevolent Dictatorship" (although Pat would not agree).
  The final decision about what goes into Slackware remains with Patrick Volkerding. But... he listens to common sense.
- Not a one-man show!
  Slackware has a core team of contributors
  - Non-public testing of new, or updated, packages
  - Contributing changes to Slackware core scripts (bootup, network, installation, ...)
  - Proposing for new packages to be added
  - Providing a private discussion forum to decide on future steps

From past to future

Looking forward

- Self-imposed constraints (team size, design philosophy)
  Slackware will never cater for the masses. True "out of the box" experience would require a bigger team - but goes against the "vanilla" approach anyway
- Server vs. the desktop
  Excellent platform for software development, and running server services. It still makes a fine desktop thanks to the efforts that go into KDE, XOrg and others. But just as with Gnome's dismissal, KDE could be removed if the effort of integrating into Slackware becomes too big (slim chance, though).
- Hardware platforms
  After adding x86_64 to the list of supported platforms, which led to a re-write of almost every SlackBuild, there is good progress in re-synchronizing the ARM and S390 ports to baseline.

Core Team

Patrick Volkerding — volkerdi@
Eric Hameleers — alien@
Piter PUNK — piterpunk@
Robby Workman — rworkman@
Stuart Winter — mozes@
Mark Post — markkp@
Fred Emmott — fred@

Vincent Batts — vbatts@
Alan Hicks — alan@
Amritpal Bath — amrit@
Erik Jan Tromp — alphageek@
Karl Magnus Kolstø — karlmag@
Leopold Midha — netrixtardis@
John Jenkins — mrgoblin@

Other Contributors

There are several others who wish to remain anonymous

User Community

Of course, a **LOT** of the work, especially testing and bug reports, comes from the user community - thanks!

Slackware Package Management

Slackware Package Management

- Slackware **does** have a package manager.
  - `pkgtool(8)`, `installpkg(8)`, `upgradepkg(8)`, and `removepkg(8)` manage **packages** just fine.

Slackware Package Management

- Slackware **does** have a package manager.
  - `pkgtool(8)`, `installpkg(8)`, `upgradepkg(8)`, and `removepkg(8)` manage **packages** just fine.
- Slackware's native package management tools, however, do **NOT** attempt to manage package **dependencies** at all.
  - Packages are not split into *app-bin*, *app-lib*, *app-devel*, *app-doc*, and so on — every file that would normally be installed by `make install` will be present in Slackware's package.
  - Disk space is not as big of a concern today as it was in the past, so a full installation is generally recommended.
  - In a full installation, all dependencies needed by Slackware's native packages are provided.

Slackware Package Management (cont'd)

You might want to obtain information about Slackware packages that are already installed on your system.

Use `ls(1)` to list installed packages

```
# ls /var/log/packages
a2ps-4.14-i486-4
aaa_base-13.0-noarch-2
aaa_elflibs-12.34-i486-1
...
```

Use `cat(1)` to show the contents of an individual package

```
# cat /var/log/packages/a2ps-4.14-i486-4
PACKAGE NAME: a2ps-4.14-i486-4
COMPRESSED PACKAGE SIZE: 748 K
UNCOMPRESSED PACKAGE SIZE: 4650 K
...
usr/share/psutils/md68_0.ps
usr/share/psutils/md71_0.ps
```

How Slackware's Package Management Utilities Work

- Slackware packages are basically just plain compressed tar archives. They are extracted to the "/" directory unless an alternate root is specified.
- If the environment variable "`ROOT`" is set, then the package management utilities will act as if its value is the real "/".
- You will notice that all of the files in the example package are listed with relative paths — this is so that they will be placed in the correct location when the package is extracted.
- Once the files in the package are extracted, then `doinst.sh` (the postinstall script) is executed from the `$ROOT` directory (which is usually "/".

Sample Package Contents

FILELIST

```
./
etc/
etc/hejaz.conf.new
usr/
usr/bin/
usr/bin/makehejaz
usr/doc/
usr/doc/makehejaz-1.0/
usr/doc/makehejaz-1.0/README
usr/doc/makehejaz-1.0/COPYING
usr/include/
usr/include/hejaz.h
usr/lib
usr/lib/libhejaz.so.1.0
usr/man
usr/man/man1/
usr/man/man1/makehejaz.1.gz
install/
install/doinst.sh
install/slack-desc
```

makehejaz doinst.sh contents

cat install/doinst.sh

```
config() {
 NEW="$1"
 OLD="$(dirname $NEW)/$(basename $NEW .new)"
 # If there's no config file by that name, move it over:
 if [ !  -r $OLD ]; then
   mv $NEW $OLD
 elif [ "$(cat $OLD | md5sum)" = "$(cat $NEW | md5sum)" ]; then
 # toss the redundant copy
   rm $NEW
 fi
# Otherwise, we leave the new copy for the admin to consider...
}

config etc/hejaz.conf.new

( cd usr/lib ; rm -rf libhejaz.so.1 )
( cd usr/lib ; ln -s libhejaz.so.1.0 libhejaz.so.1 )
( cd usr/lib ; rm -rf libhejaz.so )
( cd usr/lib ; ln -s libhejaz.so.1 libhejaz.so )
```

Config File Handling in the Package

Note that the configuration file "`etc/hejaz.conf`" was actually installed with a "`.new`" extension. A proper Slackware package should almost always do this with config files, init scripts, and other such files that might be customized by the system administrator.

- This prevents package upgrades from overwriting config files already installed on the system.
- In the sample package listed above, the `config()` function in the `doinst.sh` file checks to see if the *hejaz.conf* file is present at `$ROOT/etc/hejaz.conf`.
  - If it does not exist, the *hejaz.conf.new* file installed by the package is moved over to *hejaz.conf*.
  - If it does exist, then the md5sum of it is compared to the new one.
    - If the md5sums match, then the two files are identical, so the new one is deleted.
    - If the md5sums do not match, the new file is left (with the `.new` extension) for the system administrator to check.

Symlink Handling in the Package

You may have noticed that the sample package contained a single library file — `libhejaz.so.1.0` — but apparently no files or symlinks named `libhejaz.so.1` or `libhejaz.so` pointing to it.

This is because Slackware's `makepkg(8)` utility removes the symlinks when creating the package — it then records them in the *doinst.sh* file so that they are created when the package is installed.

This does NOT mean that you should manually add symlink creation to a `doinst.sh` file!

Create any needed symlinks so that they actually exist in your temporary package installation directory (such as `$DESTDIR`), and then let `makepkg(8)` handle this — it's less error-prone than we are...

Installing Additional Software in Slackware

Slackware's official package set is adequate for many users, but there are quite a few additional pieces of software that are needed in many cases. The traditional "`configure && make && make install`" works, but it comes with potential problems...

- Possible conflicts with packaged software
- Difficult to track what is installed
- Possible problems with upgrading and/or removing
- Experienced users can manage these just fine, but others will eventually have problems in most cases

There are a few well-known third party package repositories...

- Many users do not want to install packages from third party sources at all.
- For those that do install packages from third party sources, it is important to find **trusted** sources.

Installing Additional Software (cont'd)

Trusted sources for packages — where are they and how can you know if they are trustworthy?

- Is the packager someone with a good reputation in the community?
- Check the package contents
  - Are the files in the correct locations?
  - Are config files properly installed?
  - Are man pages compressed?
  - Does the package include documentation (if applicable)?
  - Are the packages built on a "clean" Slackware installation?
  - Are all package dependencies documented?
- Does the packager provide complete sources and documentation of how the package was built (such as a build script)?

Sources are usually required to be hosted per the software's license!

In our opinion, packagers who do not provide sources and build methods should generally be avoided.

Third Party Repositories

- A few of the Slackware team members provide packages (and sources) for various applications and libraries
  - Eric Hameleers (alienBOB) — `http://slackware.com/~alien/`
  - Robby Workman (rworkman) — `http://rlworkman.net/pkgs/`
  - Erik Jan Tromp (alphageek) — `http://alphageek.dyndns.org/`
  - Piter PUNK — `http://piterpunk.unitednerds.org/`

This is not intended to be an all-inclusive list!

`http://slackfind.net/en/` is a repository search engine

Package quality may vary — some are known for not providing sources, not providing build instructions, and/or not building on clean installations (which causes undocumented dependencies on other packages).
Therefore, you can choose to build your own packages using quality-checked SlackBuild scripts obtained from `http://slackbuilds.org`

Using SlackBuild Scripts

Eventually, you will need some application for which either:

- no package exists, or
- you do not want to install one of the available packages

When that happens, you may be able to find a SlackBuild script for the application instead.

A SlackBuild script is essentially just a small shell script

- extracts the source code tarball
- prepares the source code for compilation (sets compile options)
- compiles the object code
- installs it to a temporary "staging" directory
- performs other needed operations (install documentation, compress man pages, add package description, etcetera)
- uses `makepkg(8)` to make a Slackware package of the application

Wrapping it up

This concludes our tour of Slackware Linux!

Other resources

- Interview with Patrick Volkerding in LinuxJournal (1994)
  `http://www.linuxjournal.com/article/2750`
- Audio interview with Patrick by Linux Today (1999)
  `http://www.linuxtoday.com/developer/1999120500405NWSM`
- Phone interview with Patrick in The Linux Link Tech Show (2006)
  `http://tllts.org/dl.php?episode=164`
- Complete overview of Slackware releases
  `http://www.nielshorn.net/slackware/slack_versions.php`
- Unofficial history of Slackware releases
  `http://www.slackdown.co.uk/history.html`

With thanks to Robby Workman (who contributed large parts of the presentation which we ran jointly at Slackware Show 2008), and Patrick Volkerding and the team for their input and feedback.

About the author

I am a Linux user since the 0.99 kernel releases and ran into Slackware Linux around 1996. The first need to make modifications to Slackware occurred when I joined IBM and was doomed to use Token Ring for which there was no support in the network scripts :-)
I am now a member of the core Slackware development team. I also maintain a repository of non-official Slackware packages and scripts, I am one of the admins of the SlackBuilds.org website, and sometimes like to write articles for my own Wiki. I was born in the Netherlands and still live there, with wife, son and several small animals.

Feedback and suggestions are welcome
`alien@slackware.com`

A copy of this presentation can be found at:
`http://www.slackware.com/~alien/tdose2009/`

Questions

Any Questions ?