Succeeding on the Bleeding Edge

Allan McRae

allan@archlinux.org

Overview

- Arch Linux
- Development process
- Involvement of the community
- Future plans

About Me...

- Not a “computer person” by day...
- Arch Linux developer for 5 years
- Responsible for the GNU Toolchain and related packages
- One of the primary Pacman package manager developers

Arch Linux – Overview

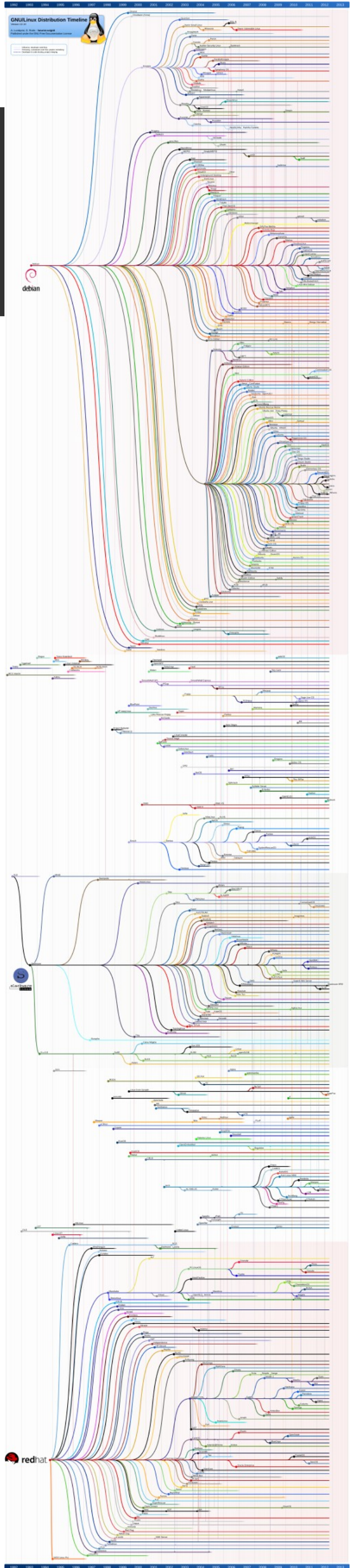
- **From the website:**

“Arch Linux is a versatile, and simple distribution designed to fit the needs of the competent Linux user.”

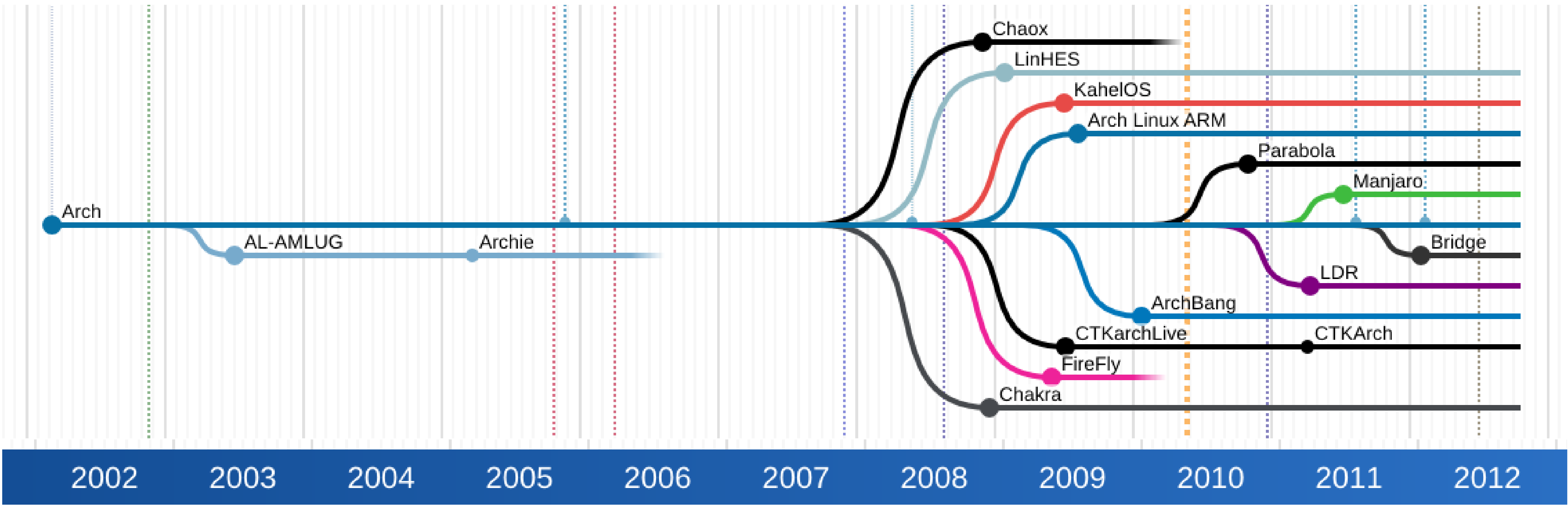
“A lightweight and flexible Linux distribution that tries to Keep It Simple.”

Linux Distributions

- Distrowatch tracks 319 active Linux distributions (with 340 more on the waiting list)
- Many are variants of another distribution
- GNU/Linux Distribution Timeline (http://futurist.se/gldt/)
  - 480 distributions
  - Major clusters starting at
  - Debian, Slackware, Red Hat



Linux Distributions – Arch Based



What Separates Linux Distros?

- Target audience
- CPU architecture
- Software selection
- Software management

What Separates Linux Distros?

- **Target audience**
  - **Beginner or <u>experienced</u> users?**
  - **<u>Desktop</u> or server usage?**
  - **Live distro?**
  - **Specialist purpose (rescue, audio, ...)**

- **CPU architecture**
  - **<u>i686</u>, <u>x86-64</u>, ARM, PPC, SPARC, ...**

What Separates Linux Distros?

- Software selection
  - Desktop environment
    (GNOME, KDE, XFCE, LXDM, …)
  - Specialist software
    (audio, scientific, gaming, …)
  - Proprietary software availability
  - Default filesystem
  - …

What Separates Linux Distros?

- Software management
- From a distributions point of view:
  - How often are packages updated?
  - When are new releases made?
  - How long are releases supported for?
- From a users point of view:
  - How do I find and install software?
  - How do I keep my system up-to-date?

The Standard Release Model

- Most Linux distributions make a release then only provide security updates for software until their next major release
- Requires a major update, typically every six months
- Can be easier to re-install...
- Requires waiting for the latest software (or installing from unsupported sources)

The Rolling Release Model

(There are six kinds of rolling release according to Wikipedia...)

- Software is continuously updated as newer versions are released
- No major distribution releases are made, as users continuously are upgrading to the “new version”
- Can be less stable...

Rolling Release and Arch Linux

- Arch Linux is on the extreme of rolling release systems
- Packages are typically updated within a few days of release (sometimes within minutes!)
- Only keep latest version of software in our repositories

Software Management

- How a user deals with software installation and updates is one of the most important aspects of a Linux distribution

- There are two main package management systems in Linux:
  - RPM – use by Red Hat, openSUSE, ...
    - (rpm → yum → ...)
  - deb – used by Debian and its derivatives
    (dpkg → apt → synaptics)
  - many others...

Arch Linux Package Manager

- Uses the “pacman” package manager
- Combines a simple binary package format with easy to use build system
- Fast! - according to Linux Format it beats the competition by a wide margin
- Does everything you expect from a package manager (update system, resolve dependencies, …)

Package Creation

- Very simple scripts required to create a package
- If you can build the software manually, then you can create a package for it
- Tool provided to build packages called “makepkg”
- Build script is placed in a file called a PKGBUILD

Package Creation

- **Start with how you would normally install a program:**

```
$ tar -xf <pkgname>-<pkgver>
$ cd <pkgname>-<pkgver>
$ ./configure
$ make
$ sudo make install
```

Package Creation

- **Separate out the parts run as a user and root into separate functions:**

```
$ tar -xf <pkgname>-<pkgver>
$ cd <pkgname>-<pkgver>

build() {
  ./configure
  make
}

package() {
  make install
}
```

Package Creation

- **makepkg will automatically handle source extraction into "$srcdir"**

```
build() {
  cd $srcdir/<pkgname>-<pkgver>
  ./configure
  make
}

package() {
  cd $srcdir/<pkgname>-<pkgver>
  make install
}
```

Package Creation

- **Files need to be installed in “$pkgdir”, which is compressed to make the package:**

```
build() {
  cd $srcdir-<pkgname>-<pkgver>
  ./configure –prefix=/usr
  make
}

package() {
  cd $srcdir-<pkgname>-<pkgver>
  make DESTDIR=$pkgdir install
}
```

Package Creation

- **Add some information about the package at the top of the file:**

```
pkgname=foo
pkgver=3.0
pkgrel=1
pkgdesc="Example software"
arch=('i686' 'x86_64')
url="http://foo.example.com"
license=('GPL')
depends=('glibc')
source=(http://$pkgname-$pkgver.tar.gz)
md5sums=('d41d8cd98f00b204e9800f8427e')
```

Package Creation

- makepkg automates many common packaging tasks:
  - Stripping debugging symbols from binaries
  - Compressing man and info pages
  - Setting compiler/linker options (CFLAGS, LDFLAGS, MAKEFLAGS)
  - Removing common unwanted files (libtool, infodir, ...)

Package Creation

- **A single file is placed in the $pkgdir directory with all the needed package annotation**
- **Then a (compressed) tar archive of the $pkgdir directory is created**
- **DONE!**

Package Creation

- PKGBUILDs are written in Bash
  - Easy to create
  - Easy to interpret
- Makes contributing PKGBUILDs for your favourite software simple!
- Working on simplifying PKGBUILDs further without losing simplicity
  - VCS source URLs
  - Common packaging functions?

Arch Linux Development

- “Community based” distribution
  (No-one gets paid)

- Relatively small team
  - 33 Developers (many inactive...)
  - 37 Trusted Users
  - + Forum Moderators, IRC Ops, Wiki Maintainers, Bug Wranglers, ...

- Lots of involvement from users

Developers

- Maintain the core of the distribution
- Make global decisions on that effect the entire distribution
- Maintain packages in the two primary repositories – [core] and [extra]

Developers

- [core] - ~200 packages
  - Everything critical to boot-up and software packaging
  - All packages go through a testing and sign-off procedure

- [extra] - ~2,800 packages
  - Widely used (>5%) software
  - Desktop environments, multimedia, programming language interpreters, office, ...

Trusted Users

- “Independently” governed group
- Provide popular software(>1% usage) in the [community] repository to supplement the [extra] repository
- ~2,900 packages

Becoming a Developer

- Actively contribute to the community
  - Provide PKGBUILDs for unpackaged sofware
  - Fixing bugs
  - Contributing code to our projects
  - ...
- Apply to become a Trusted User
  - Sponsoring and voting process...
- Be invited onto the Developer team

How Is Development Co-ordinated?

- **Mostly... it is not...**
- **Developers typically maintain a set of packages**
- **Within that set of packages they have complete control**
- **Give other developers a "heads-up" if changes are going to have wider consequences to the distribution**

My Packages

Toolchain

gcc
glibc
binutils

Autotools

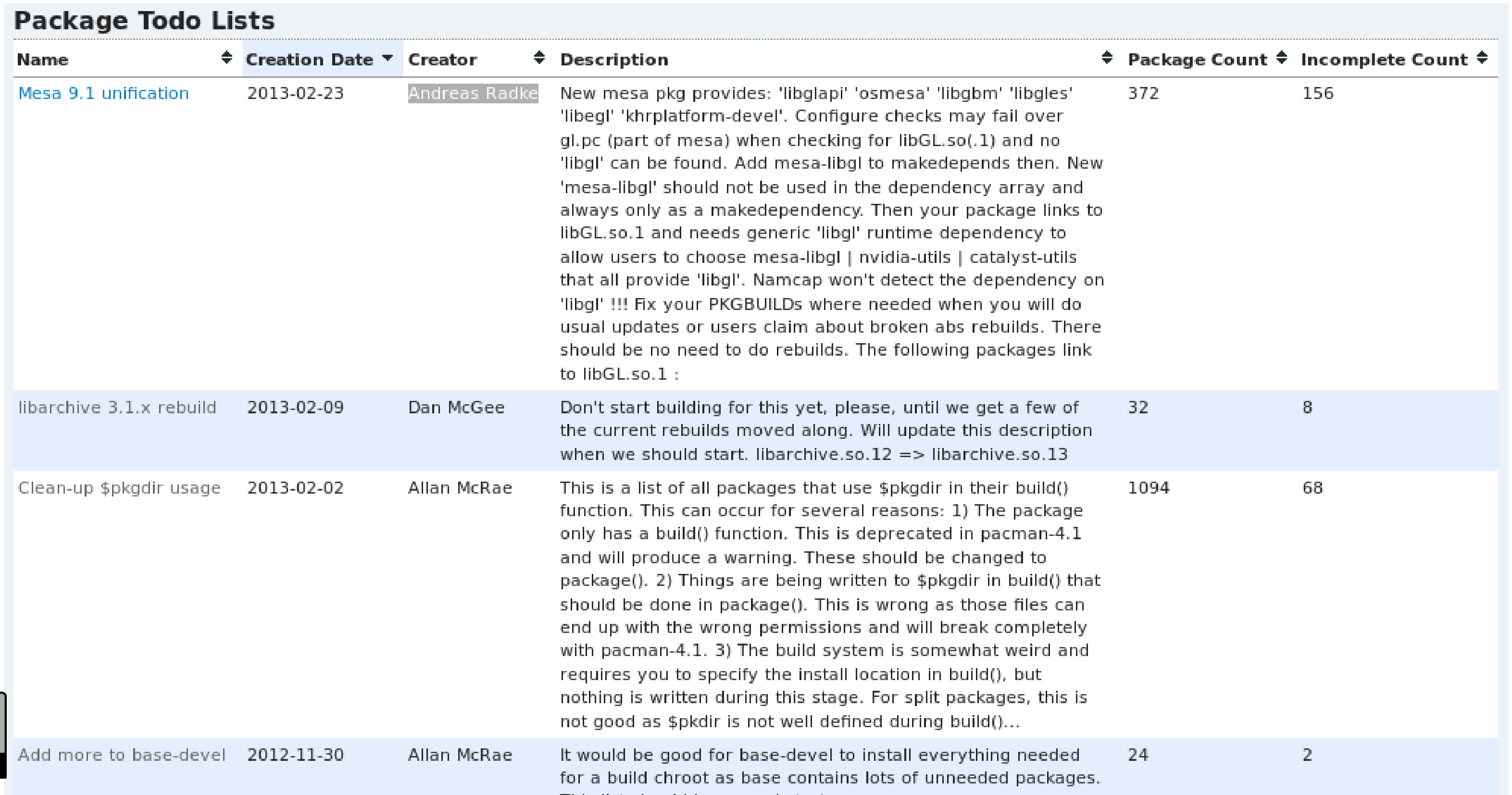
autoconf
automake
m4
libtool
pkg-config

Core Utilities

bash
coreutils
grep
make
sed
patch
gawk
tar
texinfo
...

Co-ordination Between Developers

- **We have a TODO list system for when a package update is going to require other developers adjusting their package(s)**



Package Todo Lists

Name	Creation Date	Creator	Description	Package Count	Incomplete Count
Mesa 9.1 unification	2013-02-23	Andreas Radke	New mesa pkg provides: 'libglapi' 'osmesa' 'libgbm' 'libgles' 'libegl' 'khrplatform-devel'. Configure checks may fail over gl.pc (part of mesa) when checking for libGL.so(.1) and no 'libgl' can be found. Add mesa-libgl to makedepends then. New 'mesa-libgl' should not be used in the dependency array and always only as a makedependency. Then your package links to libGL.so.1 and needs generic 'libgl' runtime dependency to allow users to choose mesa-libgl | nvidia-utils | catalyst-utils that all provide 'libgl'. Namcap won't detect the dependency on 'libgl' !!! Fix your PKGBUILDs where needed when you will do usual updates or users claim about broken abs rebuilds. There should be no need to do rebuilds. The following packages link to libGL.so.1 :	372	156
libarchive 3.1.x rebuild	2013-02-09	Dan McGee	Don't start building for this yet, please, until we get a few of the current rebuilds moved along. Will update this description when we should start. libarchive.so.12 => libarchive.so.13	32	8
Clean-up $pkgdir usage	2013-02-02	Allan McRae	This is a list of all packages that use $pkgdir in their build() function. This can occur for several reasons: 1) The package only has a build() function. This is deprecated in pacman-4.1 and will produce a warning. These should be changed to package(). 2) Things are being written to $pkgdir in build() that should be done in package(). This is wrong as those files can end up with the wrong permissions and will break completely with pacman-4.1. 3) The build system is somewhat weird and requires you to specify the install location in build(), but nothing is written during this stage. For split packages, this is not good as $pkdir is not well defined during build()...	1094	68
Add more to base-devel	2012-11-30	Allan McRae	It would be good for base-devel to install everything needed for a build chroot as base contains lots of unneeded packages.	24	2

Decision Making Process

- A discussion is started on the mailing lists for major changes that have effects beyond the developers set of packages
- Focus on technical reasons of why the proposed change is better
- No formal voting – decisions are made by lack of objection to a proposal

Example - systemd

- systemd is a Linux init system
- First process to get started during boot-up
- Starts all other processes
- Benefits:
  - Parallel start-up
  - Service start-up determined by simple configuration file
  - Service dependency management
  - Common configuration mechanism

Example - systemd

- First packages for system were placed in AUR in 2010
- Lots of work was required to make it work with Arch

> “I'm highly dubious that Arch's kernel will ever natively support systemd, but I'm willing to give that a try as well once 2.6.36 hits.”

Example - systemd

- Over the next two years...
- Moving to a standardised way of configuring aspects a system was seen as an advantage
- Changes were made to the Arch Linux init system to use these configuration files
- Started using systemd tools to do the configuration

Example - systemd

- Eventually...
    - systemd was considered stable
    - Bugs in the old Arch init system were being fixed using more and more systemd tools
    - systemd service files began to be supplied by upstream projects
    - Decision was made to switch init systems

Example - systemd

- This decision caused A LOT of controversy...
- The old system was seen as more simple:
  - Shell script – easier to debug?
  - Single configuration file
- However, “keeping it simple” as used by Arch has a different meaning:
  - Minimise Arch specific changes to packages

How Does It All Fit Together?

- Software developers write code that is supposed to work...
- By minimising Arch Linux specific changes to software, we ensure software fits together as its developer intended

Vanilla Packages

- Means packaging the software as the upstream developer intended
- Minimise patching – preferably only to fix build issues
- Result in any bug we find is (probably) not distribution specific
- Allows us to work more closely with software developers to fix bugs

Working With Software Developers

- All bug fix patches in Arch must be approved by the software developer
- That means that the Arch developers and community have become regular code contributors
- Many Arch developers also have commit access to upstream projects

Working With Software Developers

- **Increasingly common choice for software developers...**

archlinux
A simple, lightweight linux distribution.
Arch Linux
Boot Arch Linux (x86_64)
Boot Arch Linux (i686)
Boot existing OS
Run Memtest86+ (RAM test)
Press [Tab] to edit options

Community Involvement

- Users are strongly encouraged to contribute toward Arch Linux in may ways:
  - Help on the forums / IRC / mailing lists
  - Contribute PKGBUILDs
  - Documentation on the wiki
  - Provide specialist package repositories
  - ...

AUR – Arch User Repository

- Collection of user submitted PKGBUILDs that supplement software available from the official repositories
  - >40,000 packages
  - 170 new packages in last 7 days
  - 820 updated in last 7 days
  - ~16,000 updated in the last year...

- Some software represented multiple times
  - Developmental versions
  - Specific configure options

AUR – Arch User Repository

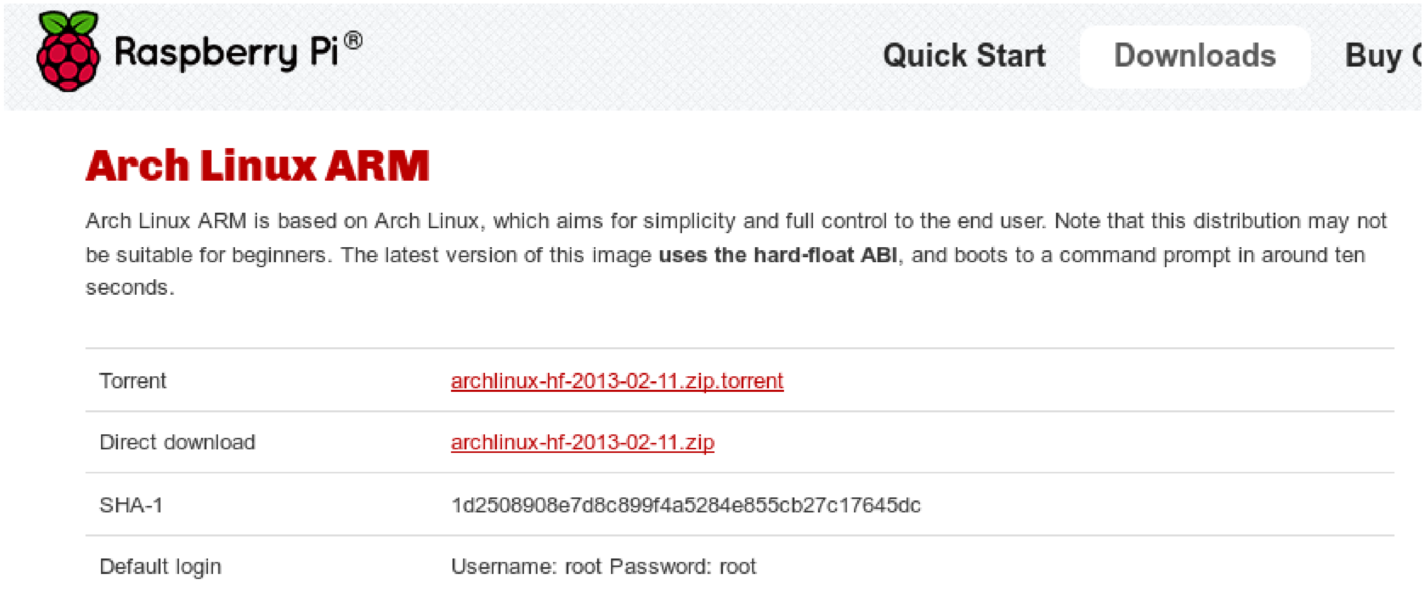
- Anyone can submit packages
- Entirely community supported and reviewed
- Completely unsupported officially
  (use at your own risk...)
- Surprisingly high quality
- Many tools that allow installing from the AUR as simply as installing from official repositories

Arch Linux Wiki

- Rapidly becoming one of the premier sources of Linux information
- Vanilla packages mean the information provided probably works on other distributions

Arch Linux ARM

- **Non-official spin-off for the ARM architecture**
- **One of the distros recommended for the Raspberry Pi**



Arch Linux ARM

Arch Linux ARM is based on Arch Linux, which aims for simplicity and full control to the end user. Note that this distribution may not be suitable for beginners. The latest version of this image **uses the hard-float ABI**, and boots to a command prompt in around ten seconds.

Torrent	archlinux-hf-2013-02-11.zip.torrent
Direct download	archlinux-hf-2013-02-11.zip
SHA-1	1d2508908e7d8c899f4a5284e855cb27c17645dc
Default login	Username: root Password: root

Future Directions for Arch Linux

- Majority response...

“Keep updating packages”

- Add more focus on a particular areas
- Add more architectures
- Simplify the system further

Keeping Packages Updated

- One of Arch Linux's greatest contributions to the Linux community
- Arch gets packages in their stable repositories before some major distributions get it in their developmental versions
- The Arch community will identify bugs early and report the issue to the software developers
- Fixes benefit all Linux distributions

Add More Architectures

- Currently we support x86 in 32bit (i686) and 64bit (x86-64) varieties
- There are community projects supporting other architectures
  - ARM (v5, v6, v7)
  - PPC
  - ...
- Would be good to provide a way for these ports to become official (like x86-64 did)

Simplifying the Filesystem

- **Usual filesystem layout has a lot of redundancies**

```
/boot
/bin
/etc
/home
/lib
/sbin
/usr
    /bin
    /lib
    /sbin
```

Simplifying the Filesystem

- **Libraries**

```
/boot
/bin
/etc
/home
/lib            (essential libraries)
/sbin
/usr
    /bin
    /lib        (rest of libraries)
    /sbin
```

Simplifying the Filesystem

- **Keep all libraries in one place**

```
/boot
/bin
/etc
/home
/lib  -> /usr/lib
/sbin
/usr
    /bin
    /lib
    /sbin
```

Simplifying the Filesystem

- **Executables – distinction between directories is vague...**

```
/boot
/bin              (essential user commands)
/etc
/home
/lib  -> /usr/lib
/sbin             (system commands)
/usr
    /bin          (most commands)
    /lib
    /sbin         (non-essential system)
```

Simplifying the Filesystem

- **Keep all libraries in one place**

```
/boot
/bin  -> /usr/bin
/etc
/home
/lib  -> /usr/lib
/sbin -> /usr/bin
/usr
    /bin
    /lib
    /sbin -> bin
```

Simplifying the Filesystem

- /etc directory holds all configuration files
- Beginning to have these placed in /usr/lib/<pkgname> with files in /etc overriding the default settings
- Would be very helpful for a rolling release system
- Requires substantial work with upstream projects to achieve...

Simplifying the Packaging System

- Many packaging task are overly repetitive...
- Examples
  - Many packages use simple “cmmi”
  - all Perl module packages look the same
  - Any time a font is installed, the font cache needs updated
  - Updated info packages need added to the info index
- Want to remove the repitition without adding complexity to packaging system

Automating More Packaging

- Task like rebuilds for library soname changes are typically trivial
- Would save a lot of time if we could automate (most of) this
- Most packages do not require architecture specific changes – build for one and automate the rest
- Would allow us to focus more on improving other areas of the distribution

Thanks

- The SINFO organisers for flying me over to talk about Arch Linux
- To every who responded to my request for information about what they planned to do with Arch Linux (even if I did not use much of it...)
- The Arch Linux community for everything that they contribute!

License