

Working with Ubuntu Linux

**Track 2 Workshop
June 2010
Pago Pago, American Samoa**



Assumptions

You are comfortable with the following:

- Core Linux concepts
 - Shells
 - Permissions
 - Graphical user interface (GUI) vs. command line interface (CLI)
- Working on the Linux command line
- Editing files in Linux (using vi, nano or other text editors)
- Basics of networking

Is this correct?



Goal

Present the “Ubuntu philosophy” and differences from other Linux distributions.

Including:

- Naming conventions
- Release conventions (Server, Desktop and LTS)
- Other flavors
- The Debian way
- Packaging system (how software is installed)
- Meta-packages
- Keeping up-to-date
- Stopping and starting services

Ubuntu Timeline

Version	Code name	Release date
4.10	Warty Warthog	2004-10-20
5.04	Hoary Hedgehog	2005-04-08
5.10	Breezy Badger	2005-10-13
6.06 LTS	Dapper Drake	2006-06-01
6.10	Edgy Eft	2006-10-26
7.04	Feisty Fawn	2007-04-19
7.10	Gutsy Gibbon	2007-10-18
8.04 LTS	Hardy Heron	2008-04-24
8.10	Intrepid Ibex	2008-10-30
9.04	Jaunty Jackalope	2009-04-23
9.10	Karmic Koala	2009-10-29
10.04 LTS	Lucid Lynx	2010-04-29
10.10	Maverick Meerkat	2010-10-10



The Debian Way

Ubuntu is built from Debian repositories and uses the Debian package management system.

- Debian is a very cautious and strict Linux distribution:
 - Minimal new versions
 - Extremely well-tested
 - No closed source software
 - Beta version of Debian as stable as release quality for most Linux distributions.
 - New versions are not released until they are ready.
 - Latest versions of software often not available in main branch as they are not considered stable or safe enough.
 - There are pluses and minuses to this approach.



The Ubuntu Take on the Debian Way

Potentially heretical slide ☺...

- Use the Debian software repository concept to classify software.
- Use the Debian package management system.
- Be more open – Ubuntu allows closed source software and drivers.
- Ubuntu pushes releases out fast, but supports releases for 2 to 5 years (Unlike Fedora Core's 18 months).
- Ubuntu aiming at both the desktop and server markets.
- The “Ubuntu Project” is supported by Mark Shuttleworth.
- Make maintaining a current system very easy to completely automatic (much like Windows).
- Support latest releases of major Open Source software projects (Firefox, Thunderbird, Gnome, OpenOffice, Xorg). Debian does not do this – much more conservative.



‘Default’ Partition Scheme

During an Ubuntu installation you can choose this option. It creates the following:

- Root partition (“/”)
 - Contains everything not in another partition
 - /bin, /sbin, /usr etc.
 - User home directories under /home
- A *swap partition* for virtual memory
- /boot for kernel boot files

What's Unique to Ubuntu

Software management

Command Line

- `dpkg`
 - `dpkg --get-selections`, `dpkg-reconfigure`, `dpkg-query`
- `apt`
 - `apt-cache`, `apt-cache policy`, `apt-cache search` `apt-get`,
`apt-get install`, `apt-get remove`, `apt-get purge`, `apt-get clean`
 - meta-packages (`build-essentials`, `ubuntu-desktop`)
- repositories – Controlled by `/etc/apt/sources.list`
- `aptitude`
 - `aptitude search`, `aptitude clean`, `aptitude remove`, `aptitude purge`

Graphical

- synaptic
- Ubuntu App Centre



Using apt

After initial install general cycle is:

1. `apt-get update`
 2. `apt-get upgrade`
- Repeat 1. If new packages, repeat 2.
 - Reboot only if new kernel image is installed.
 - Services are restarted if updated.
 - During install you can tell Ubuntu to automate this process.
 - Desktop users generally use *synaptic* or *Ubuntu App Centre* to do this.



What's Different cont.

Startup scripts

In /etc/init.d/ (System V)

Upon install services run!

Scripts are executed based on “K” and “S” links in the directories (we will take a look at this now):

```
/etc/rc0.d, /etc/rc1.d, /etc/rc2.d, /etc/rc3.d,  
/etc/rc4.d, /etc/rc5.d, /etc/rc6.d
```

Controlling services

- update-rc.d (default method)
 - sysvconfig
 - rcconf
 - rc-config



What's Different cont.

Make and GCC

- Not installed by default. Why?
- 30,000'ish packages
- Install from source is “not clean” in the Ubuntu world.
- To install:

```
apt-get install build-essential
```

What's Different cont.

The use of the *root* account is discouraged and the *sudo* program should be used to access root privileges from your own account instead.

You can do *apt-get dist-upgrade* to move between major and minor releases.

Package sources in */etc/apt/sources.list* (how you install from cd/dvd or the network).



How to Admin Your System

After you install Ubuntu you can...

- Execute system commands using *sudo* and the user account you created during install.

After you install Ubuntu you *cannot*:

- Log in as the *root* user.
- Become the *root* user using “*su -*”

You can get around this by doing:

- *sudo bash* [Opens a root shell in bash]
- *passwd* [Set a root password]

Should you do this?

Security hole!

- Ubuntu allows *root* user access via SSH by default. Setting the *root* user password opens this hole up.



Important Reads

man apt-get
man aptitude
man sources.list

Some people like aptitude, partly for the full-screen interface

Meta Packages

Annoying to new users

Provide all packages for subsystems

Initial documentation

<https://help.ubuntu.com/community/MetaPackages>

Examples include:

build-essential (libc, g++, gcc, make)

ubuntu-desktop (Xorg, gnome)

xserver-xorg-video-intel

Installing a minimal Gnome desktop

```
apt-get install --no-install-recommends ubuntu-desktop
```



There's More

But, hopefully enough to get us started...

Some Resources

www.ubuntu.com

ubuntuforums.org

www.debian.org

ubuntuguide.org

<http://en.wikipedia.org/wiki/Debian>

[http://en.wikipedia.org/wiki/Ubuntu_\(Linux_distribution\)](http://en.wikipedia.org/wiki/Ubuntu_(Linux_distribution))

GIYF (Google Is Your Friend)



Packages & Exercises

We'll reinforce some of these concepts using exercises...

