Linux System Administration

Getting started with Linux

Day 1: Modules

1. Linux overview
2. Command Line Interface or the “CLI”
3. Permissions
4. Editors
5. Ubuntu Linux and more commands

Module 5: System Admin Ubuntu

Goal

- Core commands to admin a system
- Understanding Ubuntu-specific methods
  - 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
  - Additional system administration commands

Ubuntu Timeline

Version	Code name	Release date	Supported until	
			Desktops	Servers
4.10	Warty Warthog	20 October 2004	30 April 2006	
5.04	Hoary Hedgehog	8 April 2005	31 October 2006	
5.10	Breezy Badger	13 October 2005	13 April 2007	
6.06 LTS	Dapper Drake	1 June 2006	14 July 2009	1 June 2011
6.10	Edgy Eft	26 October 2006	25 April 2008	
7.04	Feisty Fawn	19 April 2007	19 October 2008	
7.10	Gutsy Gibbon	18 October 2007	18 April 2009	
8.04 LTS	Hardy Heron	24 April 2008	12 May 2011	April 2013
8.10	Intrepid Ibex	30 October 2008	30 April 2010	
9.04	Jaunty Jackalope	23 April 2009	23 October 2010	
9.10	Karmic Koala	29 October 2009	30 April 2011	
10.04 LTS	Lucid Lynx	29 April 2010	April 2013	April 2015
10.10	Maverick Meerkat	10 October 2010	April 2012	
11.04	Natty Narwhal	28 April 2011	October 2012	
11.10	Oneiric Ocelot	13 October 2011	April 2013	
12.04 LTS	Precise Pangolin	26 April 2012[134]	April 2017[130]	

Colour	Meaning
Red	Release no longer supported
Green	Release still supported
Blue	Future release

The Debian Way

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

“Debian is a cautious and strict Linux distribution”

- Long release cycle
- Extremely well-tested
- No closed source software
- Beta version of Debian quite stable
- New versions very thoroughly tested
- 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.

Ubuntu View of the *Debian Way*

Potentially heretical slide ☺…

- Debian software repository concept to classify software.
- Use the Debian package management system.
- Ubuntu allows closed source software and drivers.
- Fast release cycle, long support (2 to 5 years)
- Ubuntu has both desktop and server versions
- “Ubuntu Project” is supported by Mark Shuttleworth.
- Keep systems current automatically (optional)
- Support latest releases of major Open Source software projects (Firefox, Thunderbird, Gnome, OpenOffice, Xorg). Debian is more conservative.

Debian/Ubuntu Unique Items

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`

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.

Services

Startup scripts

In /etc/init.d/ (System V)
In /etc/init/ (Ubuntu 12.04 LTS and Upstart)

NOTE! Upon install services run!

Controlling services

- `update-rc.d` (default method)
- Stop/Start/Restart/Reload/Status Services

```
# service <Service> <Action>
```

or, “old school”

```
# /etc/init.d/<service> <action>
```

Runlevels

As Linux boots it executes service startup using links.

Based on your “runlevel” determines what services will start.

Traditional levels are used like this:

- runlevel 1: single user mode (emergency mode)
- runlevel 2: multi-user mode (No Desktop)
- runlevel 5: multi-user mode (Desktop)

With Ubuntu We *Actually* Do…

What happens at each runlevel?

- init 1 ➔ Links in /etc/rc1.d are executed.
  Login as root user only.
  Minimal file system access.
- **init 2-5** ➔ Links in /etc/rc5.d are executed.
  Gui is started if installed.
  Day-to-day working state.
  - Ubuntu runs at “runlevel 2”
  - Other Linux, with Desktop, run at “runlevel 5”
  - This is largely semantics

Special Runlevels

Runlevel 0: Halt the system

Runlevel 6: Reboot the system

Runlevel 1: Single user mode. No network. No services. System Recovery.

You must be at the machine console or have Out-of-Band (OoB) access to your machine to use Runlevel 1.

Packages vs. Source

Make and GCC

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

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

Root account Access

- Use of the *root* account is discouraged.
- *sudo* is used to access root privileges from general user account instead.
- You can get around this very easily.

Should you run as root?

Your decision.

Accessing *root* account

Set *root* user password:

- Login as general user
- *sudo -s* (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 exposes this vulnerability.

Upgrading Ubuntu

- You can do (as root):

```
# apt-get install update-manager-core
# do-release-upgrade
```

  to move between major and minor releases.
- Package sources in /etc/apt/sources.list determine available packages and from where to download.

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
```

See what's running

Check for a process by name

- `ps auxwww | grep apache`

```
sysadm@pc102:~$ ps auxwww | grep apache
root      1029  0.0 24.5 137524 125184 ?        Ss   01:29   0:02 /usr/sbin/apache2 -k start
www-data  1062  0.0 23.1 134788 117836 ?        S    01:29   0:00 /usr/sbin/apache2 -k start
www-data  1087  0.0 23.8 414236 121236 ?        Sl   01:29   0:00 /usr/sbin/apache2 -k start
www-data  1088  0.0 23.8 414236 121240 ?        Sl   01:29   0:00 /usr/sbin/apache2 -k start
sysadm    1426  0.0  0.1   3320   804 ttyS0    S+   02:40   0:00 grep --color=auto apache
```

Stop the process by PID (Process ID). From above listing:

- `sudo kill 1029` (why this one?)
- `Sudo kill -9 1029` (force stop if hung)

```
sysadm@pc102:~$ ps auxwww | grep apache
sysadm    1430  0.0  0.1   3320   808 ttyS0    S+   02:46   0:00 grep --color=auto apache
```

Viewing files revisited

Sometimes files are viewed through a pager program (“more”, “less”, “cat”). Examples:

```
man sudo
```

```
less /usr/local/etc/nagios/nagios.cfg-sample
```

- Space bar for next page
- “b” to go backwards
- “q” to quit
- “/” and a pattern (/text) to search

Kinda looks like vi, no?

“`less` is more”

Troubleshooting: Logfiles

Log files are critical to solve problems. They reside (largely) in /var/log/

Some popular log files include:

/var/log/messages or /var/log/syslog

/var/log/apache2/error.log

/var/log/mail.log

/etc/namedb/log/* (later in the week)

To view the last entry in a log file:

```
tail /var/log/messages
```

To view new entries as they happen:

```
tail –f /var/log/messages
```

There's More

But, hopefully enough to get us started...

Some Resources

http://www.ubuntu.com

http://ubuntuforums.org

http://www.debian.org

http://ubuntuguide.org

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

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

GIYF (Google Is Your Friend)

Questions

?