Preface

Before I going deep in to LFS, I quick read LFS book and write an essay of LFS Process. Here is this essay.

At this time, there are three popular operating systems, which are Windows, OSX and Linux. The first two OS's have provided stable operating system versions that are used by many companies such as Windows XP, Windows Seven, OSX Leopard, and OSX Lion. While Linux on the other hand is less popular, but it provides free open sources operating system with many versions. Generally, it is called Linux Distribution (Linux focus (SVA) on its free open sources). This means that users can implement and create their own Linux distribution. The most frequently asked question is how to create a personal Linux distribution? Linux from Scratch (LFS) is the best answer. LFS is a project that teaches users how to create a personal distribution step by step and makes sure users understand Linux structure. Before working with LFS, It requires some of UNIX administration skill to copy or delete files in directory, list files and change the path directory. The benefit of this skill is to avoid common problems. First of all a free space must be prepared for building the LFS, then the installation of the several packages is done and the last step is to build a bootable script.

First, a free space (partition) must be prepared for building a personal Linux distribution by using a Linux command (fdisk, cfdisk), because LFS is like other Operating Systems and usually installed on a dedicated partition. After a partition is prepared,it has to be mounted and integrate with the new OS**,** follow by creating two temporary folders named "/sources" and "/tools". The /sources folder contains all the packages needed to be installed and the tools folder contains the important tools for building the LFS.

The next step is to generate the LINUX command, allowing it to communicate with hardware. Installing and compiling all packages are required. Before the installation, ninety-seven sources must be downloaded and stored into the sources folder. The easiest way to download all sources is using "wget wget-list" command. It will automatically download each link. There are two steps of installing. First, Installing the GCC compiler into the temporary tools folder and then using the GCC compiler to compile and install other packages into the temporary tools folder. Now, the temporary tools folder will be called "Toolchain". Second, using the Toolchain to compile and install all packages into the LFS system.

Now, the full LFS system is built, It is time to check the bootable script (SVA). Bootable script is the script generate by the machine to oversee and direct the system when the machine boot up. According to Gerard Beekmans (2011), "Linux uses a special booting facility named SysVinit that is based on a concept of run-levels." First, the run-levels must be configured in SysVinit file and then, configure the system hostname and set the clock script to adjust the date, time and name of the LFS system. The last step is to install GRand Unified Bootloader (GRUB) to set up the boot process. In case where the system has an existing boot loader, the current boot loader will be modified**,** be sure to do it correctly because configuring GRUB incorrectly can make the system unbootable.

To create a personal distribution, First step of all is that users must have knowledge about Linux. Then, Preparing a new partition for the place where LFS will be compiled and installed. The second step is to install and compile all packages. The last step is enabling the LFS to boot through computers. After all steps have been implemented, a personal Linux distribution will be ready to be used as a command line non-graphic interface. For a wider range of software, The Beyond Linux from Scratch is the next chapter that covers more software such as Graphic User Interface, X Application etc. According to Linux From Scratch (2011)," Beyond Linux From Scratch (BLFS) is a project that continues where the LFS book finishes. It assists users in developing their systems according to their needs by providing a broad range of instructions for installing and configuring various packages on top of a base LFS system." Creating a personal Linux distribution is not a simple task. Most users will face a lot of problems and quit. I can say that every problem can be solved and do not give up easily, no matter what happens.

Problems And Solutions

Ubuntu Server 10.10 Host

Phase: Preparing a host Linux

Problem 1: What operating system I must use for hosting?

Solution: I decide Ubuntu Server 10.10 to be a host Linux for working LFS because I familiar Ubuntu bash command before. It's no need to learn more. Moreover, Ubuntu Server is having only bash command. So, It's not use a lot of CPU or RAM.
Comment: *Most packages of Ubuntu Server are matched with the Host System Requirement.*

Phase: Deciding Hardware

Problem 1: Which hardware will be used for LFS?

Solution: I decide to use MacBook Air, Core i5 with Virtual Machine because I don't have a lot of spaces to setting up boot camp.
Comment: *Many people suggest me that when compile a packages, it's use a lot of CPU. So, I have 2 choices between MacBook Air with CPU i5, 4 GB Memory and a few space of Hard Disk 128 GB Solid State. And Asus with CPU C2D, 4 GB Memories and 320 GB Hard Disk. Another condition, if I use MacBook air with few space. I must use a Virtual Machine.*

Problem 2: Which Virtual Machine will be used for LFS?

Solution: I decide to use Parallels Desktop because it's faster than VMware fusion and it's also support with OSX Lion.
Comment: *I found 2 popular softwares; VMware Fusion and Parallels Desktop for Mac. I go researched again and found that VMware Fusion is stable more than Parallels Desktop but Parallels Desktop is fast more than VMware.*

Conclusion: I use my MacBook Air with Parallels Desktop, Setting Parallels; 2 GB of memory and 60 GB Hard Disk space.

Phase: Installing Ubuntu Server 10.10

Problem 1: How much space for each partition?

Solution: I separate into 3 partitions.
/dev/sda 8 GB ext4 mounted at root
/dev/sda2 2 GB Swap partition
/dev/sda3 40 GB --- not format, leave for free spaces. This partition will work with LFS

Problem 2: mke2fs command error?

Solution: Before using mke2fs command, I must make the /dev/sda3 partition to be any type of Linux type partition. Then I can create an ext3 file system.
Comment: *mke2fs –jv is the command that create an ext3 Linux file system. –j stand for journaling file system.*

Phase: Preparing sources file

Problem 1: How to download a lot of packages in easy way?

Solution: Using wget follow by wget-list.
Comment: *wget command, this command is follow by link or file that contain the list of link. Linuxfromsratch.org provide a wget-list file. In wget-list file contain all of needed packages for LFS*

Phase: Construct a temporary system

Additional knowledge: I found the tips that make command could use for 1 core or x core depends on CPU architecture. For me, I use make –j2, which mean make will tell CPU will use for 2 cores of CPU.

Problem 1: Compiling during the Binutils packages, Mac book was rebooting OS and all files in sources folder gone.

Solution 1: Re-download sources, re-compile from the first packages and using Caffeine software on Lion OS.
Solution 2: Maybe I needed to re-mounted partition after OS was rebooting. For safety to avoid the late problems. I use the first solution.
Comment: *Caffeine is the software to prevent Mac from going to sleep.*

Problem 2: Copy files target error in API Header package.

Solution: Copy files in to the current directory and then copy its in to the /tools target folder.

Problem 3: Got 2 compile errors in Glibc package.

Solution 1: Changing the host gcc complier version.
Solution 2: Changing host Linux.
Comment: *I go to search in Google. They said I must change compiler. So I do it and compiled until Glibc. Nothing changes. I got same error and in some website wrote that; "It's cause only Ubuntu user". Now I decide to change OS.*

End of Ubuntu Server 10.10

ArchLinux Host

Phase: Preparing a host Linux

Problem 1: What is the next host Linux?

Solution: ArchLinux
Comment: *I go research Linux distribution again and found 3 Linux distro. ArchLinux have a good environment for developer but not have Graphic Interface. Damnsmall have only a basic packages I must custom packages. The size of this distro is very small as a name. Slackware is also have a good environment for developer and have a Graphic Interface. This distro come with 4 GB DVD and when installed it's use 6 GB of disk space.*

Phase: Deciding Hardware

Problem 1: Which hardware will be used for LFS?

Solution: *MacBook air with parallels Desktop (Virtual Machine)*
Comment 1: *Setting parallels; 2 cores CPU, 2 GB of memory and 50 GB free spaces.*
Comment 2: *I planned that I will install ArchLinux in parallels desktop and Slackware for dedicate laptop. If both distro were failed. I will use Damnsmall in parallels. Now I do ArchLinux first.*

Phase: Installing ArchLinux

Comment: *To setup ArchLinux is not quite hard but when I boot cd .iso in parallels. First thing to see is a bash command line (chroot CD environment). I must start set up manually by type /arch/setup command.*

Phase: Construct a temporary system

Problem 1: A little bit error when symlink (ln) in GCC package.

Solution: It's has an error but still can symlink.

Problem 2: Got 2 errors when compile Glibc package.

Solution: Create a symlink again from gcc package and then re-compile Glibc package.
Comment: *I predict that in the future it's cause the problem for sure. So I decide to move to Slackware distro.*

End of ArchLinux

Slackware

Phase: Installing Slackware

Problem 1: No partition tools during setup step.

Solution: Prepare partition first before running set up. Using fdisk or cfdisk command.
Comment: *Cfdisk is easier than fdisk because Cfdisk can see in visual text. For using Fdisk, I must use fdisk then "p" for seeing all partition available then I can delete and format. The list command is in fdisk –help.*

Problem 2: No automatically starting GUI after boot in to Slackware.

Solution: startx command to start GUI environment
Comment: *It's only have bash command after boot in to Slackware.*

Problem 3: Cannot access Internet via wireless network card

Solution: Using wicd package to connect to Internet.
Comment 1: *wicd package is the package that maintains a wireless network card. You can found wicd in the Slackware DVD; directory /extras*
Comment 2: *Slackware is very flexible Linux Distro. Slackwarer (In community, They call user who use Slackware) must configure almost everything by them selves.*

Problem 4: No network interface when run wicd package.

Solution: Using xinit instead of startx for starting GUI environment

Phase: Final preparation

Additional Knowledge: ln command is to create a symbolic link to a file. For example, gcc or make that user often to used. It's come from using a symbolic link. In a matter of fact, gcc is link to the /usr/bin/something. The link is separate in to 2 types, first call soft link then hard link. Soft link is link to the file and create a new inode. For Hard link, linking to a file and create a new inode. So, if delete the original file. The link with soft link will crash but hard link still alive. The command for create hard link, ln file1 file2. For soft link, ln –s file1 file2
Additional Knowledge: inode carries the map of where data is it, the file permissions, etc.

Phase: Constructing a temporary system

Comment: When the files were .tar.gz It usually to use tar –xvz command. For .tar.bz2, zip compress using tar -jxvf

Additional Knowledge: Linux sed command is the stream editor used to perform basic text transformation on an input stream. For example In Linux command, sed 's/libgcc/&_eh/'; s stand for string, which would change libgcc to (suffix)_eh.

Phase: Installing basic system software

Problem 1: Man-pages not found in sources folder.

Solution: Download manually from Google.

Problem 2: Got 2 errors when make a testing with Glibc; test-aio4.out and posix/annexc.out

Solution: In document say; It's been safe to ignore.

Problem 3: Install error in Binutils package.

Solution: Re-install and follow the command that solve PTYs problems
Comment: *My Slackware has PTYs and I skipped the command to solve about PTYs problems.*

Problem 4: Many unexpected failures when making test in GCC package.

Solution: In LFS document say; A few unexpected failures cannot always be avoid.

Problem 5: IPRoute2 not found in sources folder.

Solution: Download manually from Google.

Problem 6: Bash command was crashed after making test in Vim package.

Solution: Re-install Slackware, re-do LFS from the beginning and not test with a vim package
Comment: *In document say; The bash command will error during a testing with a vim package because it's has a lot of binary file to read in bash.*

Phase: Installing basic system software

Problem 1: Don't know the IP address of DNS server

Solution: Enter IP DNS server later because It's depend on the ISP provider that will be used.

Phase: Using GRUB to set up boot process

Problem 1: My host has a boot loader LILO already.

Solution: To avoid the system inoperable, I must edit the host boot loader file.
First step, Copy the vmlinuz-3.1-lfs-7.0 kernel file from /mnt/lfs/boot/ to the host /boot/. Then, edit /etc/lilo.conf by adding the LFS code to be bootable.

Problem 2: After reboot, LILO is not update. It's has only one Slackware partition

Solution: Update LILO configure file by using lilo command on bash.

*** I move LFS to parallels desktop version 7**

Slackware on parallels desktop version 7

Phase: Boot up LFS

Problem 1: Setting up Linux console fail

Solution: Ignore this problem but Linux console still work well.
Comment: *I research in Google but its don't have any solution, so I mail to the LFS developer to find the solution. Content in mail*
" I finished the LFS Version 7 already and got this problem on boot process

Linux Host
Slackware 13.37 X86

Parallels Desktop Version 7 Setting
CPU Core i5
Memory 2 GB

I used LILO instead of GRUB that provided in LFS book

When boot process I got this;

Setting up Linux console... FAIL
FAILURE:

You should not be reading this error message.

It means that an unforeseen error took place in /etc/rc.d/rcS.d/S70console, which exited with a return value of 1. If you're able to track this error down to a bug in one of the files provided by the files provided by the book, please be so kind to inform us at lfs-dev@linuxfromscratch.org.

Anyway, I can go through the console. "

Solution 2: Clear all console files.
Comment: *Empty the /etc/sysconfig/console*

Problem 2: Keyboard map incorrect on Linux console

Solution: Editing keymap and font variable in /etc/sysconfig/console
Comment: *KEYMAP="pl2"*

FONT="lat2a-16 -m 8859-2"

Conclusion: After clearing a console file in /etc/sysconfig/console, Problem 1 and 2 were solved.

Linux from scratch finished (Slackware)

Beyond Linux from scratch

Package: GPM-1.20.6
Problem: It's not work after installation
Solution: Changing value of MDEVICE; /dev/input/mouse0 instead of /dev/psaux
Comment: *Because of parallels desktop is virtual machine so, It's not direct path to mouse device*

Package: Desktop-file-utils
Problem: ./configure have error
Solution: Install pkg-config package first.
Comment: *In order to make a test in Glib package, Installing pkg-config after install Glib, follow by desktop-fileutil.*

Package: Lesstif
Problem: /usr/lib/libstdc++.so.6.0.16-gdb.ppy is not an ELF file
Solution: Delete that file

Package: MesaLib
Problem 1: libdrm_nouveau.so was not found when ./configure
Solution: Compile libdrm-2.4.27 with "--enable-nouveau-experimental-api". Then build
MesaLib-7.11.2 according to the book

Problem 2: configure: error: LLVM is required to build Gallium R300 on x86 and x86_64
Solution: add --with-gallium-drivers="nouveau,swrast" when configuring

Additional Knowledge: Additionally, because of the large number of repetitive commands, you are encouraged to partially automate the build. The commands below (or similar) can be entered at the command line to compile each section

```
for package in $(grep -v '^#' ../${section}-${version}.wget)
do
  packagedir=$(echo $package | sed 's/.tar.bz2//')
  tar -xf $package
  cd $packagedir
// configure and install (Depend on section)
  cd ..
```

rm -rf $packagedir
done 2>&1 | tee -a ../xorg-${section}-compile.log

Package: Qt-3.38d
Problem: Error while loading shared library
Solution: Skip this package because Qt-3.38d is for the Desktop environment KDE and I plan to use Gnome

Package: caribou-0.4.1
Problem: gnome-doc-utils was not found while configure
Solution: Install gnome-doc-utils frist

Package: NSS-3.13.3, Heimdal-1.4
Problem: Compile Error
Solution: compile make flags with –j 1 instead of –j 4 for more efficiently

Package: Cyrus SASL-2.1.23
Problem: Boot and halt error
Solution: To prevent the message errors on startup or halt, Do not install the blfs boot script cyrus sasl. If installed it's, You must remove the target path blfs boot script manualy

Package: Udev
Problem: udev173 not available to download in blfs site
Solution: find udev173 in google and manually building configure by reading and modify document in package

Package: mobile-broadband-provider-info-20110511
Problem: variable source error
Solution: find package in google

Package: mod_dnssd-0.6
Problem: Package avahi-client not found but I build avahi already
Solution: Go back to avahi package and see the information after configure, The command in blfs book not to build client by default so , I rebuild again with remove –disable-python , –disable-libdaemon –disable-dbus to install avahi client.

Package: cdrtool and dvd+rwtool
Problem: Patch not found
Solution: Find another link on linux from scratch website

Package: Poppler-0.14.4
Problem: Z_BEST_COMPRESSION not declare in this scope
Solution: –disable-libpng (Not sure if this solution effect in long term ?)

Package: udisks-1.0.1
Problem: libatasmart not found while configuring
Solution: Rebuild and add configure option manualy by adding –sysconfdir and -localstatedir

Package: Gnome disk utility
Problem: avahi-ui-gtk3 was not foubd while configuing
Solution: Go back to avahi package and see the information after configure, The command in blfs book not to build client by default so , I rebuild again with remove –disable-gtk3 and install python gtk module to build avahi success

Package: HAL-0.5.14
Problem: Build error
Solution: Install V4l-utlils first (Not provide in blfs)
Comment: http://www.mail-archive.com/blfs-book@linuxfromscratch.org/msg17568.html

Package: Totem-3.2.1
Problem 1: Package mx not found
Solution: Download and install manually because blfs not provide this package

Problem 2: Configuring error, Gstreamer good plugin not found
Solution: Go back to Gstreamer good plugin then build with --sysconfdir=$GNOME_SYSCONFDIR to ensure that Gstreamer will build with Gconf support

Package: Gegl-0.1.8
Problem: ArgumentError (invalid byte sequence in US-ASCII)
Solution: Set environment export LANGUAGE=en_US.UTF-8
export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8

Problem: How to transfer Parallels Desktop(Mac) to Parallels Workstation (Windows)
Solution: Move only virtual hard disk without parallels tool installed and Create new virtual machine with an existing virtual hard disk

Problem: How to grab the standard output
Solution: Overriding the command is the best way
Comment:

Bash doesn't generally produce much output by itself. Rather, it's used to execute commands, and some of those commands may produce output.

If you want to filter the output of a script as described above, the best way would be to pipe the script through sed:

#!/usr/bin/env bash
exec 1> >(dictionaryScript)
... the ENTIRE script goes here ....

However, doing this causes issues with timing, because now your output
is being filtered by a background process instead of a foreground process.
Some people won't care about that, and some people will.

// “1” of the exec 1>, mean redirect all of the standard output

Problem: How to translate the output from echo command
Solution: First find data dictionary from Lexitron. It will fetch the xml file and then open the file through Microsoft access and reducing the data to two columns. The first column is English vocabulary and the second is the result of translating the first column into Thai. The

method to reduce other unnecessary column is to use the VI command in VI editor. :%s/’//g to delete the ‘ that will cause error with sed command and awk -F, '{print $1 "," $2}' to limit one comma. After that, override echo function by send sed command to do the job instead of bash

Comment:

```
echo(){
command echo $* | sed -f <(sed 's/\(.*\),\(.*\)/s,\\<\1\\>,\2,gI/'
/home/opponen
t/Downloads/mydic)
}
```

Thanyasit OS V1

OS type: Linux
Based on: Independent
Developer: Thanyasit Lieamsiriwong
Architecture: i686
Desktop: GNOME3
Category: Developer, Programming, Source based, Web Server

Thanyasit Linux is building for developer and programmer, fast and comfortable for old desktop because consume a low resource, completely free Linux distribution. Unlike other distros, Thanyasit Linux not provides any package management. The benefits of this are developer can download the package sources and specific compiling with their own configuring. The second thing is to make you learn how to build the package without using automates building (provide by package management). For the programmer, this distribution provides many popular languages package like Python, C/C++ (default by Linux), Perl, Java, PHP and so on. Thanyasit distribution also provides the useful specific key shortcut for Kernel Developer, Programmer and Web Server. Thanyasit Linux using XAMPP (default) for Web Server. The default environment after user login was set to the Command Line Interface, This point is very useful for Web Server. How about the GUI, Sure Thanyasit OS provide X environment (GUI) called GNOME3 but It has only the cores but use the basic GNOME application and some programming application like Netbeans.**The OS will translate every known echo command into Thai language**. Like I said, Thanyasit Linux is a free open source and I glad to allows developer to implement my first Linux OS.

Specific Command

goKernel - Access to the kernel directory
goPython2 – Call the python version 2
goPython3 – Call the python version 3
switchToPython2 – Change the default Python to version 2
switchToPython3 – Change the default Python to version 3
goXampp – Access the Xampp directory
viewApache – View the Web server configuration
goApacheDoc - Access the place where storing a website
echo – Translate output to Thai laungage

Package List (Only Beyond Linux From Scratch)

- **For description each packages**
- **Visit http://www.linuxfromscratch.org/blfs/view/svn/**

Security

AccountsService-0.6.15
acl-2.2.51
attr-2.4.44
ConsoleKit-0.4.5
Cyrus SASL-2.1.23
Iptables-1.4.12
Heimdal-1.4
liboauth-0.9.4
MIT Kerberos V5-1.6
nettle-2.4
NSS-3.13.3
OpenSSL-1.0.0g
Certificate Authority Certificates
GnuTLS-3.0.7
Linux-PAM-1.1.5
p11-kit-0.10
polkit-0.104
polkit-gnome-0.104
Shadow-4.1.4.3
Stunnel-4.52
Sudo-1.8.2
TCP Wrappers-7.6
Tripwire-2.4.2.2

File Systems and Disk Management

lvm2-2.02.67

Editors

Vim-7.3
Nano-2.1.10

General Libraries and Utilities

General Libraries

PCRE-8.30
libxml2-2.7.8
libxslt-1.1.26
GLib-2.30.2
glibmm-2.30.0

Libcroco-0.6.2
libgsf-1.14.21
libglade-2.6.4
Expat-2.0.1
Aspell-0.60.6
enchant-1.6.0
libusb-1.0.8
ICU-4.8.1.1
ISO Codes-3.30
GMime-2.4.21
libatomic_ops-1.2
libdrm-2.4.27
Pth-2.0.7
libgpg-error-1.10
libgcrypt-1.5.0
libtasn1-2.10
libunique-3.0.2
liboil-0.3.17
NSPR-4.9
libffi-3.0.10
gobject-introspection-1.30.0
libgee-0.6.0
libical-0.48
libsigc++-2.2.10
libdaemon-0.14
libatasmart-0.17
talloc-2.0.7
telepathy-glib-0.16.0
telepathy-logger-0.2.10
telepathy-mission-control-5.9.1
Folks-0.6.4.1
Boost-1.49.0
libunistring-0.9.3
JSON-GLib-0.14.2
exempi-2.1.1
Apr-1.4.6

Graphics and Font Libraries
libjpeg-8c
libpng-1.5.9
LibTIFF-4.0.1
little cms-1.19
lcms2-2.3

libmng-1.0.10
FreeType-2.4.8
Fontconfig-2.8.0
librsvg-2.34.2
libexif-0.6.19
Exiv2-0.22
Poppler-0.14.4
JasPer-1.900.1
pixman-0.24.0
clutter-1.8.2
clutter-gst-1.4.4
clutter-gtk-1.0.4
mx-1.2.1
babl-0.1.6
gegl-0.1.8
colord-0.1.14

General Utilities
autogen-5.12
bc-1.06.95
GTK-Doc-1.18
Intltool-0.50.0
Screen-4.0.3
desktop-file-utils-0.18
XScreenSaver-5.15
icon-naming-utils-0.8.90
Gperf-3.0.4
Rarian-0.8.1
Apr-Util-1.4.1

System Utilities
Which-2.20 and Alternatives
UnZip-6.0
Zip-3.0
UnRar-3.9.10
PCI Utilities-3.1.8
usbutils-004
D-BUS-1.4.16
D-Bus Bindings
HAL-0.5.14
Udev-Installed LFS Version
UPower-0.9.14
sg3_utils-1.29

Parted-3.0
udisks-1.0.1
Eject-2.1.5
gvfs-1.10.1
libarchive-2.8.5
packagekit-0.7.1
tracker-0.12.3

Programming
Check-0.9.8
CMake-2.8.6
DejaGnu-1.5
Doxygen-1.7.5
GC-7.1
GCC-4.5.1
gdb-7.3.1
Git-1.7.9.2
Guile-2.0.3
JDK-6 Update 18
LLVM-3.0
Perl Modules
PHP-5.3.8
pkg-config-0.26
Python-2.7.2
Python-3.2.2
Python Modules
Ruby-1.9.2-p290
Subversion-1.7.3
SpiderMonkey-1.0.0
Tcl-8.5.10
Vala-0.14.1
yasm-1.2.0
Other Programming Tools
Netbean

IV. Networking
Connecting to a Network
dhcpcd-5.2.12

Networking Programs
Net-tools-1.60
Wget-1.13.4
Wireless Tools-28

Networking Utilities
Traceroute-2.0.18
Whois-5.0.12
avahi-0.6.28
mod_dnssd-0.6
NetworkManager-0.9.2.0

Networking Libraries
cURL-7.24.0
libgdata-0.10.1
libnice-0.1.1
libnl-3.2.3
librest-0.7.12
libsoup-2.36.1
neon-0.29.6

Servers
Major Servers
Apache-2.4.1
BIND-9.8.1-P1
XAMPP (a.k.a LAMP)

Mail Server Software
Postfix-2.8.4
Sendmail-8.14.4

Databases
Berkeley DB-5.2.36
MySQL-5.5.17
SQLite-3.7.10

Other Server Software
OpenLDAP-2.4.23

VI. X + Window Managers
X Window System Environment
Introduction to Xorg-7.6-2
util-macros-1.15.0
Xorg Protocol Headers
makedepend-1.0.3
libXau-1.0.6
libXdmcp-1.1.0

libpthread-stubs-0.3
xcb-proto-1.6
libxcb-1.7
Xorg Libraries
xcb-util-0.3.8
MesaLib-7.11.2
Xorg Applications
xcursor-themes-1.0.3
Xorg Fonts
XKeyboardConfig-2.0
Xorg-Server-1.11.2
Xorg Drivers
xterm-276
Xorg-7.6-2 Testing and Configuration

25. X Libraries
cairo-1.10.2
cairomm-1.9.2
Pango-1.29.4
pangomm-2.28.4
atk-2.2.0
atkmm-2.22.6
gdk-pixbuf-2.24.0
gtk+-2.24.8
gtk+-3.2.3
gtkmm-3.2.0
LessTif-0.95.2
startup-notification-0.12
libwnck-3.2.1
shared-mime-info-0.91
hicolor-icon-theme-0.12
libxklavier-5.1
freeglut-2.8.0
WebKitGTK+-1.6.1
libsexy-0.1.11
libnotify-0.7.4
notification-daemon-0.7.3
cogl-1.8.2

Other Window Managers
IX. GNOME
GNOME Core Packages
Libraries

caribou-0.4.1
desktop-file-utils-0.18
GConf-3.2.3
gjs-1.30.0
GNOME Desktop-3.2.1
GNOME Doc Utils-0.20.6
gnome-js-common-0.1.2
libgnomekbd-3.2.0
libgnome-keyring-3.2.2
LibGTop-2.28.4
libgweather-3.2.1
libpeas-1.2.0
seed-3.2.0
shared-mime-info-0.91
totem-pl-parser-2.32.6
VTE-0.30.1
yelp-xsl-3.2.1
Zenity-3.2.0

Shell
dconf-0.10.0
Evolution Data Server-3.2.2
glib-networking-2.30.1
gnome-bluetooth-3.2.1
GNOME Control Center-3.2.2
GNOME Icon Theme-3.2.1.2
gnome-icon-theme-extras-3.0.0
gnome-icon-theme-symbolic-3.2.1
gnome-keyring-3.2.2
gnome-menus-3.2.0.1
gnome-online-accounts-3.2.1
gnome-packagekit-3.2.1
gnome-screensaver-3.2.0
GNOME Session-3.2.1
GNOME Settings Daemon-3.2.2
gnome-shell-3.2.1
gnome-themes-standard-3.2.1
gsettings-desktop-schemas-3.2.0
gvfs-1.10.1
mousetweaks-3.2.1
mutter-3.2.1
network-manager-applet-0.9.2.0
PulseAudio-0.9.23

telepathy-mission-control-5.9.1

Extras
gnome-backgrounds-3.2.0
gnome-user-share-3.0.1
GNOME User Docs-3.2.2
Vino-3.2.2

Utilities
brasero-3.2.0
Empathy-3.2.2
EOG-3.2.2
Epiphany-3.2.1
Evince-3.2.1
gcalctool-6.2.0
gnome-contacts-3.2.2
gnome-disk-utility-3.0.2
GNOME System Monitor-3.2.1
GNOME Terminal-3.2.1
GNOME Utilities-3.2.1
gucharmap-3.2.2
Nautilus-3.2.1
sushi-0.2.1
Yelp-3.2.1

Shell Fallback
GNOME Panel-3.2.1
Metacity-2.34.1
notification-daemon-0.7.3
dbus-python-0.84.0
polkit-gnome-0.104

OS Services
GDM-3.2.1.1

GNOME Additional Packages
Libraries
at-spi2-atk-2.3.1
at-spi2-core-2.3.1
gdl-3.2.0
gdlmm-3.2.1
GNOME Applets-3.2.1
gnome-video-effects-0.3.0

gssdp-0.12.0
GtkHTML-4.2.2
gtksourceview-3.2.3
gtksourceviewmm-3.2.0
gupnp-0.18.0
gupnp-av-0.10.1
gupnp-dlna-0.6.4
gupnp-vala-0.10.2
libgda-4.2.4
mobile-broadband-provider-info-20110511

Utilities
accerciser-3.2.1
aisleriot-3.2.2
anjuta-3.2.1
cheese-3.2.2
devhelp-3.2.0
File Roller-3.2.2
gedit-3.2.3
ghex-3.0.0
glade-3.10.2
gnome-color-manager-3.2.1
gnome-devel-docs-3.2.1
gnome-documents-0.2.1
GNOME Games-3.2.1
gnome-nettool-3.0.0
gnome-power-manager-3.2.1
Orca-3.2.2
nautilus-sendto-3.0.1
nemiver-0.9.0
rygel-0.12.5
Seahorse-3.2.2
Totem-3.2.1
Vinagre-3.2.2

Deprecated GNOME Packages
GNOME Virtual File System-2.24.4

X Software
Evolution-3.2.2
Gimp-2.6.12

Multimedia

Multimedia Libraries and Drivers
libogg-1.3.0
libvorbis-1.3.2
libcanberra-0.28
FAAC-1.28

Speex-1.2rc1
libdvdcss-1.2.10
GStreamer-0.10.35
GStreamer Base Plug-ins-0.10.35
GStreamer Good Plug-ins-0.10.30
GStreamer Bad Plug-ins-0.10.21
GStreamer Ugly Plug-ins-0.10.18
gst-ffmpeg-0.10.13
Farsight2-0.0.26
telepathy-farsight-0.0.14
libmusicbrainz-3.0.3
libsndfile-1.0.23
PulseAudio-0.9.23

CD/DVD-Writing Utilities
Cdrtools-2.01
dvd+rw-tools-7.1

Standard Generalized Markup Language (SGML)
DocBook DSSSL Stylesheets-1.79
DocBook-utils-0.6.14

Extensible Markup Language (XML)
DocBook XML DTD-4.5
DocBook XSL Stylesheets-1.76.1
xmlto-0.0.23
itstool-1.1.1

PostScript
FOP-1.0

Other PostScript Programs
Typesetting
TeX Live-20110705

Experiences

The first thing that I learned is How to build my own Linux distribution, Start from finding a host to build my distro. Then create a partition, building toolchain to compile basic package system, enter the chroot environment to compile a basic system, setting boot script and configuring the system, deploy kernel on my system and make my Linux can be bootable (update LILO). Go though the Beyond Linux From Scratch, find the package that I need to build and Installing GNOME3 core packages.

The second thing that I learned is command in Linux, I explore the command that I never see before like sed; is to change character (also string depend on command) in file. Ln; is create a symbolic link with the same inode or not (depend on command –s is for softlink) etc. Moreover, I also learn to use the vi and screen (this is very useful package). Screen is to create a virtual terminal and can switch immediately while compiling a package. Gpm to allow the you access the mount to copy and paste.

The third thing is how package work together especially XORG and GNOME3 . As I build more than 300 packages, I learned the dependencies in each package and also how to build package related to other packages with customize configuration.

The fourth thing is Software,OS and Hardware dependencies, As I use the virtual machine software (Parallels Desktop for MAC), I face many problem like I want to transfer to Windows using Parallels Workstation or VMware Workstation but it cannot; The solution is move only virtual hard disk without parallels tool installed. How to shared file between MAC and Windows and driver from parallels is not work well while using GUI (GNOME3).

The fifth thing is some bash shell programming, it's starting at the question "How to grab the output from the bash outside script scope". Some people suggest me to grab the output from /proc/$BASHPID but it is very messy. So, I try to grab the output using shell script. I learned the strange bash command like PROMPT_COMMAND, trap, exec 1> >($command) and so on. Moreover, I learned how to override and alias builtin bash command like function echo{ code …} (This is bashrism)or in POSIX Linux style is echo(){code …}. Also the parameter that pass to the echo by using $ sign. After I mail to Bash GNU developer, I know the basic number level of the standard Input/Output; which are 0 1 2, 0 stand for Input, 1 stand for Output and 2 stand for error output. Combining with the command exec, exec 1> >($command)

means redirect all standard output to the $command. Inside the bash source code (or bash tarball), According to Greg Wooledge, “Bash doesn't generally produce much output by itself. Rather, it's used to execute commands, and some of those commands may produce output.”. In most bash script, It’s contain the #!/usr/bin/env $symlinkCommand on the top of the script. Env is very useful to specific the symlinkCommand such as you have 2 python versions python2.7 and python3; /usr/bin/env /usr/bin/python2.7 mean you specific the python version 2.7. I have studied a lot from using VI. VI is not just an ordinary text editor, it can find and replace the text with more advance condition and option. The SED command is another thing that I have learned. It is similar to VI in the matter of condition can it can function in a more complex task.

The last thing … As I face with a lot of problems while building package. Now I absolutely understand how to install each package and also fix some bug that BLFS team not provide in the book! (actually It’s depend on error) For example, Some error must change or customize my own configuration, change the environment flags like pkgconfig(CFLAGS CCFlags)to specific the path and make the system know where binary file will system use, decreasing MAKEFLAGS to avoid the error in important package, Go back to previous dependencies package rebuild with my own customize configuration. This is just the example, Now I like to download the package source and customize my own configuration. It’s very fun if I face with error.

In conclusion, I learned many things from LFS and BLFS and I plan to learn more about Linux Structure that how Linux work in deep. And also expose the Kernel that I think is very interesting.