Pembangunan Sistem Operasi Berbasis Linux Menggunakan Metode *Linux From Scratch*

[1]Wamiliana, [2]Wisnu Wardhana dan [3]Fahmi Kharismaldie

[1]Jurusan Matematika FMIPA Unila
[2]CV. Linux Lampung
[3]Jurusan Ilmu Komputer FMIPA Unila

Abstract

GNU/Linux is an operating system that allows the users to build or develop their own operating system. There are two methods that can be used, Remaster and Linux From Scratch. The difference is the LFS method builds a system from zero while Remaster using existing Linux distributions [4]. The goal of this research is to investigate the steps of building Linux base operating system using LFS method. The operating system has been built successfully and can be used to fulfill the daily needs such as playing music or video, browsing, programming, and other tasks.

Keywords: *GNU/Linux, Linux From Scratch, open source, operating system*

1 Pendahuluan

Sistem operasi adalah salah satu komponen vital dalam sistem komputer. Sistem operasi bertugas untuk mengelola pemakaian perangkat keras komputer seperti prosesor, memori, perangkat I/O, media penyimpanan, dll. Selain itu, sistem operasi juga berfungsi sebagai suatu lapisan yang menghubungkan perangkat keras komputer dengan perangkat lunak [5][6][8].

Sistem operasi hingga saat ini terus berkembang. Salah satu sistem operasi yang menjadi pusat perhatian dalam perkembangan ini adalah GNU/Linux. GNU/Linux menjadi pusat perhatian karena perkembangannya yang sangat cepat. Hal ini dimungkinkan karena GNU/Linux menganut filosofi *open source* [2][3]. Perangkat lunak *open source* memberikan *source code*[1] yang secara bebas dapat digunakan, dipelajari, didistribusikan dan dikembangkan kembali oleh para pengguna [2][3][7]. Oleh karena itu, perkembangan sistem operasi GNU/Linux mengalami pertumbuhan yang pesat.

Para pengguna sistem operasi GNU/Linux diberikan kesempatan untuk membangun dan mengembangkan sistem operasi yang sesuai dengan kebutuhan mereka masing-masing [1]. Terdapat dua metode yang dapat digunakan, yaitu metode *Remaster* dan *Linux From Scratch (LFS)*. Perbedaan keduanya adalah *Linux From Scratch* membangun sistem operasi dari nol sedangkan *Remaster* membangun atau mengembangkan distribusi Linux yang sudah ada [1][4].

Pada tulisan ini akan didiskusikan mengenai pembangunan suatu sistem operasi Linux menggunakan metode *LFS*. Bagian dua dari tulisan ini adalah berisi tentang metode yang digunakan, bagian tiga tentang pembahasan dan di bagian empat diberikan kesimpulan.

[1] *Source code* adalah sejumlah perintah komputer yang ditulis oleh programmer untuk membuat program atau *software*. Perintah ini ditulis dalam bahasa pemrograman seperti C atau Java. [7]

2 Metode Penelitian

2.1 Studi Literatur

Pada tahap ini dilakukan pencarian konsep dan informasi yang dibutuhkan dalam pembangunan sistem operasi berbasis Linux. Pencarian dilakukan pada buku, jurnal, artikel, internet, dan sumber informasi lainnya. Kata kunci dari pencarian ini adalah sistem operasi, *free software, open source, GNU/Linux*, metode pembangunan sistem operasi Linux, dan *Linux From Scratch*.

2.2 Analisis dan Perancangan

Analisis dilakukan untuk mengetahui apa saja yang dibutuhkan untuk membangun sistem operasi Linux dengan metode *LFS*. Setelah itu rancangan sistem operasi dibangun berdasarkan kebutuhan tersebut. Sistem operasi ini dibuat sebagai *desktop* untuk memenuhi kebutuhan sehari-hari, seperti menjelajah internet, memutar musik dan video, mengetik, *programming*, dan lain-lain. Sistem operasi ini juga dibangun secara sederhana dan seminimal mungkin.

2.3 Pembangunan Sistem *LFS*

Pada tahap ini dimulai pembangunan sistem operasi Linux menggunakan metode *LFS*. Secara garis besar pembangunan dimulai dari persiapan *host system*, pembuatan partisi baru, pembangunan *temporary system*, dan pembangunan sistem operasi. Seluruh tahapan ini mengacu pada standar *POSIX*[2], *LSB (Linux Standar Base)*, dan *FHS (File System Hierarchy)* yang merupakan standar dalam sistem operasi berbasis Linux [1]. *S*tandar-standar tersebut diperlukan untuk menjamin kualitas dari sistem operasi yang dibangun.

2.4 Pemasangan Paket *Software* Tambahan

Pada tahap ini dilakukan pemasangan paket *software* untuk menambah fitur pada sistem operasi yang dibangun. Fitur-fitur ini diperlukan untuk mendukung sistem keamanan, *file system, general library, general utility, system utility, programming, networking, multimedia,* dan *GUI (Graphical User Interface).*

2.5 Pengujian

Pengujian dilakukan untuk mendapatkan berbagai informasi dari sistem operasi yang telah dibangun. Informasi tersebut berkaitan dengan fungsi-fungsi yang terdapat dalam sistem operasi. Pengujian ini bertujuan untuk memastikan bahwa seluruh fungsi dapat dijalankan oleh sistem operasi. Fungsi-fungsi tersebut diantaranya *boot process, login system, shell, GUI, storage, networking, multimedia,* dan *programming.*

3 Pembahasan

3.1 Analisis dan Perancangan

Terdapat dua kebutuhan utama yang harus dipenuhi untuk membangun sistem operasi Linux menggunakan metode *LFS*, yaitu kebutuhan *software* dan *hardware*. Pada bagian *hardware* diperlukan sebuah komputer yang akan berperan sebagai *host*, sedangkan pada bagian *software* dibutuhkan beberapa paket *software* sebagai bahan untuk membangun sistem. Paket *software* ini terdiri dari *source code* program dan beberapa diantaranya berupa *software patch* dan kernel[3] linux.

[2] ***POSIX*** adalah *Portable Operating System Interface, X* di sini mengacu pada sistem berbasis UNIX, di lain hal diartikan juga *Portable Operating System Standard for Computer Environment* yaitu sebagai standarisasi karakteristik sistem UNIX. [5][6][8]

[3] Kernel adalah inti dari sistem operasi yang bertanggung jawab untuk mengendalikan dan mengolah kerja seluruh *hardware* komputer. [5][6][8]

Table 1 Spesifikasi *Hardware Host* dan Target

No	***Hardware*/ *Software***	**Keterangan**
1	Sistem Operasi	Debian Squeeze 6.0.6
2	Arsitektur	32 bit
3	Prosesor	Intel Pentium Dual Core T4300 (2,1 GHz, 800 MHz FSB)
4	Memori	DDR2 3 GB
5	VGA	Intel GMA 4500M
6	Ethernet	Atheros AR8131
7	Wireless	Atheros AR928X
8	Audio	Intel HD Audio
9	Storage	250 GB 5400 RPM

Setelah seluruh kebutuhan *hardware* dan *software* terpenuhi, perlu dilakukan konfigurasi *hardware* dan *software* pada *host*. Selanjutnya seluruh paket *software* dapat dibangun.

3.2 Pembangunan Sistem Operasi

Pembangunan sistem terdiri dari tahap persiapan *host system*, pembuatan partisi, pembangunan *temporary system*, dan pembangunan sistem *LFS*.

3.2.1 Persiapan *Host System*

Pada tahap ini dilakukan pemasangan sistem operasi dan seluruh *software* yang diperlukan untuk membangun paket *software LFS*. Sistem operasi yang digunakan adalah *Debian Squeeze 6.0.6*. Pada sistem operasi ini sudah terdapat beberapa *software* yang diperlukan. Setelah itu dilakukan pengecekan terhadap *software* tersebut berdasarkan ketersediaan dan versinya.

Pada Figure 1 terdapat s*cript* yang berfungsi untuk mengecek keberadaan dan versi dari *software* yang ada pada sistem operasi. *Script* tersebut *bash*[4] *script*. Alur dari *script* ini terdiri dari buat *file script*, tulis *script*, dan eksekusi *script*. Perintah untuk membuat *file* ditunjukkan oleh baris pertama. Penulisan *script* ditunjukkan dari baris ke-2 dari awal hingga baris ke-2 dari akhir. Perintah untuk mengeksekusi *script* ditunjukkan oleh baris terakhir.

Berdasarkan hasil yang ditunjukkan oleh *script* tersebut, terdapat lima *software* yang belum tersedia pada *host*. Paket-paket tersebut adalah *Binutils*, *Bison*, *Gawk*, *GCC*, dan *Make*. Untuk menambahkan paket-paket tersebut pada *host*, dapat dilakukan eksekusi perintah berikut ini.

[4] *Bash (Bourne Again Shell)* adalah *shell* atau antar muka pengguna yang digunakan sebagai tempat untuk memberikan perintah pada komputer. *Bash* biasa digunakan pada sistem operasi berbasis UNIX dan Linux.

```
$ apt-get install <nama software>
```

Perintah tersebut berfungsi untuk mencari *software* yang diinginkan dari repositori dan memasangnya secara otomatis.

```
$ cat > version-check.sh << "EOF"
#!/bin/bash
# Simple script to list version numbers of critical development tools
export LC_ALL=C
bash --version | head -n1 | cut -d" " -f2-4
echo "/bin/sh -> `readlink -f /bin/sh`"
echo -n "Binutils: "; ld --version | head -n1 | cut -d" " -f3-
bison --version | head -n1
if [ -e /usr/bin/yacc ];
  then echo "/usr/bin/yacc -> `readlink -f /usr/bin/yacc`";
  else echo "yacc not found"; fi
bzip2 --version 2>&1 < /dev/null | head -n1 | cut -d" " -f1,6-
echo -n "Coreutils: "; chown --version | head -n1 | cut -d")" -f2
diff --version | head -n1
find --version | head -n1
gawk --version | head -n1
if [ -e /usr/bin/awk ];
  then echo "/usr/bin/awk -> `readlink -f /usr/bin/awk`";
  else echo "awk not found"; fi
gcc --version | head -n1
ldd --version | head -n1 | cut -d" " -f2- # glibc version
grep --version | head -n1
gzip --version | head -n1
cat /proc/version
m4 --version | head -n1
make --version | head -n1
patch --version | head -n1
echo Perl `perl -V:version`
sed --version | head -n1
tar --version | head -n1
echo "Texinfo: `makeinfo --version | head -n1`"
xz --version | head -n1
echo 'main(){}' > dummy.c && gcc -o dummy dummy.c
if [ -x dummy ]
  then echo "gcc compilation OK";
  else echo "gcc compilation failed"; fi
rm -f dummy.c dummy
EOF
$bash version-check.sh
```

Figure 1 *Script* untuk Cek Versi *Software*

3.2.2 Pembuatan Partisi

Pada pembuatan partisi ini sekitar 60 GB dialokasikan sebagai tempat untuk menaruh sistem *LFS* yang dibangun. Proses pembuatan dilakukan dengan menggunakan program *e2fsprogs* yang dibangun sendiri. Hal ini dikarenakan program *e2fsprogs* yang sudah terpasang pada *host* telah memiliki beberapa fitur tambahan yang tidak cocok digunakan pada sistem yang akan dibangun. Bila program ini digunakan, akan mengakibatkan kesalahan pada saat proses *booting* sistem. Partisi ini dibentuk dengan *file system ext3.*

```
cd /tmp
tar -xzvf /e2fsprogs-1.42.5.tar.gz
cd e2fsprogs-1.42.5
mkdir -v build
cd build
../configure
make       # "make install" tidak dilakukan karena program
           # hanya dibangun, tidak dipasang
./misc/mke2fs -jv /dev/sda9
cd /tmp
rm -rfv e2fsprogs-1.42.5
```

Figure 2 Kompilasi Paket *e2fsprogs*

3.2.3 Pembangunan *Temporary System*

Temporary system adalah suatu kesatuan yang terdiri dari *compiler*, *assembler*, *linker*, *library*, dan *utility*. *Temporary system* ini digunakan untuk kompilasi paket-paket *software* sistem LFS yang bebas dari pengaruh program pada *host*.

Terdapat 28 paket *software* yang dikompilasi untuk membangun *temporary system* ini. Seluruh paket tersebut dikompilasi dengan prosedur umum sebagai berikut.

a. *Shell* yang digunakan adalah *bash* dan *user* yang digunakan adalah *lfs*.
b. Seluruh paket *software* sistem LFS disimpan dalam satu direktori `/mnt/lfs/sources`.
c. Pindah direktori ke `/mnt/lfs/sources`.
d. Ekstrak paket *software* yang akan dibangun.
e. Pindah direktori ke paket *software* yang baru diekstrak.
f. Bangun paket *software*.
g. Pindah kembali ke direktori `/mnt/lfs/sources`.
h. Hapus direktori hasil ekstrak.

Berikut ini adalah contoh kompilasi paket *software* Binutils versi 2.22.

```
tar xvfj binutils-2.22.tar.bz2
cd binutils-2.22/
patch -Np1 -i ../binutils-2.22-build_fix-1.patch
mkdir -v ../binutils-build
cd ../binutils-build

../binutils-2.22/configure \
--prefix=/tools     \
--with-sysroot=$LFS      \
--with-lib-path=/tools/lib     \
--target=$LFS_TGT     \
--disable-nls     \
--disable-werror

make
make install
```

Figure 3 Membangun Paket Binutils-2.22

3.2.4 Pembangunan *LFS System*

Tahap ini adalah tahapan utama dari pembangunan sistem *LFS*. Pada tahap ini dilakukan kompilasi paket-paket *software* yang menjadi bahan utama pembangunan sistem *LFS*. Kompilasi dilakukan menggunakan *temporary system* yang telah berhasil dibangun pada tahap sebelumnya. Langkah-langkah yang diperlukan adalah sebagai berikut.

a. Mempersiapkan *virtual kernel file system*.
b. Memasuki lingkungan *chroot*.
c. Membuat pohon direktori sistem.
d. Pembangunan paket-paket *software*.
e. Menjadikan sistem *bootable*.

Virtual kernel file system adalah suatu *file system* yang tidak menggunakan disk space seperti file system pada umumnya, tetapi menggunakan space yang berada di dalam memori. *File system* ini digunakan oleh kernel untuk berkomunikasi ke dan dari kernel itu sendiri.

`Chroot` adalah perintah yang umum dikenal pada sistem operasi berbasis UNIX. Perintah ini digunakan untuk berpindah dari *root* yang sedang aktif ke *root* yang dituju. Dalam hal ini, `chroot` digunakan untuk memasuki *root temporary system*.

```
chroot "$LFS" /tools/bin/env -i \
      HOME=/root            \
      TERM="$TERM"          \
      PS1='\u:\w\$ '        \
      PATH=/bin:/usr/bin:/sbin:/usr/sbin:/tools/bin \
      /tools/bin/bash --login +h
```

Figure 4 Perintah *chroot*

Opsi *–i* diberikan pada perintah *env* agar membersihkan seluruh variabel yang sudah diatur di lingkungan *shell* yang bekerja sebelum menjalankan *chroot*. Kemudian hanya variable `HOME`, `TERM`, `PS1` dan `PATH` yang akan diatur kembali. Perintah *chroot* di atas akan mengakibatkan *shell* bekerja pada *root* yang berada di variabel `$LFS`, yaitu direktori `/mnt/lfs`. Perhatikan bahwa `/tool/bin` disertakan terakhir pada variabel PATH. Maksudnya adalah *temporary tool* tidak akan digunakan lagi ketika sistem *LFS* telah selesai dibangun.

Setelah memasuki lingkungan *temporary system,* hal pertama yang dilakukan adalah pembuatan pohon direktori. Pohon ini adalah struktur penyimpanan direktori utama dari sistem *LFS*. Pada umumnya pohon direktori pada sistem operasi berbasis UNIX menggunakan standar *FHS*. Begitu juga pada pohon direktori sistem *LFS* ini.

Selanjutnya seluruh paket *software* sistem *LFS* dibangun melalui *temporary system*. Prosedur yang digunakan untuk membangun paket software ini hampir sama dengan prosedur yang digunakan pada pembangunan paket *software temporary system*.

Jika seluruh paket telah selesai dibangun langkah berikutnya adalah melakukan konfigurasi agar sistem *bootable*, aritnya adalah sistem dapat melakukan proses *booting*. *Booting* adalah proses memasuki sistem operasi pada saat komputer pertama kali dihidupkan. Proses ini sangat penting karena walaupun sistem telah berhasil dibangun, akan menjadi percuma jika sistem tidak dapat digunakan. Konfigurasi dilakukan pada *file* `/etc/fstab`, *init script* dalam `/etc/rc.d/`, `/etc/module.d/`, dan `/boot/grub/grub.cfg`.

Tahapan ini cukup sulit dilakukan. Namun jika seluruh proses pembangunan dilakukan dengan benar, maka sistem akan berfungsi dengan baik.

3.3 Pemasangan Paket *Software* Tambahan

Pemasangan paket *software* tambahan dilakukan setelah sistem inti *LFS* telah berhasil dibangun. Tahap ini dilakukan untuk memberikan fitur tambahan pada sistem *LFS*, seperti dukungan sistem keamanan, *file system*, *general library*, *general utility*, *system utility*, *programming*, *networking*, *multimedia*, dan *GUI (Graphical User Interface)*. Masing-masing fitur tersebut dapat terdiri dari banyak paket *software*. Prosedur yang dilakukan untuk kompilasi paket-paket *software* ini hampir sama dengan prosedur untuk kompilasi paket-paket *software* sistem *LFS*.

3.4 Pengujian

Berikut ini adalah hasil dari pengujian yang dilakukan terhadap sistem operasi yang telah berhasil dibangun.

Table 2 Hasil Pengujian Sistem

No	Kasus	Harapan	Keterangan
1	Boot process	Sistem dapat mendeteksi dan menjalankan seluruh perangkat *hardware* dan *software*	Sukses
2	Login system	User dapat melakukan login ke dalam sistem	Sukses
3	Shell	User dapat menjalankan perintah melalui shell	Sukses
4	GUI	Sistem operasi dapat menampilkan GUI	Sukses
5	Storage	Sistem dapat melakukan baca tulis pada media penyimpanan	Sukses
6	Jaringan	Sistem dapat terhubung ke jaringan melalui kabel atau wireless	Sukses
7	Multimedia	Sistem dapat memutar musik, video, dan mengolah gambar	Sukses
8	Pemrograman	Sistem dapat melakukan kompilasi terhadap source code bahasa pemrograman khususnya C/C++, Perl, Python, dan Java	Sukses

Berdasarkan hasil pengujian tersebut, sistem dapat menjalankan seluruh fungsi yang diuji. Namun pada tampilan GUI, beberapa fungsi belum dapat berfungsi secara baik. Secara keseluruhan sistem operasi ini layak digunakan dalam kehidupan sehari-hari.

4 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, seluruh langkah, tahapan, serta proses dalam membangun sistem operasi Linux dengan metode *LFS* dapat diketahui secara jelas. Sistem operasi yang dihasilkan dapat digunakan dalam kehidupan sehari-hari. *LFS* mengajarkan bagaimana sistem operasi Linux dan program-program di dalamnya bekerja sama dan bergantung satu sama lain. *LFS* memberikan pengetahuan dan pemahaman yang dalam tentang penggunaan sistem operasi Linux. Oleh karena itu, *LFS* dapat menjadi salah satu cara yang efektif untuk mempelajari sistem operasi GNU/Linux.

5 Referensi

[1] Beekmans, G.. *Linux From Scratch*. http://www.linuxfromscratch.org/lfs/downloads/7.2/LFS-BOOK-7.2.pdf (12 November 2012).
[2] Esteve, J.J. & Boldrito, R.S. *GNU/Linux Advanced Administration*. Eureca Media, SL (2009).
[3] Hicks, A.. Slackware Linux Essentials. Slackware Linux, Inc. (2005).
[4] Masrurkhah, A. A., Danesh, A. S. & Taklimi, S. N. G. *A Survey on Implementation of A Linux-based Operating System Using LFS Method*. International Journal of Computer Science Issues **9,** 170-174 (2012).
[5] Silberschatz, A., Galvin, P.B., & Gagne, G. *Operating System Concepts Essentials*. John Wiley & Sons Inc. (2011).
[6] Stalling, W.. *Operating System : Internals and Design Principles*. Prentice Hall (2012).
[7] Stallman, R. M.. *Free Software, Free Society : Selected Essays of Richard M. Stallman*. GNU Press (2002).
[8] Tanenbaum, A.S. *Operating Systems : Design and Implementation*. Prentice Hall (2006).