MENGENAL SISTEM OPERASI LINUX

Titin Winarti

Diterbitkan oleh :
Semarang University Press
Semarang
2009

Perpustakaan Nasional : Katalog dalam Terbitan (KDT)
ISBN : 978-602-9012-10-0

MENGENAL SISTEM OPERASI LINUX
153 halaman + xi

Titin Winarti

Tata Letak : Priyono
Desain sampul : Saiful Hadi

Cetakan I tahun 2009

Penerbit
Semarang University Press
Jl. Soekarno Hatta, Semarang

Kata Pengantar

Buku ini disusun dari keinginan penulis untuk membuat buku pegangan bagi mahasiswa Fakultas Teknologi Informasi dan Komunikasi Universitas Semarang di bidang Sistem Operasi Linux yang mudah dipahami oleh mereka yang sedang belajar dan bermaksud mengenal linux.

Diharapkan pembaca dapat memahami perintah-perintah yang ada dengan mudah, cepat, dan tuntas. Buku ini tidak membahas semua kemampuan Linux yang sangat banyak, karena tujuannya adalah untuk memberikan bekal dasar bagi pembaca yang ingin menggeluti dunia linux.

Mudah-mudahan buku ini dapat dijadikan pegangan, khususnya bagi para mahasiswa. Kepada semua pihak yang telah membantu penulisan dan penerbitan buku ini, kami sampaikan terima kasih. Saran dan kritik untuk penyempurnaan materi buku ini, sangat penulis harapkan. Semoga buku ini dapat memberikan manfaat.

Semarang, Mei 2009

Penyusun

Daftar Isi

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENGANTAR LINUX

1.1. Apakah LINUX itu

LINUX adalah nama sistem operasi yang dapat diterapkan pada berbagai jenis mesin, dari PC hingga mainframe. Linux diciptakan oleh Linus Torvard. Sistem operasi adalah perangkat lunak komputer yang mengatur dan mengendalikan operasi dasar sistem komputer. Linux terdiri atas sejumlah program (daftar instruksi untuk memperoleh hasil tertentu) yang dirancang untuk mengontrol interaksi antara fungsi-fungsi pada mesin yang beraras rendah dengan program aplikasi. Tugas dari sistem operasi diantaranya :

1. Melakukan fungsi manajemen sistem berkas
2. Mengendalikan berbagai sumber pada sistem, seperti disk dan printer
3. Mengatur sejumlah pemakai yang menggunakan sistem bersamaan
4. Membentuk penjadualan proses-proses di dalam sistem

Linux adalah sistem operasi semacam LINUX yang bersifat gratis.

Beberapa sifat dan keistimewaan LINUX

1. Portabilitas

Sistem Linux mudah diadaptasikan ke sistem komputer yang lain. Sifat portabilitas ini membawa Linux dapat dipakai pada berbagai jenis komputer. Kini Linux telah menyebar pada berbagai jenis sistem, dari notebook, mikrokomputer (PC), hingga mainframe. Bagi pemakai, hal seperti ini sangatlah menguntungkan. Mengapa menguntungkan? Sebab portabilitas berarti ketidakbergantungan pada suatu perangkat keras.

2. Multiuser

Berarti sejumlah orang (pemakai) dapat menggunakan sistem secara bersamaan dan berbagai sumber (disk, printer, dan sebagainya).

Keuntungan adanya multiuser :

- Penghematan perangkat keras

Sebab perangkat keras (misalnya *printer, disk*) dapat dipakai oleh orang banyak.

- Data dapat diakses oleh orang banyak secara serentak

Ini berarti tidak ada penduplikasian data. Selain itu konsistensi data lebih terjamin.

3. Multitasking

Seorang pemakai dapat melakukan beberapa pekerjaan dalam waktu yang bersamaan dari sebuah terminal. Pekerjaan-pekerjaan yang tidak memerlukan interaksi dari pemakai (seperti melakukan pengurutan data dan pengecekan kosa kata) bisa dilaksanakan di latar belakang. Pemrosesan ini memungkinkan saat suatu pekerjaan sedang dilaksanakan oleh sistem, pemakai dapat melakukan tugas-tugas yang lain.Kemampuan sistem operasi yang memungkinkan seseorang dapat melaksanakan beberapa tugas pada saat bersamaan dinamakan multitasking.

4. Sistem Berkas Yang Hierarkis

Sistem berkas yang hirarkis memungkin pemakai mengorganisasikan informasi atau data dalam bentuk yang mudah untuk diingat dan mudah untuk mengaksesnya. Informasi-informasi yang ada dapat diatur misalnya dikelompokkan per pemakai atau berdasarkan suatu departemen.

5. Shell LINUX

Shell Linux menjadi jembatan antara pemakai dan sistem. Ia bertindak sebagai penerjemah perintah yang sangat bermanfaat bagi pemakai. Kemampuan shell mencakup dua hal :

1. **Modus interaktif**

   Pada mode ini, pemakai dapat memberikan perintah dan kemudian shell akan mengerjakan perintah yang diberikan. Hal ini dapat diulang-ulang. Sebab begitu shell telah selesai menjalankan perintah, shell akan menunggu pemakai memberikan perintah kembali.

2. **Modus pemrograman**

Pada modus ini, pemakai dapat menyusun suatu program yang berupa sejumlah perintah yang biasa disebut *script* shell. Selanjutnya, shell akan mengerjakan perintah-perintah tersebut secara berurutan. Hal seperti ini sangat bermanfaat untuk menangani pekerjaan yang bersifat rutin.Pada modus ini pemakai dapat membuat suatu prototipe suatu kegiatan tanpa harus menggunakaan bahasa pemrograman seperti C.

Baik pada mode interaktif maupun pemrograman, pemakai dapat dengan mudah mengarahkan keluaran-keluaran perintah yang normalnya ke layar menjadi suatu berkas. Bahkan pemakai dapat juga mengatur agar hasil suatu perintah menjadi masukan bagi perintah yang lain.

6. Utilitas

Sistem operasi LINUX tersusun atas sejumlah program, yang antara lain berupa utilitas. Utilitas-utilitas yang tersedia pada LINUX mempunyai tugas yang bermacam-macam, antara lain berhubungan dengan :

- Manajemen berkas
- Penyunting berkas
- Pendukung komunikasi
- Pendukung pengembangan perangkat lunak

Dengan mengkombinasikan utilitas-utilitas yang ada, pemakai dapat membuat program baru untuk melaksanakan tugas seperti yang diharapkan. Hal ini dapat dilakukan dengan cepat dan mudah.

1.2. Masyarakat LINUX

Semula pemakaian LINUX terbatas pada kalangan tertentu terutama lingkungan universitas, kini LINUX juga banyak dipakai untuk menangani aplikasi bisnis. LINUX juga seringkali digunakan untuk memecahkan persoalan yang kompleks pada permasalahan statistik dan *engineering.*

Perkembangan LINUX juga tidak lepas dengan lembaga-lembaga atau organisasi-organisasi yang melakukan standarisasi. Antara lain :

- ANSI, menyediakan standar pemrograman C.
- POSIX, mengenai kernel LINUX.
- X/OPEN, mendefinisikan lingkungan untuk mendukung portabilitas perangkat lunak.

- ISO, ikut megembangkan beberapa standar, terutama dalam lingkup komunikasi computer.
- X-Consurtium, membuat pedoman antarmuka yang berdasarkan teknologi X-Window.
- AT-T, mengeluarkan standar sistem V yang disebut SVID. Standar ini menyebutkan fasilitas-fasilitas pada LINUX yang dijamin tidak bakal berubah pada rilis-rilis mendatang.

1.3. Sejarah LINUX

Linux pada awalnya dibuat oleh seorang mahasiswa Finlandia yang bernama Linus Torvalds. Dulunya Linux merupakan proyek hobi yang diinspirasikan dari Minix, yaitu sistem UNIX kecil yang dikembangkan oleh Andrew Tanenbaum. Linux versi 0.01 dikerjakan sekitar bulan Agustus 1991. Kemudian pada tanggal 5 Oktober 1991, Linus mengumumkan versi resmi Linux, yaitu versi 0.02 yang hanya dapat menjalankan shell bash (GNU Bourne Again Shell) dan gcc (GNU C Compiler).

Saat ini Linux adalah sistem UNIX yang sangat lengkap, bisa digunakan untuk jaringan, pengembangan software dan bahkan untuk pekerjaan sehari-hari. Linux sekarang merupakan alternatif sistem operasi yang jauh lebih murah jika dibandingkan dengan sistem operasi komersial (misalnya Windows 9.x/NT/2000/ME).

Linux mempunyai perkembangan yang sangat cepat. Hal ini dapat dimungkinkan karena Linux dikembangkan oleh beragam kelompok orang. Keragaman ini termasuk tingkat pengetahuan, pengalaman serta geografis. Agar kelompok ini dapat berkomunikasi dengan cepat dan efisien, internet menjadi pilihan yang sangat tepat.

Karena kernel Linux dikembangkan dengan usaha yang independent, banyak aplikasi yang tersedia, sebagai contoh, C Compiler menggunakan **gcc** dari **Free Software Foundation** GNU’s Project. Compiler ini banyak digunakan pada lingkungan Hewlett-Packard dan Sun.

Banyak aplikasi Linux yang dapat digunakan untuk keperluan kantor seperti untuk spreadsheet, word processor, database dan program editor grafis yang memiliki fungsi dan tampilan seperti Microsoft Office, yaitu Star Office. Selain itu, juga sudah tersedia versi Corel untuk Linux dan aplikasi seperti Matlab yang pada Linux dikenal sebagai Scilab.

1.4. Berbagai Versi LINUX

- **RedHat**, distribusi yang paling populer, minimal di Indonesia. RedHat merupakan distribusi pertama yang instalasi dan pengoperasiannya mudah.
- **Debian**, distribusi yang mengutamakan kestabilan dan kehandalan, meskipun mengorbankan aspek kemudahan dan kemutakhiran program. Debian menggunakan .deb dalam paket instalasi programnya.
- **Slackware**, merupakan distribusi yang pernah merajai di dunia Linux. Hampir semua dokumentasi Linux disusun berdasarkan Slackware. Dua hal penting dari Slackware adalah bahwa semua isinya (kernel, library ataupun aplikasinya) adalah yang sudah teruji. Sehingga mungkin agak tua tapi yang pasti stabil. Yang kedua karena dia menganjurkan untuk menginstall dari source sehingga setiap program yang kita install teroptimasi dengan sistem kita. Ini alasannya dia tidak mau untuk menggunakan binary RPM dan sampai Slackware 4.0, ia tetap menggunakan libc5 bukan glibc2 seperti yang lain.
- **SuSE**, distribusi yang sangat terkenal dengan YaST (Yet another Setup Tools) untuk mengkonfigurasi sistem. SuSE merupakan distribusi pertama dimana instalasinya dapat menggunakan bahasa Indonesia.
- **Mandrake**, merupakan varian distro RedHat yang dioptimasi untuk pentium. Kalau komputer kita menggunakan pentium ke atas, umumnya Linux bisa jalan lebih cepat dengan Mandrake.
- **WinLinux,** distro yang dirancang untuk diinstall di atas partisi DOS (WIndows). Jadi untuk menjalankannya bisa di-klik dari Windows. WinLinux dibuat seakan-akan merupakan suatu program aplikasi under Windows.

1.5. Latihan Soal

1. Linux adalah nama sebuah ..... seperti halnya Windows dan MS-DOS
2. ..... adalah istilah yang menunjukkan sejumlah orang dapat memakai sebuah sistem dan berbagi peranti seperti disk dan printer.
3. Sifat ..... dari Linux membuat Linux dengan mudah dapat diimplementasikan pada berbagai jenis mesin
4. ..... adalah organisasi yang berperan dalam menentukan standar pemrograman C

5. Badan yang pertama kali menentukan standar sistem operasi yang portabel adalah.....
6. Bahasa yang digunakan untuk membuat Linux.....
7. Nama pencipta Linux adalah .....

BAB VII
OPERASI BERKAS

7.1.Cara Menamakan Berkas

Nama berkas pada LINUX umumnya dibatasi hingga 14 karakter. Namun, pada beberapa sistem dapat mencapai 256 karakter.

Pedoman dalam menciptakan nama berkas adalah sebagai berikut:

- Karakter yang biasa di pakai untuk menyusun berkas adalah:
  A hingga Z
  a hingga z
  0 hingga 9
  .(titik)
  _(garis bawah)
  Meskipun karakter-karakter lain secara teknis dapat digunakan,sebaiknya di hindari.
- Tidak boleh menggunakan nama berkas yang hanya berupa titik atau dua titik,sebab kedua simbol tersebut mempunyai arti tersendiri .
- Jika suatu nama berkas berawalan dengan titik, berkas akan bersifat tersembunyi (*hidden berkas*). Berkas seperti ini tidak akan terlihat kalau perintah Is diberikan.(tanpa pilihan –a)
- Huruf kecil dan huruf capital mempunyai makna yang berbeda pada pemberian nama berkas.
- Untuk mempermudah dalam mengingat, nama berkas biasanya diberi suatu indikasi yang menyatakan isi berkas. Sebagai contoh, berkas yang berisi program C sebaiknya menggunakan akhiran .c.
  Akhiran-akhiran yang umum digunakan diantaranya:

.pas	=	Program PASCAL
.cbl	=	Program COBOL
.c	=	Program C
.dat	=	Berkas Data
.bas	=	Program BASIC

  Contoh:

- Contoh nama berkas yang berlaku:

  urut.c

  masakan

  readme.txt

  kuartal1_2

  .salam (berkas tersembunyi)

- Contoh yang salah atau tidak di anjurkan:

  apa? (karakter ? perlu dihindari)

  .. (nama ini milik sistem,yang berarti direktori induk)

  star*.kecil (karakter * perlu di hindari)

7.2. Menciptakan Berkas

Sebuah berkas dapat diciptakan dengan berbagai cara. Misalnya,dengan menyalin berkas ataupun menciptakan melalui editor teks. Sebuah berkas (khususnya berkas teks) juga dapat dibuat dengan menggunakan perintah cat. Caranya sebagai berikut:

1. Ketik perintah cat . nama_berkas dan tekan <ENTER>.
2. Ketikkan apa saja yang anda inginkan. Anda dapat menekan <ENTER> untuk berpindah baris.
3. Akhiri dengan memberikan kode special eof, yang digunakan untuk menyatakan akhiri berkas.

Sebagai Contoh:

```
$ cat > file1.txt
Sedikit demi sedikit
Lama-lama menjadi bukit
<Ctrl-D>
$
```

Pada contoh ini, tanda > merupakan simbol pengalihan arah keluaran. Setelah <Ctrl-D>/kode eof diberikan, berkas file.txt akan tercipta. Isinya:

Sedikit demi sedikit
Lama lama menjadi bukit

Untuk membuat berkas kosong (berkas yang tidak berisi apa-apa). Anda dapat memakai utilitas touch. Contoh:

```
$touch file2.txt
$_
```

Akan menciptakan berkas kosong bernama file2.txt.

Perintah	:cat
Perintah	:cat
Kategori	:Utilitas LINUX
Fungsi	:Untuk melihat isi sebuah atau beberapa berkas teks. Bisa juga dimanfaatkan untuk menciptakan berkas teks.
Format	:1.cat berkas… 2.cat > berkas
Hasil	:Pada format pertama, isi berkas ditampilkan di layar. Pada format kedua,a kan tercipta berkas dengan nama berkas. Seluruh data yang diketik dari keyboard hingga kode <eof> diberikan akan menjadi isi dari berkas.

Gambar 7.1 Perintah cat

7.3.Melihat Isi Berkas

Isi dari suatu berkas teks dapat dilihat dengan menggunakan utilitas cat. Sebagai contoh, isi berkas file.txt yang terbentuk pada contoh sebelumnya dapat dilihat dengan cara memberikan perintah seperti dibawah ini.

```
$ cat file1.txt
```

Sedikit demi sedikit
Lama-lama menjadi bukit

$_

Utilitas cat juga dapat digunakan untuk melihat isi beberapa berkas sekaligus. Sebagai contoh:

cat /etc/passwd /etc/group

Akan menampilkan isi berkas /etc/paswwd dan /etc/group. Apabila isi berkas sangat panjang. Anda perlu memberikan kode stop <Ctrl-s> dan start <Ctrl-Q> bergantian agar isi berkas dapat anda baca.

Selain menggunakan cat, anda juga bias menggunakan utilitas pg (berasal dari kata “page”,yang artinya halaman). Perintah pg sangat bermanfaat untuk melihat berkas yang sangat panjang. Sebab penampilanisi berkas akan diatur perlayar. Setiap tampilan layar disebut halaman.

Perintah	:	pg
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk menampilkan isi berkas per layar
Format	:	pg berkas…
Hasil	:	jika isi berkas melebihi satu layar, isinya akan ditampilkan per layar

Gambar 7.2. Perintah pg

Setiap kali sesudah sebuah layar ditampilkan, pg memungkinkan pemakai untuk :

- Mengakhiri tampilan berkas
- Melanjutkan ke halaman berikutnya atau kembali ke halaman sebelumnya
- Mundur atau maju beberapa halaman

Beberapa perintah yang dapat diberikan ke pg sewaktu perintah ini menunggu tanggapan dari pemakai (berupa munculnya *prompt* :) dapat dilihat pada tabel 7.1.

Tabel 7.1 Perintah Tanggapan Pada pg

Perintah	Keterangan
<Enter>	Ke halaman berikutnya
<Spasi>	Ke halaman berikutnya
i<Enter>	(i berupa bilangan) ke halaman i
+*i*<Enter>	(i berupa bilangan) ke i halaman berikutnya
-*i*<Enter>	(i berupa bilangan) ke i halaman sebelumnya
$<Enter>	Ke halaman terakhir
h<Enter>	Menampilkan bantuan (informasi) perintah
n<Enter>	Ke berkas berikutnya (untuk berkas berganda)
p<Enter>	Ke berkas sebelumnya (untuk berkas berganda)
q<Enter>	Keluar dari pg

Catatan : Linux tidak menyediakan utilitas pg.

Pada LINUX juga dapat perintah dengan dengan fungsi serupa pg, yaitu more.

Perintah	:	more
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk menampilkan isi berkas per layar
Format	:	more berkas…
Hasil	:	jika isi berkas melebihi satu layar, isinya akan ditampilkan Perlayar

Gambar . 7.3 Perintah more

Untuk setiap akhir halaman,more akan menampilkan promt berupa --more—

Diikuti dengan presentase isi berkas hingga tampilan sekarang. Adapun beberapa perintah yang dapat diberikan saat *prompt* tersebut muncul dapat dilihat pada Tabel 7.2.

Tabel 7.2 Perintah Tanggapan Untuk *more*

Perintah	Keterangan
<Enter>	1 baris berikutnya akan ditampilkan
<Spasi>	Ke halaman berikutnya
b	Ke halaman sebelumnya
*n*b	(dengan n berupa bilangan) ke n halaman sebelumnya

h atau ?	Informasi bantuan mengenai perintah
q atau Q	Keluar dari more

7.4.Nama Berkas Bermakna Ganda

Beberapa perintah seperti **cat**, **pg** dan **more** dapat menerima argumen berupa beberapa berkas. Pada perintah-perintah seperti ini, anda dapat menggunakan simbol-simbol khusus untuk membentuk sebuah nama berkas bermakna ganda,yang dapat menyatakan satu atau beberapa berkas. Cara seperti ini dapat menghemat penulisan argument, terutama kalau melibatkan sejumlah berkas yang namanya mempunyai kesamaan pola.

Simbol-simbol yang menyusun nama berkas bermakna ganda (biasa disebut *wildcard*) dapat dilihat di tabel 6.3.

Tabel 7.3 Simbol Khusus Pembentuk Nama Berkas Berganda

Simbol	Keterangan
*	Sebuah simbol * berarti cocok dengan nol, satu, atau beberapa karakter apa saja.
?	Sebuah simbol * berarti cocok dengan sebuah karakter apa saja.
[ ]	Simbol [ ] secara berpasangan menyatakan cocok dengan sebuah karakter terdapat di dalam tanda kurung tersebut.
[-]	Simbol minus (-) di dalam tanda [] menyatakan jangkauan. Misalnya [a-e] berarti a,b,c,d atau e.
[!]	Simbol tanda seru (!) didalam [ ] dan terletak sesudah simbol [berarti yang cocok dengan karakter apa saja selain yang mengikuti tanda !. (Simbol ini tidak di kenal pada C shell).

Contoh nama berkas bermakna ganda:

- * semua berkas pada direktori kerja
- *.* semua berkas pada direktori kerja yang mengandung sebuah titik.
  Contoh yang cocok : latihan.c a.bcde data.bulan.ini
  Contoh yang tidak cocok : .profile hai
- .* Sebuah berkas tersembunyi (berkas yang berawalan titik)

- Berkas?.txt semua berkas yang diawali dengan berkas diikuti dengan sebuah karakter apa saja dan kemudian berakhiran.txt
- *[xyz] semua berkas yang berakhiran dengan x, y atau z
- ?? semua berkas pada direktori kerja
- [!f]* semua berkas yang tidak berawaln f.

Penerapan nama berkas bermakna ganda akan banyak dijumpai sesudah subbab ini.Misalnya pada proses:

- Menyalin sejumlah berkas
- Memindahkan sejumlah berkas ke suatu direktori
- Menghapus sejumlah berkas

7.5.Melihat Informasi Berkas

Perintah yang digunakan untuk menampilkan isi direktori telah diperkenalkan pada bab terdahulu. Perintah ini yaitu **Is** (berasal dari kata "list"). Perintah ini mempunyai sejumlah pilihan. Beberapa diantaranya ditunjukkan pada gambar 6.2

Perintah	:	ls
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk melihat informasi berkas
Format	:	Is [pilihan] [berkas…]
		Beberapa pilihan:

- -a Menampilkan seluruh berkas, termasuk berkas tersembunyi.
- -A Serupa pilihan –a, tetapi berkas dengan nama . dan .. tidak disertakan.
- -d Menampilkan informasi dari nama direktori, bukan berkas yang terkandung di
- dalamnya.
- -F Identifikasi dari setiap berkas akan diberikan.
- -i Bilangan *inode* akan ditampilkan.
-
- -I (Huruf I) Setiap berkas akan ditampilkan dalam sebuah baris dengan disertai informasi antara lain permisi akses, pemilik berkas dan ukuran berkas.
- -r Nama berkas akan diurutkan secara terbalik, dari Z ke A.
- -R Secara rekursif menampilkan isi seluruh subdirektori.

- -t Nama berkas akan ditampilkan berdasarkan yang terbaru.

Hasil : Hasi dari perintah Is ditampilkan ke layar.

Gambar 7.4 Perintah ls

Perintah Is tanpa Argumen dan Pilihan

Kalau perintah **ls** diberikan tanpa pilihan dan argumen, nama-nama berkas (tanpa informasi yang lain) pada direktori kerja yang ditampilkan.

```
$ ls
Blank file1.txt file2.txt kosong passwd
$_
```

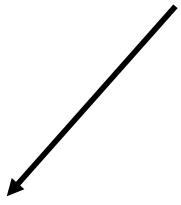
Menampilkan dengan format Panjang

Untuk melihat informasi yang merupakan atribut dari berkas, pilihan –I (huruf I, bukan angka 1) perlu disertakan.Contoh:

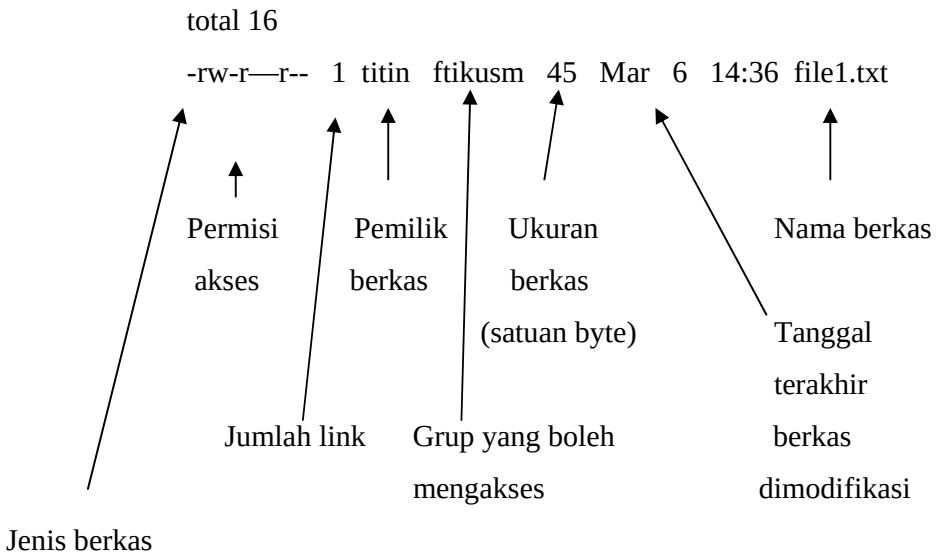
```
$ ls   -l
Total 8
-rw-r—r--   1 titin    ftikusm     0 Mar  6   14:29  blank
-rw-r—r--   1 titin    ftikusm    45 Mar  6   14:36  file1.txt
-rw-r—r--   1 titin    ftikusm    0 Mar  6   14:37  file2
-rw-r—r-- 1 titin   ftikusm    0 Mar 6  14:29 kosong
$_
```

Gambar 1.6 memberikan penjelasan terhadap atribut berkas, hasil dari pilihan –I.

Total entri direktori yang ditampilkan dalam satuan blok (1 blok = 512 byte).



Total entri direktori yang di tampilkan dalam satuan blok (1 blok =512 byte)



Gambar 7.5 Informasi berkas format panjang

Tabel 7.4 Kode Jenis/Tipe Berkas

Simbol	Keterangan
-	Berkas Biasa
d	Direktori
b	Peranti blok(block device)
c	Peranti Karakter(character device)
I	Simbolic link
m	Shared memory(sistem v)
p	FIFO /named pipe (sistem v)
s	Socket/semaphore

Menyertakan Berkas Tersembunyi

Agar berkas-berkas tersembunyi ikut ditampilkan, anda dapat ,menambahkan pilihan –al. Sebagai contoh:

$ ls –al

Total 44

drwx------	3 titin	ftikusm	4096	Mar	6	14:27	.
drwxr-xr-x	5 root	root	4096	Mar	6	2002	..
-rw-r—r--	1 titin	ftikusm	263	Mar	6	14:28	.bash_history

-rw-r—r--	1 titin	ftikusm	24	Mar	6	2002	.bahs_logout
-rw-r—r--	1 titin	ftikusm	191	Mar	6	2002	.bahs_profile
-rw-r—r--	1 titin	ftikusm	124	Mar	6	14:29	.bashrc
-rw-r—r--	1 titin	ftikusm	0	Mar	6	2002	blank
-rw-r—r--	1 titin	ftikusm	820	Mar	6	14:36	.emacs
-rw-r—r--	1 titin	ftikusm	45	Mar	6	14:37	file1.txt
-rw-r—r--	1 titin	ftikusm	0	Mar	6	2002	file2
drwxr-xr-x	3 titin	ftikusm	4096	Mar	6	14:27	.kde
-rw-r—r--	1 titin	ftikusm	0	Mar	6	14:29	kosong
-rw-r—r--	1 titin	ftikusm	1428	Mar	6	14:29	paswwd
-rw-r—r--	1 titin	ftikusm	3511	Mar	6	2002	.screenrc

$_

Berkas-berkas seperti .bash_profile (yang berawalan titik) adalah berkas-berkas tersembunyi yang tidak akan terlihat kalau pilihan –a (atau –A) tidak di sertakan. Berkas dengan nama . dan .. tidak disertakan kalau pilihan –A diberikan. Tetapi berkas seperti .bash_profile akan ditampilkan.

Melihat Informasi Berkas tertentu

Untuk melihat informasi dari sebuah berkas. Anda dapat menambahkan argument berupa nama berkas. Contoh:

```
$ ls  –l  file 1.txt
-rw-r–r--   1 titin ftikusm 45 Mar  6 14:36     file1.txt
$_
```

Untuk melihat isi berkas-berkas pada suatu direktori yang bukan berupa direktori kerja, Anda dapat menambahkan argumen berupa nama direktori. Contoh untuk menampilkan isi direktori/etc:

$ ls –l/etc

Total 1472

-rw-r—r--	1 root	root	15223	Jun	25	2001	a2ps.cfg
-rw-r—r--	1 root	root	2561	Jun	25	2001	a2ps-site.cfg
-rw-r—r--	1 root	root	12	Sep	9	14:10	adjtime
drwxr-xr-x	4 root	root	1024	Mar	6	2002	alchemist
-rw-r—r--	1 root	root	1048	Aug	31	2001	aliases

-rw-r—r-- 1 root root 12288 Mar 6 13:54 aliases.db
-rw-r—r-- 1 root root 370 Jun 25 2002 anacrontab
-rw-------- 1 root root 1 Aug 3 2001 at.deny
<dan seterusnya>
$_

Menampilkan Informasi Direktori Saja

Kalau yang ingin dilihat adalah informasi dari suatu direktori, tetapi bukan berkas-berkas yang ada di dalamnya, anda perlu menambahkan pilihan –d selain –I. Contoh:

$ ls –ld/etc
drwxr-xr-x 45 root root 4096 Mar 6 14:28 /etc
$_
Tampak hasil berbeda dengan ls –l/etc.

Mengidentifikasi Berkas

Jika pilihan –F diberikan, Is akan menandai masing-masing berkas dengan kode yang tertera pada Tabel 1.5. Berikut adalah contoh hasil **ls –aF**.

$ ls -aF
./
../
.bash_history
.bahs_logout
.bahs_profile
.bashrc
blank
.emacs
file2
.kde

kosong

simbol@

telpon

$_

Perhatikan bahwa pada contoh di atas dapat terdapat simbol.Seperti / dan @ yang terletak di bagian akhir.Simbol-simbol ini mempunyai makna khusus sebagaimana terlihat pada tabel berikut.

Tabel 7.5 Simbol Hasil Perintah ls dengan Pilihan –F.

Simbol	Keterangan
(blank)	Berkas Biasa
*	Program executable
/	Direktori
=	Socket
@	Simbolic link

Pemakaian Wildcard

Contoh sebagai berikut menunjukkan pemakaian *wildcard* pada argumen ls.

```
$ ls      /bin/[bc]*
/bin/basename
/bin/bash
/bin/bash2
/bin/bsh
/bin/cat
/bin/chgrp
/bin/chmod
/bin/chown
/bin/consolechars
/bin/cp
/bin/cpio
/bin/csh
/bin/cut
```

$_

Perintah di atas menampilkan semua berkas pada direktori /bin yang berawalan b atau c.

7.6. Menyalin Berkas

Untuk keperluan menyalin suatu berkas, LINUX menggunakan utilitas bernama **cp** (berasal dari kata copy), berkas yang akan disalin biasa disebut sebagai berkas sumber dan berkas hasil disebut berkas target.

Perintah	:	**cp**
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk menyalin berkas
Format	:	1. cp *berkas_sumber berkas targer* 2. cp *berkas…dir*
Hasil	:	Pada format pertama, *berkas_target* akan terbentuk dan merupakan salinan dari *berkas_sumber* Pada format kedua, sejumlah berkas didepan *dir* akan disalin ke direktori *dir*.

Gambar 7.6 *perintah cp*

Sebagai contoh :

```
$ cp file1.txt filebaru.txt
$_
```

Perintah diatas akan membentuk berkas bernama filebaru.txt yang merupakan salinan dari file1.txt. Isi kedua berkas tersebut sama, sebagaimana ditunjukkan pada contoh berikut :

```
$ cat file1.txt
sedikit demi sedikit
lama-lama menjadi bukit

$ cat filebaru.txt
sedikit demi sedikit
```

lama-lama menjadi bukit

$_

Seandainya, berkas sumber tidak ada, pesan kesalahan akan ditampilkan.

Contoh :

$ cp coba_ah hasil

cp coba_ah : No such file or directory

$_

Pada keadaan seperti ini, berkas hasil tidak terbentuk.

Kalau berkas target sudah ada, kadang akan ada konfirmasi, tetapi hal ini tergantung pada implementasi **cp**. Contoh :

```
$ cp   /etc/passwd    filebaru.txt
```

akan menyebabkan filebaru.txt barisi salinan dari /etc/passwd, bukan lagi berisi salinan dari file1.txt. Bila tidak ada konfirmasi, isi berkas target semula akan ditindih.

Perlu diketahui, pada penyalinan berkas :

- Permisi berkas salin ditentukan oleh *umask*. Jadi bisa saja permisi akses antara berkas asal dan berkas hasil berbeda.
- Pemilik berkas salinan dan grup yang tercatat pada berkas salinan adalah sesuai dengan nama pemakai yang melakukan penyalinan.

Untuk melihat hal ini, Anda dapat melihat berkas asal dan berkas salinan hasil perintah **cp** di atas.

$ ls –l /etc/passwd filebaru.txt

-rw-r--r-- 1 root root 1428 Mar 6 14:26 /etc/passwd

-rw-r--r-- 1 kadit ftikusm 1428 Mar 6 14:30 filebaru.txt

$-

perhatikan, bagian permisi akses, nama pemilik dan grupnya. Ada perbedaan antara berkas sumber dan berkas hasil salinan.

Menyalin Sejumlah Berkas

Perintah **cp** juga mempunyai versi yang lain, berbentuk :

cp *file1 file2 file3 … dir*

Pada bentuk seperti ini, file1, file2, file3 dan seterusnya akan disalin ke direktori bernama dir. Pada bagian berkas, Anda juga dapat menggunakan *wildcard*.

Catatan:

LINUX juga mempunyai perintah bernama copy, yang dapat dipakai untuk menyalin isi suatu direktori ke deroktori lain.

7.7. Menghapus Berkas

Menghapus sebuah berkas atau beberapa berkas dapat dikerjakan dengan menggunakan utilitas **rm** (berasal dari kata *remove*). Format perintah ini dapat dilihat pada gambar dibawah ini.

Perintah	:	**rm**
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk menghapus berkas
Format	:	rm [pilihan] *berkas* …
		Pilihan :
		Diantaranya berupa :
		• -i rm akan meminta konfirmasi dari pemakai sebelum menghapus
Hasil	:	Tanpa pilihan –i, berkas yang menjadi argumen rm akan dihapus tanpa meminta persetujuan dari pemakai..

Gambar 7.7 perintah rm

Sebagai contoh, Anda dapat menghapus berkas filebaru.txt melalui perintah :

```
$ rm filebaru.txt
rm: remove ‘filebaru’? y<ENTER>

$ ls filebaru.txt
ls: filebaru.txt: no such file or directory
$_
```

Pesan kesalahan dari perintah **ls** di atas menunjukkan filebaru.txt sudah tidak ada lagi.

Menghapus Sejumlah Berkas

Untuk menghapus sejumlah berkas, Anda dapat menggunakan nama berkas bermakna ganda. Contoh :

$ rm berkas[bk]*

Akan menghapus berkas yang berawalan b atau k.

7.8. Mengganti Nama Berkas

Nama suatu berkas yang sudah terbentuk dapat diganti dengan menggunakan utilitas **mv** (besrasal dari kata move). Sebagai contoh, diinginkan untuk mengganti nama file1.txt menjadi pepatah.txt. perintahnya :

```
$ mv file1.txt pepatah.txt
$_
Anda dapat memeriksa hasilnya :


$ ls file1.txt pepatah.txt
ls: file.txt: no such file or directory pepatah.txt
$_
```

Tampak bahwa file1.txt tidak ada.

Sebagai gantinya adalah berkas pepatah.txt.

Perintah	:	**mv**
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk mengganti nama berkas / direktori atau untuk memindahkan berkas ke direktori lain
Format	:	1. mv *file1 file2* 2. mv *file1…dir*
Hasil	:	Pada format pertama, *nama file1 diganti file2*

	Pada format kedua, *file1* dan seterusnya akan dipindahkan ke direktori *dir*.

Gambar 7.8 perintah mv

Perintah **mv** juga dapat dipakai untuk memindahkan suatu berkas ke direktori lain.

7.9 Mengidentifikasi Berkas

Suatu berkas dapat diidentifikasi dengan menggunakan perintah **berkas** (utilitas). Perintah ini dapat melaporkan jenis berkas. Walaupun laporan yang dihasilkan tidak selalu akurat, perintah ini sangat bermanfaat untuk mengetahui suatu berkas berupa text atau bukan.

Perintah	:	**file**
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk memperoleh gambaran isi dari suatu berkas
Format	:	berkas *nama_berkas*
Hasil	:	Informasi tentang jenis berkas pada layar.

Gambar 7.9. Perintah file

contoh :

```
$ berkas/etc/passwd
/etc/passwd: ASCII text
$_
```

Hasil diatas menyatakan bahwa /etc/passwd adalah berkas text ASCII.

Untuk melihat berbagai laporan yang dihasilkan oleh perintah **berkas**, Anda dapat mencoba perintah berikut :

$ berkas /etc/*
/ect/DIR_COLORS: ASCII English text
/ect/Muttrc: ASCII English text

/ect/x11: directory
/ect/a2ps-site.cfg: ASCII English text
/ect/a2ps.cfg: ASCII English text
/ect/adjtime: ASCII text
/ect/alchemist: directory
/ect/aliases: ASCII English text
/ect/aliases.db: Berkele DB (Hash,version 7,native byte-order)
/ect/anacrontab: ASCII text
/ect/at.deny: can’t read
/ect/at.deny:
/ect/auto.master: ASCII English text
/ect/auto.misc: ASCII English text
/ect/bashrc: ASCII text
/ect/cdrecord.conf: ASCII English text
/ect/cipe: directory
/ect/cron.d: directory
/ect/cron.daily: directory
/ect/cron.hourly: directory
/ect/cron.monthly: directory
/ect/cron.weekly: directory
/ect/crontab: ASCII text
/ect/csh.cshrc: ASCII text
/ect/csh.login: ASCII text
/ect/default: directory
/ect/dhcpc: directory
/ect/dhcpcd: simbolic link to dhcpc
/ect/dumpdates: empy
/ect/esd.conf: ASCII text
/ect/exports: empty
/ect/fam.conf: ASCII English text
<dan seterusnya>
$_

Informasi yang ditampilkan antara lain menyatakan berkas kosong (*empety*), script shell (*Bourne/Korn shell command text*), dan teks ASCII (*ascii text*), sedangkan

informasi *permission danied* atau *cannot open for reading* menyatakan bahwa pemakai tidak mempunyai permisi untuk membaca berkas tersebut.

Untuk mengidentifikasi semua berkas pada direktori kerja, perintah :

file *

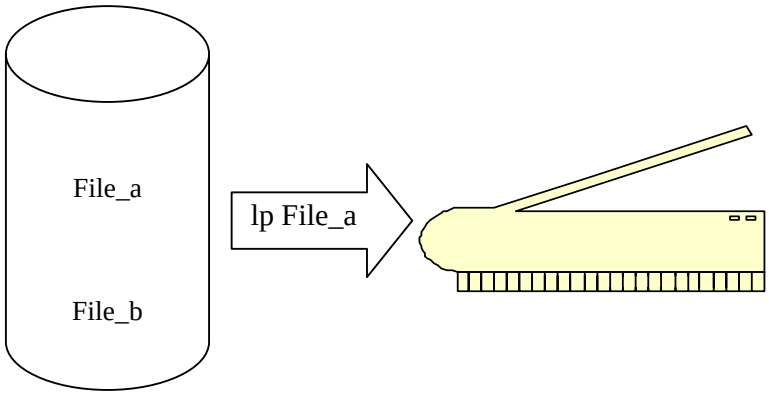
Dapat dipakai.

7.10 Mencetak Berkas

Apabila anda menginginkan suatu informasi tercetak ke printer, anda dapat memakai utilitas **lp** (berasal dari kata line printer). Apabila sistem memiliki printer yang dijadikan sebagai *default*, perintah lp dapat berbentuk :

`lp` *`namefile`* *…*

Apabila sistem memiliki beberapa printer, anda dapat mengarahkan ke suatu printer dengan memberikan pilihan –d diikuti dengan nama pengenal printer.



Gambar 7.10. Perintah lp

Catatan: Pada beberapa sistem, perintah untuk mencetak berkas ke printer bukan berupa lp, melainkan berupa lpr.

7.11. Membuat Link

Link adalah pointer (petunjuk) dari sebuah berkas yang menunjukkan ke *inode*. Pada LINUX, sebuah *inode* dapat dimiliki oleh lebih dari sebuah berkas. Hal

ini memungkinkan suatu berkas diberi nama lebih dari satu. Dalam hal ini, jumlah nama berkas yang menunjukkan ke sebuah *inode* disebut jumlah *link*.

Kegunaan *link* adalah agar suatu berkas dapat diacu dalam sejumlah direktori yang berbeda. Hal ini memungkinkan data yang sama dapat digunakan oleh sejumlah pemakai. *Link* lebih disukai daripada penyalinan disebabkan penyalinan berarti pemduplikasian data. Dengan menggunakan *link*, ruang disk untuk data tidak perlu dua kali. Untuk membuat *link*, LINUX menyediakan utilitas bernama **ln** (berasal dari kata *link*). Format perintah **ln** dapat dilihat pada gambar 1.11.

Perintah	:	**ln**
Kategori	:	Utilitas LINUX
Fungsi	:	Untuk membuat suatu *link* terhadap suatu berkas
Format	:	ln [pilihan] *sumber target* … Pilihan : Diantaranya berupa : • -s = *simbolic link*
Hasil	:	Jika pilihan -s tidak diberikan, akan terbentuk berkas dengan nama *targer* yang menunjuk ke *inode* yang ditunjuk oleh *sumber* Jika pilihan -s diberikan, akan terbentuk *simbolic link* (bukan *hard link*).

Gambar 7.11. Perintah ln

Untuk melihat efek perintah ini, marilah kita lihat berkas bernama pepatah.txt yang isinya sebagai berikut :

```
$ cat pepatah.txt
sedikit demi sedikit
lama-lama menjadi bukit

$_
```

Informasi lengkap tentang atribut berkas tersebut dapat dilihat melalui perintah :

```
$ ls –l pepatah.txt
-rw-r--r-- 1 titin  ftikusm 45 Mar  6 14:36 pepatah.txt
```

```
$_
```

Tampak bahwa jumlah *link* dari pepatah.txt berupa 1. Jumlah *link* akan segera berubah setelah perintah berikut diberikan :

$ ln pepatah.txt peribahasa.txt

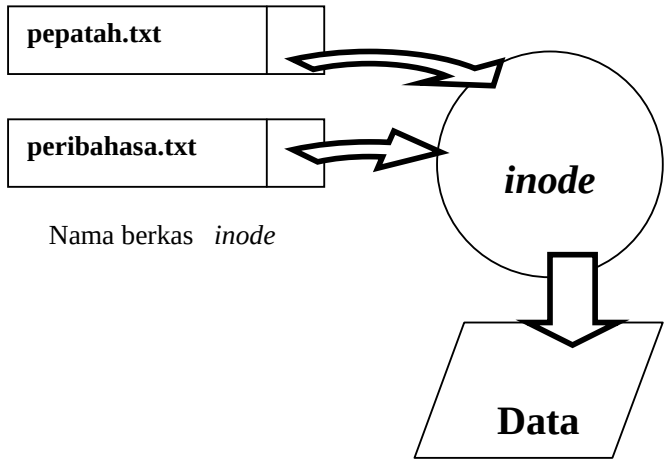
```
-rw-r--r--    2 titin    ftikusm    45 Mar  6 14:36 pepatah.txt
-rw-r--r--    2 titin    ftikusm    45 Mar  6 14:36 peribahasa.txt
$_
```

Tampak, kini jumlah *link* dari pepatah.txt menjadi 2, yang menyatakan bahwa ada satu berkas lagi yang merupakan *link* dari pepatah.txt yaitu berkas peribahasa.txt.

Dengan adanya perintah **ln** seperti itu, kedua berkas tersebut akan menunjuk ke *inode* yang sama. Hal itu dapat dilihat melalui perintah **ls** dengan pilihan beripa -i.

```
$ ls  -i   pepatah.txt    peribahasa.txt
61 pepatah.txt
61 peribahasa.txt
$_
```

Tampak, nomor *inode* kedua berkas tersebut sama.



Gambar 7.12 Dua berkas menuju inode yang sama

Jika salah satu dari berkas pada contoh di atas dirubah, yang lain juga berubah, sebab fisik dari kedua berkas tersebut hanya ada satu.

Catatan: Pada perintah ep, fisik berkas asli dan berkas hasil salinan berbeda, tetapi isinya sama. Perubahan pada salah satu berkas tidak mempengaruhi yang lainnya.

Sekalipun *inode* dari pepatah.txt dan peribahasa.txt sama, jika salah satu berkas tersebut dihapus tidaklah menghapus berkas yang lain. Yang hilang hanya nama berkas yang dihapus. Selain itu, jumlah *link* dari berkas lainnya akan berkurang satu. Contoh :

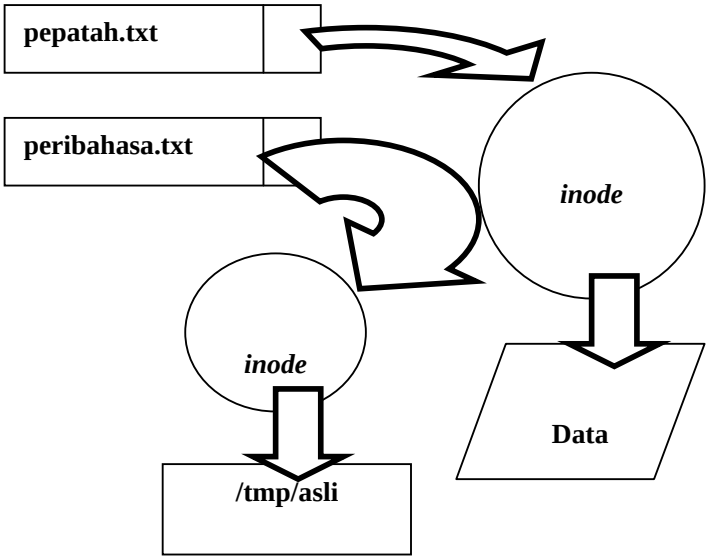
```
$ rm peribahasa.txt
rm: remove ‘peribahasa’? y<ENTER>
$ ls -l pepatah.txt
-rw-r--r--          1 titin          ftikusm          45 Mar  6 14:36 pepatah.txt
$_
```

Kalau jumlah *link* semula dari pepatah.txt bernilai 2, kini bernilai 1.

Jenis *link* yang digambarkan diatas disebut ***hard link***. Jenis *link* ini hanya dapat diterapkan untuk berkas biasa saja dan tidak dapat digunakan untuk direktori. Jenis *link* yang kedua adalah ***soft link*** atau lebih dikenal dengan sebutan ***simbolic link***. Jenis *link* ini terdapat pada sistem yang berkiblat pada BSD, dan dapat digunakan untuk melakukan *link* terhadap berkas maupun direktori. Berbeda dengan *hard link, simbolic link* akan ditampakan pada daftar direktori dengan bentuk :

Nama_alias->nama_berkas_yang_sesungguhnya

Selain itu, *simbolic link* dan berkas asli tidak menunjuk ke *inode* yang sama. Itulah sebabnya jika berkas aslinya dihapus, berkas *link* akan kehilangan data (pada *hard link*, berkas asal dari *link* bisa dihapus tanpa mempengaruhi berkas hasil *link*). *Simbolic link* dapat dipakai untuk memperluas sistem berkas, sebab suatu berkas *link* dapat menunjuk keberkas lain yang letak partisinya berbeda dengan partisi dari berkas *link* itu sendiri (perli diketahui, secara logika suatu disk dapat terbagi atas sejumlah partisi).



Gambar 7.13 Simbolic link (ln -s /tmp/asli).

Catatan: Sebagai perbandingan, pada sistem operasi seperti MS-DOS, sebuah berkas hanya memiliki sebuah nama.

Untuk membuat *simbolic link*, pilihan -s perlu diberikan.

```
ln -s sumber target
```

Pada saat perintah **ls** diberikan, *simbolic link* dinyatakan dalam format : *target ->sumber*

Dalam format panjang (pilihan -l pada ls), tipe berkas dari *simbolic link* berupa huruf l.

Berbeda dengan *hard link*, *simbolic link* bisa digunakan untuk direktori.

Contoh untuk membuat *simbolic link* :

```
$ cat >/tmp/asli
   Berkas ini digunakan untuk keperluan
   tes simbolik link
   <Ctrl-D>
```

```
$_
$ ln -s  /tmp/asli simbol
$ ls -l  /tmp/asli simbol
  lrwxrwxrwx 1 titin     ftikusm 9 Mar 6 14:51 simbol -> /tmp/asli
  -rw-r--r-- 1 titin     ftikusm  55 Mar 6 14:51 /tmp/asli
$_
```

Berbeda dengan *hard link,* pada *simbolic link* jumlah *link* tetap bernilai satu.

7.12. Latihan Soal

1. Sebutkan dan jelaskan alasannya , dibawah ini nama berkas yang salah :
   a. halo.halo.bandung
   b. kantor.cabang
   c. anak pria.1
   d. anak_pria_1
2. Jika didalam direktori kerja terdapat berkas-berkas sebagai berikut :
   .profile
   latihan1
   a.bcdef
   xyz
   x12
   123
   Berkas-berkas manakah yang tidak ditampilkan oleh perintah ls*
3. Perintah yang digunakan untuk menampilkan berkas-berkas dengan nama yang diawali a,b atau c adalah
   a. ls abc*
   b. ls abc?
   c. ls [!abc]*
   d. ls [abc]*
   4. Yang termasuk sebagai berkas tersembunyi adalah
   a. a.abc
   b. aabc
   c. .aabc
   d. a.a.b.c
5. Hasil dari suatu perintah ls –l adalah sebagai berikut :
   Total 6
   -rw-r—r-- 1 titin 0 aug 26 14: 21 blank

-rw-r—r--	1	titin	45	sep	06	06:34	file1.txt
-rw-r—r--	1	titin	0	sep	06	06:35	file2.txt
drw-r—r--	1	titin	2280	aug	26	14: 21	latihan

Jawablah pertanyaan berikut :

a. Pemilik latihan adalah

b. Ukuran dari berkas file2.txt adalah .....byte

c. Jenis berkas dari latihan adalah

d. Jumlah link dari berkas file1.txt adalah

e. Jumlah berkas biasa ada ..... buah

6. Perintah untuk menyalin berkas a dengan hasil berupa b

a. ln a b

b. cpy a b

c. cpy b a

d. cp a b

e. cp b a

7. Perintah ..... dipakai untuk menciptakan berkas kosong

8. Perintah cat tidak dapat dipakai untuk

a. menampilkan isi beberapa berkas

b. menciptakan sebuah berkas

c. menciptakan beberapa berkas

d. menampilkan isi sebuah berkas

9.Perintah .... dipakai untuk menghapus berkas

10.Perintah ..... digunakan untuk mengganti nama berkas

11.Pilihan pada perintah ls yang dipakai untuk menampilkan bilangan inode adalah

a. –a

b. –i

c. –d

d. –l

12.Perintah yang digunakan untuk mencetak isi berkas ke printer adalah

13. Sebutkan sebuah berkas yang cocok dengan ?ay dan m* !

BAB XIII
PEMROGRAMAN SHELL

13.1. Mencegah Shell Menginterpretasikan Karakter

Beberapa karakter Shell mempunyai arti khusus. Contohnya tabel dibawah ini :

Tabel 13.1. Daftar Karakter Shell

Karakter	Keterangan
>	Pengalihan arah keluar
<	Pengalihan arah masuk
>/<	Pengalihan arah keluaran
|	Pipa
*	Nol/karakter apa saja
?	Sebuah karakter apa saja
[ ]	Salah satu karakter ada didalam tanda tsb
[-]	Salah asatu karakter berada dalam jangkauan sebelum dan sesudah tanda -
$	Substitusi dengan nilai variable yang terletak sesudah tanda dolar
&	Mengeksekusi perintah dilatar belakang
;	Pemisah antar perintah
( )	Pengelompokan perintah yang dieksekusi di subshell
{ }	Pengelompokan perintah yang akan dieksekusi pada shell sekarang
\	Karakter sesudah ini akan diperlakukan sebagai karakter biasa
‘ ‘	Petik tunggal untuk mematikan karakter bermakna khusus (Kecuali ! pada C shell)
“ “	Petik ganda untuk mematikan semua karakter bermakna khusus (kecuali $,’

	dan \ serta ! pd C shell)
Ctrl+D	Tanda akhir berkas
Karakter	Keterangan
~	Shell akan mensubstitusi dengan home direktori jika berdiri sebagai kata sendiri atau diikuti dengan ! (Khusus C shell dan korn shell)
||	Digunakan sebagai pemisah antar perintah yang ada dibelakang tanda ini, berdasarkan kondisi pada perintah yang ada didepan tanda ini.
&&	Diletakkan sebagai pemisah antar perintah untuk mengeksekusi perintah yang ada dibelakang tanda ini, berdasarkan kondisi pada perintah yang ada didepan tanda ini.
$Variable	Shell akan mensubstitusi dengan nilai dari variable

Untuk mencegah shell menginterpretasikan karakter seperti itu perlu ditulis tanda kutip (quote). Tanda kutip ini berupa :

- Backslash (\ )
- Petik tunggal (‘)
- Petik ganda (“)

13.2. BACKSLASH

Karakter ini dapat digunakan untuk mematikan metakarakter.

Contoh :

$ echo 3 \> 2

3>2

$__

Pada contoh ini, \> berarti karakter > (bukan karakter >).

Perintah :

echo 3>2

Mempunyai makna yang lain dengan perintah :

echo 3 \> 2

Sebab, > merupakan simbol untuk pengalihan arah keluaran (output redirection).

13.3. PETIK GANDA

Dengan karakter ganda, karakter khusus akan menjadi biasa. Karakter khusus yang tidak dapat dijadikan karakter biasa. Contoh :

- $ (dolar)
- \ (bacslash)
- ‘ (backquote)

Karakter – karakter tersebut harus diawali dengan backslash (\) agar berlaku sebagai karakter biasa. Contoh :

echo “$HOME”

Akan menampilkan isi variable HOME, dan bukan menampilkan tulisan

$HOME

$ echo “SHOME’

/home /titin

$__

Agar $HOME tidak diartikan sebagai “isi dari variable HOME”,tanda $ perlu diawali dengan backslash. Contoh :

$ echo “\SHOME”

$HOME

$__

Untuk menghasilkan tulisan :

“selamat pagi”

Pada layar, dengan awalan dan akhiran tanda petik ganda,diperlukan tulisan sebagai brkt :

$ echo “\”selamat pagi\” “

“selamat pagi”

$__

13.4. PETIK TUNGGAL

Karakter ini lebih ampuh. Mempunyai arti kusus, jika didalam tanda petik ganda akan diperlakukan seperti biasa kalau terletak didalam tanda petik tunggal.

Contoh :

$ echo ‘$HOME’

$HOME

$__

Tampak dengan tanda petik tunggal, karakter $ akan diperlakukan sebagai biasa. Tidak demikian halnya jika diletakkan dalam tanda petik ganda.

13.5. Jika Tanda Petik Tunggal dan Petik Ganda Belum Lengkap

Pada Bourne shell, Bourne Again Shell, maupun Korn Shell, jika tombol <ENTER> ditekan sementara suatu tanda petik ganda belum ditutup, shell akan menunggu tanda petik penutup pada baris berikutnya.

Contoh :

$ echo ‘Hai<ENTER>

>_

Pada contoh diatas tanda petik tunggal penutup belum diberikan. Maka shell menunggu tanda petik itu diberikan. Dengan menampilkan simbol prompt shell berupa tanda >. Dengan memberikan tanda petik dan menekan <ENTER>, seperti berikut :

>’<Enter>

Perintah echo tsb akan segera dieksekusi.

Jika kasus serupa terjadi pada C shell akan menampilkan pesan kesalahan :

Unmatchead ‘.

Atau :

Unmatchead ‘.

% echo ‘Hai<enter>

Unmatchead ‘.

%_

13.6. Menulis Beberapa Perintah Dalam Satu Baris

Shell memperkenalkan tanda perintah atau lebih ditulis dalam sebuah baris. Pada keadaan ini syarat yang diperlukan hanyalah memisahkan dengan tanda (;). Adapun sebelum dan sesudah boleh ada spasi. Contoh :

pwd<enter>

ls<enter>

Kedua baris dapat ditulis dalam sebuah baris :

pwd;ls<enter>

13.7. Tanda backslash diakhir baris

Jika shell menemukan tanda backslash (\) diakhir baris, shell akan menganggap garis berikut adalah kelanjutan dari baris tsb. Contoh :

*<enter>

Dengan kata lain

ls \<enter>

*<enter>

Akan diperlakukan sebagai :

ls *<enter>

13.8. Substitusi perintah

Karakter backquote (‘) berguna untuk melakukan substitusi perintah, sehingga hasilnya dapat diletakkan pada perintah yang lain.

‘Perintah’

Contoh :

$ echo ”Direktori Kerja Sekarang:’pwd’ “

Direktori Kerja Sekarang : /usr /titin

$_

Contoh lain :

$ echo ”jumlah yang login:’who | we –l’ orang”

$_

Pada contoh diatas, subtitusi perintah dilakukan antar string “jumlah yang login: “ dan “ orang “. Perlu diketahui, we –l digunakan untuk menghitung jumlah baris dari keluaran perintah who. Hasil ini dapat ditaruh dalam variable.

13.9. Pengelompokkan perintah

Simbol () papat digunakan untuk mengkombinasikan standart output atau standart error dari sejumlah perintah menjadi satu keluaran.

Contoh :

$(ls –l /bin; who) | we –l

\pada contoh ini we –l akan menghitung baris keluaran dari is dan who. Dengan kata lain, simbol () perintah akan menjadikan perintah Is –l /bin dan who sebagai sebuah kelompok perintah.

Perbedaan yang jelas akan terlihat pada contoh dibawah ini :

$ ls –l /bin; who | we –l

Pada contoh ini yang dihitung hanya keluaran dari who.

Perintah-Perintah yang terletak didalam tanda () sebenarnya dieksekusi oleh subshell(bukan oleh shell utama). Keistimewaan yang diberikan oleh subshell terdapat perintah untuk mengubahdirektori kerja, pengubahan tersebut tidak mempengaruhi direktori kerja shell utama, sekembalinya dari subshell, direktori kerja semula tidak berubah.

Contoh :

```
$ cd
$ pwd
  /home /titin
$ (cd/bin;pwd)
  /bin
$pwd
  /home /titin
$_
```

Perintah cd digunakan untuk mengaktifkan home directory sebagai directory kerja. Perintah pwd yang pertama menyatakan bahwa direktori kerja berupa:

/home /titin

Kemudian:

(cd /bin;pwd)

Menyebabkan /bin diset sebagai direktori kerja. Hal ini ditunjukan oleh informasi yang diberikan oleh perintah pwd :
/bin

Sekembalinya dari subshell, perintah pwd digunakan untuk memeriksa kembali direktori kerja yang sekarang. Terlihat direktori kerja bukanlah /bin melainkan tetap berupa /home /titin. Ini membuktikan bahwa pengubahan direktori kerja pada subshell tidak mempengaruhi direktori kerja shell utama.
Khusus pada bourne shell dan korn shell terdapat bentuk pengelompokkan perintah brp:
{ list ;}
Dalam hal ini list adalah sederetan perintah atau lebih ditulis dalam satu baris, antar perintah perlu dipisahkan dengan titik koma (;).

Beberapa aturan perintah :

- Antara { dan perintah pertama harus terdapat minimal sebuah spasi
- Perintah terakhir sebagai tanda } (yang berada satu baris dengan }) harus diakhiri dengan tanda titik koma (;).
- Antara tanda titik koma dan } tidak harus terdapat spasi.
- Jika didepan tanda { terdapat perintah, perintah harus diakhiri dengan tanda titik koma (;).

Bentuk { list; } mempunyai perbedaan terhadap bentuk (list). Pada bentuk { list; }, semua perintah dikerjakan oleh shell yang sama dan bukan oleh subshell.
Contoh :

```
$ cd
$ pwd
/home /titin
$ {cd/bin;pwd;}
/bin
$pwd
/bin
$pwd
$_
```

13.10. Pemakaian && dan || diantara dua perintah

Dua buah perintah dalam satu baris dapat dipisahkan oleh simbol && atau ||.

Kedua simbol tersebut dipakai untuk mengatur eksekusi dari perintah yang terletak sesudah simbol – simbol tersebut berdasarkan kondisi pada perintah yang terletak dikanan simbol tsb.

A. Simbol &&

Bentuk pemakaiannya :

Perintah_l && perintah_2

Dalam hal ini perintah_2 akan dieksekusi kalau peintah_1 melaksanakan tugasnya.

CATATAN

Keberhasilan tugas yang di emban oleh suatu perintah biasanya akan dinyatakan dengan nilai keluar (exit code) bernilai 0. Nilai keluar ini dapat diperiksa melalui perintah :

echo $? (pada bourne shell, bourne again shell, dan Korn shell)

echo $status (pada Cshell)

Contoh ;

$ ls

$ echo $?

0 ← Nilai keluar = 0, berarti tak ada sesuatu kesalahan

$_

$ls alakadabra

$ echo $?

1 ← Nilai keluar = 1, karena berkas *alakadabra* tidak ada

$_

Contoh Penggunaan && :

$ ls f_dan && echo berkas tersebut ada

  f_dan: No such berkas or directory

$ touch f_dan

$ Is f_dan && echo berkas tersebut ada
f_dan
berkas tersebut ada
$_

Pada contoh pemakaian && yang pertama, perintah echo tidak dieksekusi karena perintah Is menghasilkan kesalahan (yaitu karena berkas *f_dan* tidak ada). Pada contoh pemakaian && yang kedua, berhubung berkas *f_dan* ada , maka perintah echo ikut dieksekusi.

Simbol ||

Bentuk pemakaianya :

Perintah_1|| *perintah_2*

Dalam hal ini, *perintah_2* akan dieksekusi kalau *perintah_1* TIDAK berhasil melaksanakan tugasnya (ada sesuatu kesalahan).

Contoh penggunaan ||:

$ ls f_atau || acho berkas tersebut tidak ada
f_atau: No such berkas or directory
Berkas tersebut tidak ada
$ touch f_atau
$ ls f_ atau || echo berkas tersebut tidak ada
f_atau
$_

Pada contoh di atas, peritah echo justru dieksekusi kalau perintah pertama (Is) menghasilkan kesalahan.

Penggabungan && dan ||

Simbol && dan || dapat dikombinasikan, seperti ditunjukan oleh contoh berikut:

$ ls f_coba && echo berkas ada || echo berkas tak ada
F_coba : No such berkas or directory
Berkas tak ada
$ touch f_coba
$ ls f_coba && echo berkas ada | echo berkas tidak ada
f_coba

berkas ada
$_

Pada contoh di atas :
ls f_coba && echo berkas ada || echo berkas tidak ada

Perintah :
echo berkas ada
echo berkas ada
yang dieksekusi

Pengeksekusian Perintah di Latar Belakang

LINUX merupakan sistem operasi yang memungkinkan setiap pemakai sistem dapat menjalankan beberapa perintah yang dieksekusi secara bersamaan. Dalam hal ini, hanya sebuah proses (program yang sedang dieksekusi) yang berjalan di latar depan untuk setiap terminal dan lainya berjalan dilator belakang.

Suatu ciri proses yang berjalan dilatar belakang adalah begitu perintah diberikan, *shell* akan menampilkan *prompt* kembali sehingga memungkinkan pemakai untuk memberikan perintah yang lain, sementara perintah yang dieksekusi dilatar belakang tetap berjalan. Proses yang dieksekusi dilatar belakang biasanya dilakukan pada proses yang memakan waktu lama dan tidak ada interaksi dari *keyboard*. Adapun hasil pemrosesan diletakkan ke berkas. Misalnya, proses untuk memperoleh daftar berkas dari keseluruhan sistem. Jika proses seperti ini dilakukan dilatar depan, Anda tidak akan dapat melakukan apa-apa hingga proses tersebut berakhir.

Agar suatu proses dijalankan di latarbelakang, tambahkan tanda & diakhir perintah. Contoh:

```
ls   -IR/>ls.tmp 2>&1 &
```

Pada contoh ini:

```
ls  -IR/> ls.tmp 2>&1
```

Akan dieksekusi dilatar belakang.

Perlu diketahui, setelah perintah dengan eksekusi dilatarbelakang diberikan, shell akan melaporkan informasi semacam berikut:

```
[1] 293
```

Mematikan Proses dilatar Belakang

Jika dikehendaki untuk mematikan proses yang berjalan dilatarbelakang anda bisa menggunakan perintah kill. Bentuk pemakaian perintah ini :

kill [-sinyal] nomor_proses…

Kode sinyal yang dipakai untuk mematikan proses adalah 9 dan 15. Adapun nomor proses adalah kode identitas dari prosses yang hendak dimatikan. Nomor proses dari suatu perintah yang dieksekusi di latarbelakang ditampilkan pada saat perintah tersebut diberikan. Misalnya angka 293 :

Kill -15 293

Sebab 15 adalah default dari sinyal.

Namun, adakala dengan perintah seperti data, proses tidak dapat dimatikan (terjadi pada proses yang kebal terhadap sinyal nomor 15). Kalau terjadi hal seperti ini, berikan perintah :

kill -9 239

Yang dengan pasti akan mematikan proses.

Melihat Proses

Proses-proses yang sedang berjalan dan merupakan milik anda dapat dilihat dengan menggunakan perintah ps. Contoh:

```
$ ps
PID TTY    TIME CMD
1441 ttyl  00:00:00 bash
1632 ttyl  00:00:00 ps
$_
```

Arti informasi diatas :

- PID :menyatakan nomor proses
- TTY :menyatakan kode terminal tempat proses dibentuk
- TIME :total waktu untuk proses
- COMMAND nama perintah dari proses

Dengan kata lain, proses pada terminal tempat ps diberikan hanya ada dua buah. Bash nama shell yang sedang digunakan (Bourne Again shell), ps adalah proses yang dibentuk adanya pemberian perintah ps.

Perintah tersebut juga akan melaporkan proses di latarbelakang, kalau ada.

Sebagai contoh :

```
$ sleep 50&
[1] 319
$ps
  PID TTY  TIME CMD
1441      ttyl     00:00:00 bash
1633      ttyl     00:00:00 sleep
1635      ttyl     00:00:00 ps
$
```

Perlu diketahui dengan menambah pilihan –ef pada ps, semua proses dalam sistem akan diperlihatkan

Lain-lain

Masih banyak kemampuan shell yang tidak dibahas pada buku ini, terutama yang berguna untuk modus pemrograman. Misalnya kemampuan dalam melakukan subtitusi parameter yang sangat bervariasi yang terdapat pada korn shell.

Ringkasan

- Karaker-karakter yang mempunyai arti khusus bagi shell seperti ? dan * akan diperlakukan sebagai karakter biasa kalau ditulis dalam tanda kutip(quote). Tanda kutip yang biasa dipakai.

-backslash(\)

-Petik ganda (“)

-Petik tunggal (‘)

- Dua buah perntah atau lebih bias ditulis dalam sebuah baris, asalkan antar perintah dipisahkan oleh tanda titik koma (;)

- Untuk menuliskan perintah yang sangat panjang, pemakai dapat menuliskannya lebih dari satu baris. Syaratnya, perintah yang belum berakhir perlu ditulis dengan diakhiri oleh tanda backslash (\).
- Ada dua cara untuk mengelompokkan perintah :
    1. Ditulis di dalam tanda (perintah_perintah)
    2. Ditulis dengan format {perintah_perntah}
- Simbol && dan || juga dapat dipakai untuk memisahkan dua pertanyaan, dengan format :

1. pertanyaan_1 && pertanyaan_2
2. pertanyaan_1 || pertanyaan_2

pada format pertama, pertanyaan_2 dieksekusi hanya kalau pertanyaan_1 tidak mengalami sesuatu kesalahan.

Pada format kedua, pertanyaan_2 hanya dieksekusi kalau pertanyaan_1

- Suatu perintah akan dijalankan dilatarbelakang kalau perintah tersebut ditulis dengan diakhiri tanda &
- Suatu perintah yang berjalan di latar belakang dapat dimatikan dengan memberikan perintah :
-

Kll -15 nomor_proses

- Perintah ps bermanfaat untuk mengetahui proses-proses yang sedang berjalan.

13.11. Latihan Soal

1. Perintah untuk menghasilkan tulisan
   3>5 Pada layar adalah
2. Suatu perintah akan dijalan dilatar belakang kalau diakhiri dengan simbol
3. Perintah untuk menampilkan tulisan
   $HOME
   Pada layar :
   a. echo $HOME
   b. echo “$HOME”
   c. echo /$HOME
   d. echo ‘$HOME’

4. Dua buah perintah yang berada di dalam sebuah baris perlu ditulis dengan simbol selain
    a. ;
    b. ||
    c. &&
    d. ,
5. Utilitas yang digunakan untuk melihat proses-proses yang sedang berlangsung yaitu
6. Perintah yang akan menyebabkan hasil dari date dan pwd dimasukkan ke berkas bernama hasil.tmp
    a. date; pwd > hasil.pwd
    b. (date; pwd) > hasil.pwd
    c. (date|| pwd) > hasil.pwd
    d. date|| pwd > hasil.pwd
7. hasil dari perintah : echo “/”HAI, CING” adalah :
    a. “HAI,CING”
    b. “HAI,CING
    c. \”HAI,CING
    d. \HAI,CING