Manajemen File pada Linux

Disusun Untuk Memenuhi Tugas Akhir
Mata Kuliah Sistem Operasi

Adi Asriadi
Haris Bhakti Prasetyo
M Rasyid Sidiq
Shodiq Fathoni

PROGRAM STUDI MATEMATIKA
JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI JAKARTA
JUNI 2006

Daftar Isi

Daftar Gambar

Kata Pengantar

Puji Syukur dan beberapa untaian kata indah untuk mengungkapkan rasa syukur kehadirat Allah SWT. Segala hidayah, nikmat dan karunianya, sehingga kami dapat menyelesaikan tugas pada mata kuliah sistem operasi ini dengan lancar yang telah menjadi pengalaman berharga untuk saya untuk menyambut hari yang lebih baik. Dan tak lupa Sholawat dan Salam selalu dicurahkan dan dilantunkan kepada suri tauladan Nabi Muhammad SAW yang telah mengajarkan kita untuk menjadi manusia yang berakhlak mulia.

Penulisan makalah ditulis atas dasar sebagai prasyarat mata kuliah sistem operasi yang berbobot 3 SKS namun untuk mengenal secara jauh ruang lingkup tentang ilmu komputer yang tentunya hubungannya dengan kinerja manajemen file sistem operasi di linux.

Makalah ini disusun tidak lepas dari banyak semua pihak yang telah membantu selama pelaksanaan PKL ini. Dalam kesempatan ini Kami akan mengucapkan terima kasih yang sedalam-dalamnya kepada pihak yang saya hormati :

- Orang Tua kami, yang telah memberikan bantuan semangat dan motivasi materiil, moril , spiritual dan dukungannya.
- Bapak Med Irzal S.Kom, selaku dosen mata kuliah sistem operasi yang telah memberikan ilmu yang berguna untuk kami aplikasikan
- Teman-teman kami seperjuangan khususnya team work kelompok kami yang telah meluangkan waktunya mengerjakan tugas ini.

Adapun untuk penyusunan makalah tentang manajemen file/berkas di linux dikerjakan di rumah salah satu kelompok kami, haris. Dan terselesaikan makalah ini tidak lepas dari kerjasama tim kami. Dibawah ini adalah nama-nama yang mempunyai andil besar tersusunnya makalah ini :

- Adi Asriadi : Sebagai narasumber untuk informasi sistem file linux, mencari bahan/referensi di internet, mengatur tampilan LaTeX 2_ε(Karena makalah ini ditulis pada aplikasi LaTeX 2_ε).
- Haris Bhakti Prasetyo : Studi literatur perpustakaan di Fasilkom UI sebagai pendukung dari referensi, membuat kerangka dari makalah yang

akan dibahas, konsep bab Pendahuluan, juga sebagai sekretaris pengetikan isi makalah ini.

- M Rasyid Sidiq : Mencari ide tentang konsep Mounting pada bagian sistem berkas di Linux.
- Shodiq Fathoni : Mencari ide tentang konsep Filesystem Hierarchy Standard.

Dan Kami sadari dalam isi makalah ini masih jauh dari kesempurnaan oleh karena itu kritik dan saran sangat saya harapkan untuk membangun motivasi Kami untuk perbaikan karya ini dan berkarya lebih baik masa mendatang. Kami harapkan semoga Makalah ini sangat bermanfaat sebagai menambah pengetahuan pembaca tentang dunia IPTEK yang makin berkembang di masa depan.

Bab 1

Pendahuluan

1.1 Latar Belakang

Semua aplikasi komputer butuh menyimpan dan mengambil informasi. Ketika sebuah proses sedang berjalan, proses tersebut menyimpan sejumlah informasi yang terbatas, dibatasi oleh ukuran alamat virtual. Untuk beberapa aplikasi, ukuran ini cukup, namun untuk lainnya terlalu kecil.

Masalah berikutnya adalah apabila proses tersebut berhenti maka informasinya hilang. Padahal ada beberapa informasi yang penting dan harus bertahan beberapa waktu bahkan selamanya.

File atau berkas adalah sebuah unit tempat menyimpan informasi. File ini dapat diakses lebih dari satu proses, dapat dibaca, dan bahkan menulis yang baru. Informasi yang disimpan dalam file harus persisten, dalam artian tidak hilang sewaktu proses berhenti. file-file ini diatur oleh sistem operasi, bagaimana strukturnya, namanya, aksesnya, penggunaanya, perlindungannya, dan implementasinya. Bagian dari sistem operasi yang mengatur masalah-masalah ini disebut sistem file.

Mempelajari sistem operasi di linux tidaklah sulit karena pada dasarnya jika penggunanya dapat mengetahui prosedur pada sistem UNIX, pemakai linux dapat mempelajarinya dengan mudah. pada manajemen file di linux yang perlu ditekankan kita dapat mengetahui direktori link, atributnya seperti apa, bagaimana mengoperasikan filenya, bagaimana struktur direktorinya (hierarchy), bagaimanakah konsep mounting, dan bagaimana display/tampilan dari linux/window manager.

1.2 Mengapa kita harus mengetahui manajemen file di Linux

Makin banyaknya aplikasi-aplikasi software pada akhir-akhir ini, sistem operasi dituntut berkembang juga karena sistem operasi sebagai sarana pen-

dukung untuk menjalankan aplikasi tersebut. Masalah di dunia komputer pada dekade ini adalah masalah makin berkembangnya produk aplikasi software bajakan. Harga antara produk bajakan dengan yang original relatif jauh sekali. Parahnya lagi banyak kalangan pengguna komputer menggunakan produk bajakan, alasannya kualitasnya tidak jauh berbeda dengan yang original.

Linux diciptakan untuk mengatasi masalah diatas karena linux adalah sistem operasi yang bersifat open source (disebarkan secara luas) dibawah lisensi GNU *General Public Licence* (GPL). Sebelum mempelajari linux, kita harus mengetahui terlebih dahulu konsep dari sistem filenya. Secara garis besar sistem file yang ada di linux seperti :

1. Root yang lambangnya "/"
2. terminal (seperti *command prompt* di windows)
3. mounting (untuk memanggil drive dari harddisk, floopy, maupun usb)
4. *File Sistem* EXT2, EXT3, dan Reiser (seperti NTSF dan FAT pada di Windows)
5. Struktur direktori Linux
6. dll

Hal yang perlu dicermati mengenai file adalah apakah file itu terjangkit oleh virus atau tidak. Pada di Windows sangatlah sensitif terhadap virus dan ini berbeda pada sistem operasi di linux karena sampai saat ini penulis tidak menemukan virus di linux yang tidak mengenal terhadap virus. Akan tetapi pada saat ini banyak software antivirus yang berjalan diatas platform Linux, ini dibuktikan dengan adanya AVG Free for Linux, Antivir dll.

Keselamatan dan keamanan terhadap file dari virus haruslah diperhatikan karena berbagai informasi yang telah tersimpan di file sangatlah penting. Akhirnya pengguna komputer sebaiknya mengetahui manajemen file yang mengatur berbagai aktivitas file di linux sebagai alternatif untuk mengatasi masalah sistem operasi.

Bab 2

Konsep Berkas Linux

2.1 Pengertian Dasar

File system atau manajemen file adalah metode dan struktur data yang digunakan sistem operasi untuk mengatur dan mengorganisir file pada disk atau partisi. File system juga dapat diartikan sebagai partisi atau disk yang digunakan untuk menyimpan file-file dalam cara tertentu. Cara memberi suatu file system ke dalam disk atau partisi dengan cara melakukan Format.

File system Linux kebanyakan menggunakan ext2 (baca: second extended) atau ext3, file system yang tidak mengalami fragmentasi seperti halnya file system windows (FAT/FAT32). Ext2 juga memiliki system security yang baik dengan menerapkan access permission untuk owner, group owner, dan other.

2.2 Atribut File

Setiap sistem dalam manajemen file mempunyai sistem atribusi yang berbeda-beda, namun pada dasarnya di linux mempunyai atribut seperti berikut ini:

- Nama: Nama berkas simbolik ini adalah informasi satu-satunya yang disimpan dalam format yang dapat dibaca oleh pengguna.
- Indentifiers: Tanda unik ini yang biasanya merupakan sebuah angka, mengenali berkas didalam sebuah berkas; tidak dapat dibaca oleh pengguna.
- Tipe: Informasi ini diperlukan untuk sistem-sistem yang mendukung tipe berbeda (misal: .tar.gz pada kompresi, .tex pada dokumen latex).
- Lokasi: Informasi ini adalah sebuah penunjuk pada sebuah device tersebut(misal: harddisk, UFD(usb flashdisk), floppy, DVD rom dll).
- Ukuran: Ukuran dari sebuah berkas (dalam bytes, words, atau, blocks) dan mungkin ukuran maksimum dalam atribut juga.

- Permission : Informasi yang menentukan siapa yang dapat melakukan read, write, execute, dan lainnya.
- Waktu dan identifikasi pengguna : informasi ini dapat disimpan untuk pembuatan berkas, modifikasi terakhir, dan penggunaan terakhir. Data-data ini dapat berguna untuk proteksi, keamanan, dan monitoring pengggunaan.

Inode adalah Informasi yang mengidentifikasikan suatu file secara unik. Inode mengidentifikasikan lokasi tempat file disimpan, dan karakteristik dari file tersebut. (owner, date, dsb); tetapi nama file tidak disimpan sebagai bagian dari inode. Informasi yang disimpan dalam inode antara lain :

Izin Akses File (File Permission)

Tidak seperti halnya sistem operasi DOS, setiap file Linux memiliki status izin akses (file permission). Maksudnya setiap file memiliki informasi untuk mengatur siapa yang berhak untuk membaca, menjalankan atau mengubah file tersebut.

Linux merupakan sistem operasi multiuser dan umumnya digunakan sebagai sistem operasi untuk jaringan. Oleh karena itu untuk menjaga privasi file, keamanan serta integritas sistem agar tidak terganggu, izin akses file digunakan untuk melindungi file/sistem dari orang lain yang tidak mempunyai hak.

Bayangkan tanpa adanya fasilitas ini maka mail anda akan dapat di baca oleh seluruh orang yang terhubung dalam jaringan yang sama. File-file anda tidak akan dapat dijamin keamanannya dari penghapusan dan pencurian oleh orang lain. Oleh karena itu penting bagi anda untuk memahami izin akses suatu file.

Perhatikan file di bawah ini :

```
darkstar:~$ ls -l filetes
-rw-r- -r- - l juli users 121 Dec 17 12:12 filetes
```

Notasi yang dicetak tebal-miring itulah yang menyatakan izin akses file.

```
- rw- r-- r-- 1 juli users
1   2   3   4
```

Gambar 2.1: Izin akses file

Perhatikan bahwa notasi di atas terdiri atas sepuluh digit yang dapat dikelompokkan sebagai berikut:

Notasi pertama menyatakan tipe dari file tersebut. Tanda dash (-) menyatakan bahwa file tersebut adalah file biasa. Untuk direktori maka lokasi tersebut akan berisi karakter d, karakter l untuk link file, dan beberapa tipe lain yang tidak dibahas pada bab ini.

Notasi ke dua yang terdiri dari tiga karakter menunjukkan status file untuk pemilik (owner) dalam hal ini adalah juli. Adapun tiga karakter tersebut mengikuti aturan berikut:

- Karakter pertama untuk menentukan hak baca dari pemilik, karakter r berarti pemilik memiliki hak baca terhadap file tersebut sedangkan bila berisi karakter dash (-) berarti pemilik file tidak memiliki hak baca terhadap file tersebut.

- Karakter kedua untuk menunjukkan hak tulis terhadap file tersebut, dalam hal ini user tersebut memiliki hak untuk menulis atau mengubah file tersebut.

- Karakter ketiga untuk menentukan apakah file tersebut dapat dieksekusi oleh pemilik. Bila file tersebut dapat dieksekusi maka digit tersebut akan berisi karakter x.

Notasi ketiga juga terdiri atas tiga karakter, dimana tiga karakter ini menentukan izin akses file untuk seluruh user yang memiliki grup yang sama dengan user tersebut. Dalam hal ini grup dari pemilik file tersebut adalah users. Cara memahami tiga karakter izin grup ini pun sama dengan izin untuk pemilik file yang sudah dijelaskan sebelumnya. Dalam hal ini seluruh user yang termasuk dalam grup user hanya memiliki hak baca terhadap file tersebut.

Tiga karakter terakhir (notasi ketiga) digunakan untuk menentukan izin file untuk user lain yang tidak termasuk dalam grup tersebut (diistilahkan sebagai others). Dalam hal ini others hanya memiliki hak baca dan tidak hak tulis ataupun hak menjalankannya.

Secara umum file tersebut hanya dapat dibaca dan ditulis oleh pemilik yaitu juli, dan user lain yang berada satu grup (users) serta orang lain hanya dapat membaca file tersebut.

2.3 Operasi File

Fungsi dari berkas adalah untuk menyimpan data dan mengizinkan kita membacanya. Dalam proses ini ada beberapa operasi yang dapat dilakukan berkas. Ada pun operasi-operasi dasar yang dilakukan berkas, yaitu:

- Membuat Berkas (Create):
  Kita perlu dua langkah untuk membuat suatu berkas. Pertama, kita harus temukan tempat didalam sistem berkas. Kedua, sebuah entri untuk berkas yang baru harus dibuat dalam direktori. Entri dalam direktori tersebut merekam nama dari berkas dan lokasinya dalam sistem berkas.

- Menulis sebuah berkas (Write) :
  Untuk menulis sebuah berkas, kita membuat sebuah system call yang menyebutkan nama berkas dan informasi yang akan ditulis kedalam berkas.

- Membaca Sebuah berkas (Read):
  Untuk membaca sebuah berkas menggunakan sebuah system call yang menyebut nama berkas yang dimana dalam blok memori berikutnya dari sebuah berkas harus diposisikan.

- Memposisikan Sebuah Berkas (Reposition):
  Direktori dicari untuk entri yang sesuai dan current-file-position diberi sebuah nilai. Operasi ini di dalam berkas tidak perlu melibatkan M/K, selain itu juga diketahui sebagai file seek.

- Menghapus Berkas (Delete):
  Untuk menghapus sebuah berkas kita mencari dalam direktori untuk nama berkas tersebut. Setelah ditemukan, kita melepaskan semua spasi berkas sehingga dapat digunakan kembali oleh berkas-berkas lainnya dan menghapus entry direktori.

- Menghapus Sebagian Isi Berkas (Truncate):
  User mungkin mau menghapus isi dari sebuah berkas, namun menyimpan atributnya. Daripada memaksa pengguna untuk menghapus berkas tersebut dan membuatnya kembali, fungsi ini tidak akan mengganti atribut, kecuali panjang berkas dan mendefinisikan ulang panjang berkas tersebut menjadi nol.

Keenam operasi diatas merupakan operasi-operasi dasar dari sebuah berkas yang nantinya dapat dikombinasikan untuk membentuk operasi- operasi baru

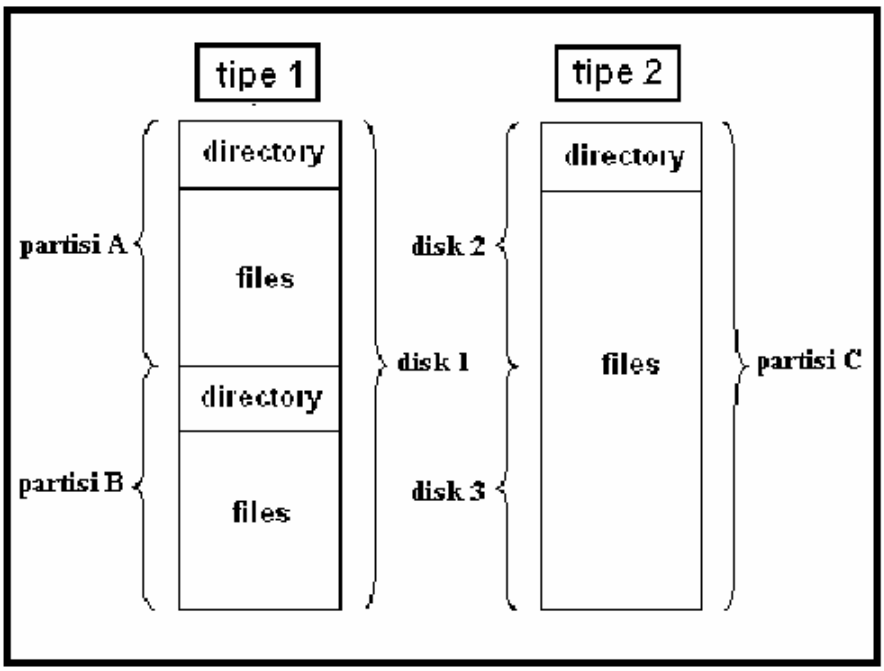
lainnya. Contohnya, apabila kita ingin menyalin sebuah berkas, maka kita memakai operasi create untuk membuat berkas baru, read untuk membaca berkas yang lama, dan write untuk menulisnya pada berkas yang baru.

2.4 Struktur Direktori

Beberapa sistem komputer menyimpan banyak sekali berkas-berkas dalam disk, sehingga diperlukan suatu struktur pengorganisasian data agar lebih mudah diatur.

Direktori atau folder merupakan suatu entitas dalam sebuah berkas sistem yang mengandung berkas atau mengandung direktori lain. Sebenarnya, pada hakikatnya berkas atau berkas terdapat dalam disk, direktori hanya menyediakan link atau menunjuk pada berkas yang ada.

Dengan demikian, dapat disimpulkan bahwa direktori digunakan sebagai sarana untuk pengorganisasian berkas pada suatu sistem komputer. Dengan direktori, berkas-berkas dapat dikelompokkan. Berkas tersebut dapat berisi berkas ataupun direktori lain, sehingga direktori dapat juga disebut sebagai berkas istimewa.

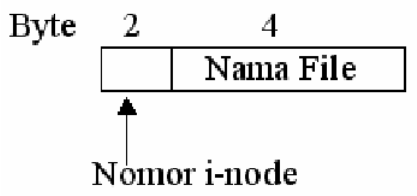


Gambar 2.2: File Organization

Pada atribut, perbedaan pada direktori dan atribut pada berkas yaitu direktori tidak mempunyai tipe, sedangkan berkas mempunyai banyak tipe.

Direktori pada Unix

Struktur direktori yang digunakan dalam Unix adalah struktur direktori tradisional. Seperti yang terdapat dalam gambar direktori entri dalam Unix, setiap entri berisi nama berkas dan nomor inode yang bersangkutan. Semua informasi dari jenis, kapasitas, waktu dan kepemilikan, serta block disk yang berisi inode. Sistem Unix terkadang mempunyai penampakan yang berbeda,tetapi pada beberapa kasus, direktori entri biasanya hanya string ASCII dan nomor inode.



Gambar 2.3: Direktori Unix

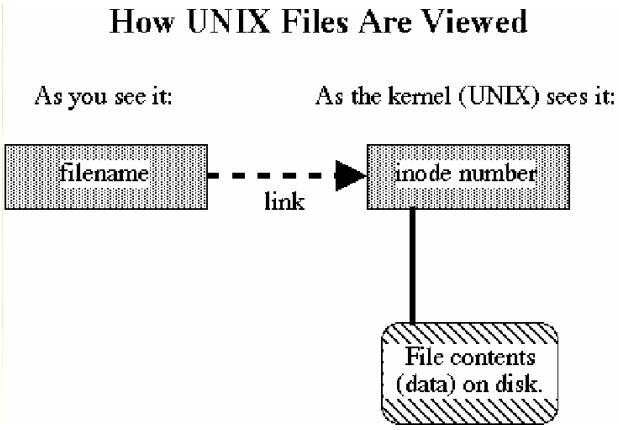
Saat berkas dibuka, sistem berkas harus mengambil nama berkas dan mengalokasikan *block disk* yang bersangkutan, sebagai contoh, nama *path* /usr/ast/ mbox dicari, dan kita menggunakan Unix sebagai contoh, tetapi algoritma yang digunakan secara dasar sama dengan semua hirarki sistem direktori sistem.

Pertama, sistem berkas mengalokasikan direktori *root*. Dalam Unix inode yang bersangkutan ditempatkan dalam tempat yang sudah tertentu dalam disk. Kemudian, Unix melihat komponen pertama dari *path*, usr dalam direktori root menemukan nomor inode dari direktori /usr. Mengalokasikan sebuah nomor inode adalah secara *straight-forward*, sejak setiap inode mempunyai lokasi yang tetap dalam disk. Dari inode ini, sistem mengalokasikan direktori untuk /usr dan melihat komponen berikutnya, dst. Saat dia menemukan entri untuk ast, dia sudah mempunyai inode untuk direktori /ust/ast. Dari inode ini, dia dapat menemukan direktorinya dan melihat mbox. Inode untuk berkas ini kemudian dibaca ke dalam memori dan disimpan disana sampai berkas tersebut ditutup.

Nama *path* dilihat dengan cara yang relatif sama dengan yang absolut.

Dimulai dari direktori yang bekerja sebagai pengganti *root directory*. Setiap direktori mempunyai entri untuk. dan ".." yang dimasukkan ke dalam saat direktori dibuat. Entri "." mempunyai nomor inode yang menunjuk ke direktori di atasnya/orangtua (*parent*), "." kemudian melihat ../dick/prog.c hanya melihat tanda ".." dalam direktori yang bekerja, dengan menemukan nomor inode dalam direktori di atasnya/parent dan mencari direktori disk. Tidak ada mekanisme spesial yang dibutukan untuk mengatasi masalah nama ini. Sejauh masih di dalam sistem direktori, mereka hanya merupakan ASCII string yang biasa. Dari nomor inode ini kita memperoleh inode yang merupakan suatu struktur data yang menyimpan informasi berkas. Penghapusan berkas dilakukan dengan cara melepas inode. Inode ini berisi informasi tentang :

- tipe
- ukuran
- waktu
- owner
- blok-blok disk



Gambar 2.4: Bagaimana file di UNIX ditampilkan

2.5 Konsep Mounting

2.5.1 Mounting

Mounting adalah proses mengkaitkan sebuah sistem berkas yang baru ditemukan pada sebuah piranti ke struktur direktori utama yang sedang dipakai. Piranti-piranti yang akan di-mount dapat berupa cd-rom, disket atau sebuah zip-drive. Tiap-tiap sistem berkas yang akan di-mount akan diberikan sebuah mount point, atau sebuah direktori dalam pohon direktori sistem Anda, yang sedang diakses. Sistem berkas yang dideskripsikan di /etc/fstab(fstab adalah singkatan dari filesystem tables) biasanya akan di-mount saat komputer baru dimulai dinyalakan, tapi dapat juga me-mount sistem berkas tambahan dengan menggunakan perintah:

```
mount[nama_piranti]
```

Atau dapat juga dengan menambahkan secara manual mount point ke berkas /etc/fstab. Daftar sistem berkas yang di-mount dapat dilihat kapan saja dengan menggunakan perintah mount. Karena izinnya hanya diatur read-only di berkas fstab, maka tidak perlu khwatir pengguna lain kana mencoba mengubah dan menulis mount point yang baru. Seperti biasa saat ingin mengutak-atik berkas konfigurasi seperti mengubah isi berkas fstab, pastikan untuk membuat berkas cadangan untuk mencegah terjadinya kesalahan teknis yang dapat menyebabkan suatu kekacauan. Kita dapat melakukannya dengan cara menyediakan sebuah disket atau recovery-disk dan mem-back-up berkas fstab tersebt sebelum membukanya di editor teks untuk diutak-atik. Pada linux, isi sebuah berkas dibuat nyata tersedia dengan mengabungkan sistem berkas ke dalam sebuah sistem direktori melalui sebuah proses yang disebut mounting.

2.5.2 Mounting Overview

Mounting membuat sistem berkas, direktori, piranti dan berkas lainnya menjadi dapat digunakan digunakan di lokasi-lokasi tertentu, sehingga memungkinkan direktori itu menjadi dapat diakses. Perintah mount mengintruksikan sistem operasi untuk mengkaitkan sebuah sistem berkas ke sebuah direktori khusus.

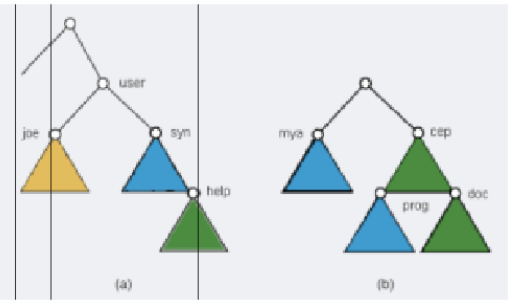
2.5.3 Mounting point

Mount point adalah sebuah direktori dimana berkas baru menjadi dapat diakses. Untuk me-mount suatu berkas atau direktori, titik mountnya harus berupa direktori,dan untuk me-mount sebuah berkas,mount pointnya juga harus berupa berkas.

Ada dua jenis mounting, yaitu remote mounting dan mounting lokal. Remote mounting dilakukan dengan sistem remote dimana data dikirimkan melalui jalur komunikasi. Network sistem berkas seperti Network File System (NFS), menharuskan agar file diekspor dulu sebelum dimount. Mounting lokal dilakukan di sistem lokal.

Biasanya, sebuah sistem berkas, direktori, atau sebuah berkas di-mount ke sebuah mount point yang kosong, tapi biasanya hal tersebut tidak diperlukan. Jika sebuah berkas atau direktori yang akan menjadi mount point berisi data, data tersebut tidak akan dapat diakses selama direktori/berkas tersebut sedang dijadikan mount point oleh berkas atau direktori lain. Sebagai akibatnya, berkas yang di-mount akan menimpa apa yang sebelumnya ada di direktori/berkas tersebut. Data asli dari direktori itu dapat diakses kembali bila proses mounting sudah selesai.

Saat sebuah sistem berkas di-mount ke sebuah direktori, izin direktori root dari berkas yang di-mount akan mengambil alih izin dari mount point. Pengecualiannya adalah pada direktori induk akan memiliki atribut .. (double dot). Agar sistem operasi dapat mengakses sistem berkas yang baru, direktori induk dari mount point harus tersedia.



Gambar 2.5: Mount Point

2.5.4 Mounting Sistem Berkas, Direktori, dan Berkas

Ada dua jenis mounting: remote mounting dan mounting lokal. Remote mounting dilakukan dengan sistem remote dimana data dikirimkan melalui jalur telekomunikasi. Remote sistem berkas seperti Network File Systems (NFS), mengharuskan agar file diekspor dulu sebelum di-mount. mounting lokal dilakukan di sistem lokal.

Tiap-tiap sistem berkas berhubungan dengan piranti yang berbeda. Sebelum kita menggunakan sebuah sistem berkas, sistem berkas tersebut harus dihubungkan dengan struktur direktori yang ada (dapat root atau berkas yang lain yang sudah tersambung). Sebagai contoh, kita dapat me-mount dari /home/server/database ke mount point yang dispesifikasikan sebagai /home/user1, /home/user2, and /home/user3 :

- /home/server/database /home/user1
- /home/server/database /home/user2
- /home/server/database /home/user3

Bab 3

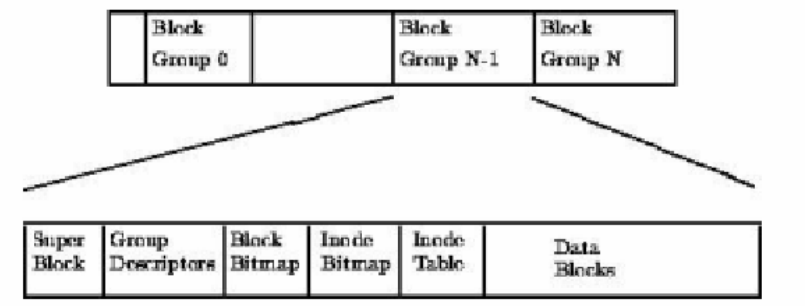
Sistem File Linux

3.1 Sistem File EXT2

1. Keterangan

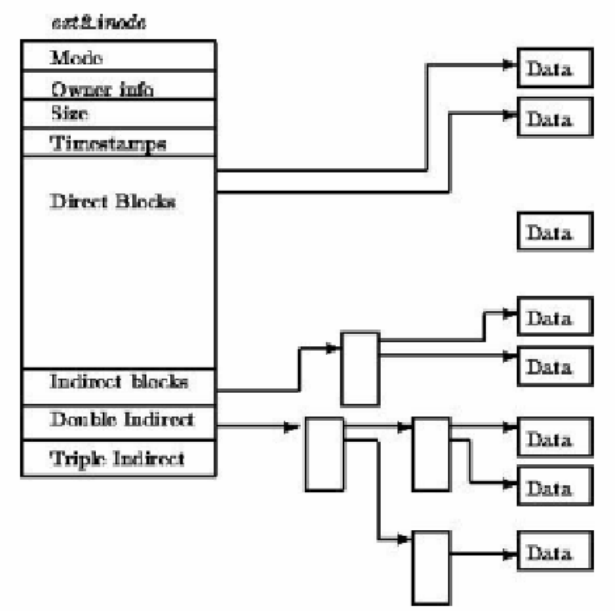
   EXT2 adalah file sistem yang ampuh di linux. EXT2 juga merupakan salah satu file sistem yang paling ampuh dan menjadi dasar dari segala distribusi linux. Pada EXT2 file sistem, file data disimpan sebagai data blok. Data blok ini mempunyai panjang yang sama dan meski pun panjangnya bervariasi diantara EXT2 file sistem, besar blok tersebut ditentukan pada saat file sistem dibuat dengan perintah mk2fs. Jika besar blok adalah 1024 bytes, maka file dengan besar 1025 bytes akan memakai 2 blok. Ini berarti kita membuang setengah blok per file.

   EXT2 mendefinisikan topologi file sistem dengan memberikan arti bahwa setiap file pada sistem diasosiasiakan dengan struktur data inode. Sebuah inode menunjukkan blok mana dalam suatu file tentang hak akses setiap file, waktu modifikasi file, dan tipe file. Setiap file dalam EXT2 file sistem terdiri dari inode tunggal dan setiap inode mempunyai nomor identifikasi yang unik. Inode-inode file sistem disimpan dalam tabel inode. Direktori dalam EXT2 file sistem adalah file khusus yang mengandung pointer ke inode masing-masing isi direktori tersebut.



Gambar 3.1: Blocks dalam filesystem

2. Inode dalam EXT2



Gambar 3.2: Inode dalam ext2

Inode adalah kerangka dasar yang membangun EXT2. Inode dari setiap kumpulan blok disimpan dalam tabel inode bersama dengan peta bit yang menyebabkan sistem dapat mengetahui inode mana yang telah teralokasi dana inode mana yang belum. MODE: mengandung dia informasi, inode apa dan izin akses yang dimiliki user. OWNER INFO: user atau grop yang memiliki file atau direktori SIZE: besar file dalam bytes TIMESTAMPS: kapan waktu pembuatan inode dan waktu terakhir dimodifikasi. DATABLOKS: pointer ke blok yang mengandung data.

EXT2 inode juga dapat menunjuk pada device khusus, yang mana device khusus ini bukan merupakan file, tatapi dapat menangani program sehingga program dapat mengakses ke device. Semua file device di dalam drektori /dev dapat membantu program mengakses device.

3.2 Sistem File EXT3

EXT3 adalah peningkatan dari EXT2 file sistem. Peningkatan ini memiliki beberapa keuntungan, diantaranya:

- Setelah kegagalan sumber daya, "unclean shutdown", atau kerusakan sistem, EXT2 file sistem harus melalui proses pengecekan dengan program e2fsck. Proses ini dapat membuang waktu sehingga proses booting menjadi sangat lama, khususnya untuk disk besar yang mengandung banyak sekali data. Dalam proses ini, semua data tidak dapat diakses.
  yang disediakan oleh EXT3 menyebabkan tidak perlu lagi dilakukan pengecekan data setelah kegagalan sistem. EXT3 hanya dicek bila ada kerusakan hardware seperti kerusakan hard disk, tetapi kejadian ini sangat jarang. Waktu yang diperlukan EXT3 file sistem setelah terjadi "unclean shutdown" tidak tergantung dari ukuran file sistem atau banyaknya file, tetapi tergantung dari besarnya jurnal yang digunakan untuk menjaga konsistensi. Besar jurnal default memerlukan waktu kira-kira sedetik untuk pulih, tergantung kecepatan hardware.
- Integritas data
  EXT3 menjamin adanya integritas data setelah terjadi kerusakan atau "unclean shutdown". EXT3 memungkinkan kita memilih jenis dan tipe proteksi dari data.
- Kecepatan
  Daripada menulis data lebih dari sekali, EXT3 mempunyai throughput yang lebih besar daripada EXT2 karena EXT3 memaksimalkan pergerakan head hard disk. Kita bisa memilih tiga jurnal mode untuk memaksimalkan kecepatan, tetapi integritas data tidak terjamin.
- Mudah dilakukan migrasi
  Kita dapat berpindah dari EXT2 ke sistem EXT3 tanpa melakukan format ulang.

3.3 Perbandingan ext2 dengan ext3

- Secara umum prinsip-prinsip dalam EXT2 sama dengan EXT3. Metode pengaksesan file, keamanan data, dan penggunaan disk space antara kedua file system ini hampir sama.
- Perbedaan mendasar antara kedua file system ini adalah konsep

journaling file system yang digunakan pada EXT3.

- Konsep journaling ini menyebabkan ext2 dan ext3 memiliki Perbedaan perbedaan dalam hal daya tahan dan pemulihan data dari kerusakan.
- Konsep journaling ini menyebabkan EXT3 jauh lebih cepat daripada EXT2 dalam melakukan pemulihan data akibat terjadinya kerusakan.

3.4 Sistem File Reiser

Reiser file sistem memiliki jurnal yang cepat. Ciri-cirinya mirip EXT3 file sistem. Reiser file sistem dibuat berdasarkan balance tree yang cepat. Balance tree unggul dalam hal kinerja, dengan algoritma yang lebih rumit tentunya.

Reiser file sistem lebih efisien dalam pemenfaatan ruang disk. Jika kita menulis file 100 bytes, hanya ditempatkan dalam satu blok. File sistem lain menempatkannya dalam 100 blok. Reiser file sistem tidak memiliki pengalokasian yang tetap untuk inode. Resier file sistem dapat menghemat disk sampai dengan 6 persen.

3.5 NILFS [1]

Linux dalam perkembangannya untuk meningkatkan kemampuannya dari filesystem yang telah ada, diciptakan filesystem baru yang stabil dan menawarkan performa lebih baik dari filesystem UFS milik solaris. Filesystem yang benama NILFS 1.0 (New Implementation Of a Log-Structured File system) ini sudah sediakan oleh NTT Labs (Nippon Telegraph and Telephone's Cyber Space Laboratories).

Maksud dari log-structured adalah menuliskan semua data dalam format mirip log yang ditulis berkelanjutan, dan dalam penulisannya hanya menambahkan baris-baris yang baru ditulis, bukan untuk ditulis ulang dari awal secara keseluruhan. Pendekatan ini dikatakan dapat

[1] Artikel infolinux edisi November 2005

mengurangi waktu pencarian data selain itu dapat pula meminimalisasi seperti halnya kehilangan data yang sering terjadi pada filesytem-filesytem yang konvensional Selain itu NILFS dapat juga membuat dan menyimpan file yang berukuran raksasan.

Bab 4

Filesystem Hierarchy Standard

4.1 Sistem File Root

Linux memetakan file system dalam satu hierarki yang disebut root (/). Partisi lainnya akan dipetakan (mount) sebagai directory di bawah root. Hal ini menguntungkan karena file system yang dapat dimount menjadi tidak terbatas. Windows melakukan pemetaan dalam drive yang terbagi menjadi A: dan B: untuk floppy drive, dan C: hingga Z: untuk partisi disk baik lokal maupun hasil mapping network.

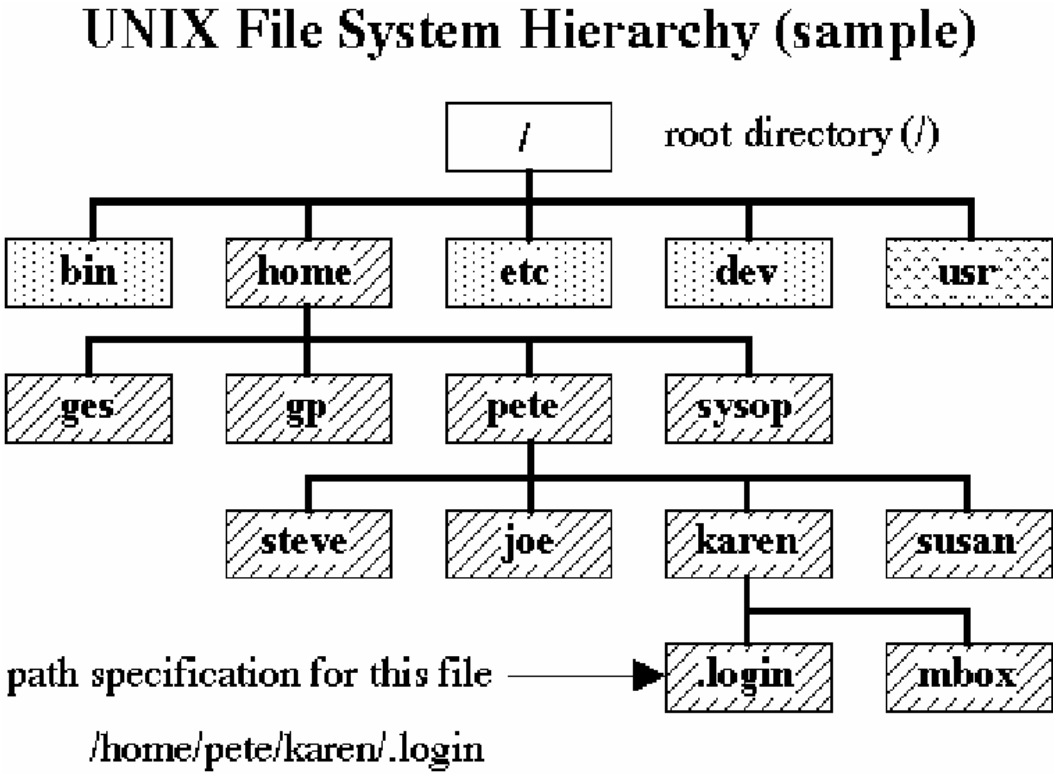
Isi dari sistem berkas root harus memadai untuk melakukan operasi boot, restore, recover, dan atau perbaikan pada sistem. Untuk melakukan operasi boot pada sistem, perlu dilakukan hal-hal untuk mounting sistem berkas lain. Hal ini meliputi konfigurasi data, informasi boot loader dan keperluan-keperluan lain yang mengatur start-up data. untuk melakukan recovery dan perbaikan dari sistem, hal-hal yang dibutuhkan untuk mendiagnosa dan memulihkan sistem yang rusak harus dilakukan dalam sistem berkas root.

Untuk restore suatu sistem, hal-hal yang dibutuhkan untuk back-up sistem, seperti floopy disk, tape, dsb harus berada dalam sistem berkas root. Aplikasi pada komputer tidak diperbolehkan untuk membuat berkas atau subdirektori di dalam direktori root, karena untuk meningkatkan kemampuan dan keamanan, partisi root sebaiknya dibuat seminimum mungkin. Selain itu, lokasi-lokasi lain dalam FHS menyediakan fleksibelitas yang lebih dari cukup untuk package manapun.

Direktori root Linux memiliki beberapa direktori yang merupakan standar direktori pada banyak distro Linux. Direktori-direktori tersebut antara lain :

Direktori	Isi
/bin	berisi file-file binary standar yang dapat digunakan oleh seluruh user baik user biasa maupun super user
/boot	berisi file-file yang digunakan untuk booting Linux termasuk kernel image
/dev	berisi file system khusus yang merupakan refleksi device hardware yang dikenali dan digunakan sistem
/etc	berisi file-file konfigurasi sistem, biasanya hanya boleh diubah oleh super user
/home	berisi direktori-direktori yang merupakan direktori home untuk user biasa dan aplikasi tertentu
/lib	berisi file-file library yang digunakan untuk mendukung kerja kernel Linux
/mnt	direktori khusus yang disediakan untuk mounting (mengaitkan) device disk storage ke sistem dalam bentuk direktori
/proc	berisi file system khusus yang menunjukkan data-data kernel setiap saat
/root	direktori home untuk user root (user khusus dengan priviledges hampir tak terbatas)
/sbin	sama seperti direktori bin, tetapi hanya super user yang sebaiknya menggunakan binary-binary tersebut mengingat fungsi-fungsi binary yang terdapat di direktori ini untuk maintenance sistem
/tmp	berisi file-file sementara yang dibutuhkan sebuah aplikasi yang sedang berjalan
/usr	berisi library, binary, dokumentasi dan file lainnya hasil instalasi user
/var	berisi file-file log, mailbox dan data-data aplikasi

Tabel 4.1: Direktori Linux



Gambar 4.1: Contoh Filesystem hierarchy pada UNIX

4.2 Hirarki /usr

"/usr" adalah bagian utama yang kedua dari sistem berkas. "/usr" bersifat *shareable* dan *read-only*. Hal ini berarti "/usr" bersifat *shareable* diantara bermacam-macam host FHS-*compliant*, dan tidak boleh di-*write*. Package perangkat lunak yang besar tidak boleh membuat subdirektori langsung di bawah hirarki "/usr" ini.

4.2.1 Persyaratan

Direktori	Keterangan
bin	Sebagian besar perintah pengguna
include	Berkas *header* yang termasuk dalam program-program C
lib	Pustaka
local	Hirarki lokal (kosong sesudah instalasi main)
sbin	Sistem biner non-vital
share	Data arsitektur yang independen

Tabel 4.2: Direktori/link yang dibutuhkan dalam "/usr"

4.2.2 Pilihan spesifik

Direktori	**Keterangan**
X11R6	Sistem X Window, Versi 11 Release 6
games	Games dan educational biner
lib	Pustaka
lib (qual)	Format pustaka alternatif
src	Kode source

Tabel 4.3: Direktori/link yang merupakan pilihan dalam "/usr"

Penjelasan

- **"/usr/X11R6": Sistem X Window, Versi 11 Release 6**
  Hirarki ini disediakan untuk Sistem X Window, Versi 11 Release 6 dan berkas-berkas yang berhubungan. Untuk menyederhanakan persoalan dan membuat XFree86 lebih kompatibel dengan Sistem X Window, link simbolik di bawah ini harus ada jika terdapat direktori "/usr/X11R6":
  - /usr/bin/X11 → /usr/X11R6/bin
  - usr/lib/X11 → /usr/X11R6/lib/X11
  - usr/include/X11 → /usr/X11R6/include/X11

  Link-link di atas dikhususkan untuk kebutuhan dari pengguna saja, dan perangkat lunak tidak boleh di-install atau diatur melalui link-link tersebut.
- **"/usr/bin": Sebagian perintah pengguna**
  Direktori ini adalah direktori primer untuk perintah- perintah executable dalam sistem.
- **"/usr/include": Direktori untuk include-files standar**
  Direktori ini berisi penggunaan umum berkas include oleh sistem, yang digunakan untuk bahasa pemrograman C.
- **"/usr/lib": Pustaka untuk pemrograman dan *package***
  "/usr/lib" meliputi berkas obyek, pustaka dan biner internal yang tidak dibuat untuk dieksekusi secara langsung melalui pengguna atau shell script. Aplikasi-aplikasi dapat menggunakan subdirektori tunggal di bawah "/usr/lib". Jika aplikasi tersebut menggunakan subdirektori, semua data yang arsitektur-dependent yang digunakan oleh aplikasi tersebut, harus diletakkan dalam subdirektori tersebut juga.

  Untuk alasan historis, "/usr/lib/sendmail" harus merupakan link simbolik ke "/usr/sbin/sendmail". Demikian juga, jika "/lib/X11" ada, maka "/usr/lib/X11" harus merupakan link simbolik ke "/lib/X11", atau ke mana pun yang dituju oleh link simbolik "/lib/X11".
- **"/usr/lib (qual)": Format pustaka alternatif**
  "/usr/lib(qual)" melakukan peranan yang sama seperti "/usr/lib" untuk format biner alternatif, namun tidak lagi membutuhkan link simbolik seperti "/usr/lib(qual)/sendmail" dan "/usr/lib(qual)/X11".

- **"/usr/local/share"**
  Direktori ini sama dengan "/usr/share". Satu-satunya pembatas tambahan adalah bahwa direktori '/usr/local/share/man' dan '/usr/local/man' harus synonomous (biasanya ini berarti salah satunya harus merupakan link simbolik).
- **"/usr/sbin": Sistem biner standar yang non-vital**
  Direktori ini berisi biner non-vital mana pun yang digunakan secara eksklusif oleh administrator sistem. Program administrator sistem yang diperlukan untuk perbaikan sistem, mounting "/usr" atau kegunaan penting lainnya harus diletakkan di "/sbin".
- **"/usr/share": Data arsitektur independen**
  Hirarki "/usr/share" hanya untuk data-data arsitektur independen yang read-only. Hirarki ini ditujukan untuk dapat di-share diantara semua arsitektur platform dari sistem operasi; sebagai contoh: sebuah site dengan platform i386, Alpha dan PPC dapat me-maintain sebuah direktori /usr/share yang di-mount secara sentral.

  Program atau paket mana pun yang berisi dan memerlukan data yang tidak perlu dimodifikasi harus menyimpan data tersebut di "/usr/share" (atau "/usr/local/share", apabila di- install secara lokal). Sangat direkomendasikan bahwa sebuah subdirektori digunakan dalam /usr/share untuk tujuan ini.
- **"/usr/src": Kode source**
  Dalam direktori ini, dapat diletakkan kode-kode source, yang digunakan untuk tujuan referensi.

4.3 Hirarki "/var"

"/var" berisi berkas data variabel, meliputi berkas dan direktori spool, data administratif dan logging, serta berkas transient dan temporer. Beberapa bagian dari "/var" tidak shareable diantara sistem yang berbeda, antara lain: "/var/log", "/var/lock" dan "/var/run". Sedangkan, "/var/mail", "/var/cache/man", "/var/cache/fonts" dan "/var/spool/news" dapat di-share antar sistem yang berbeda.

"/var" ditetapkan di ini untuk memungkinkan operasi mount "/usr" read-only. Segala sesuatu yang melewati "/usr", yang telah ditulis se-

lama operasi sistem, harus berada di ”/var”. Jika ”/var” tidak dapat dibuatkan partisi yang terpisah, biasanya ”/var” dipindahkan ke luar dari partisi root dan dimasukkan ke dalam partisi ”/usr”.

Bagaimana pun, ”/var” tidak boleh di-link ke ”/usr”, karena hal ini membuat pemisahan antara ”/usr” dan ”/var” semakin sulit dan biasa menciptakan konflik dalam penamaan. Sebaliknya, buat link ”/var” ke ”/usr/var”.

Direktori	**Keterangan**
cache	Data *cache* aplikasi
lib	Informasi status variabel
local	Data variabel untuk "/usr/local"
lock	*Lock*berkas
log	Berkas dan direktori *log*
opt	Data variabel untuk "/opt"
run	Relevansi data untuk menjalankan proses
spool	Aplikasi data *spool*
tmp	Berkas temporer yang disimpan di dalam *reboot* sistem

Tabel 4.4: Direktori/link yang dibutuhkan dalam ”/var”

Penjelasan

- ”/var/cache”: Aplikasi data cache
  ”/var/cache” ditujukan untuk data cache dari aplikasi. Data tersebut diciptakan secara lokal sebagai time-consuming M/K atau kalkulasi. Aplikasi ini harus dapat menciptakan atau mengembalikan data. Tidak seperti ”/var/spool”, berkas cache dapat dihapus tanpa kehilangan data. Berkas yang ditempatkan di bawah ”/var/cache” dapat expired oleh karena suatu sifat spesifik dalam aplikasi, oleh administrator sistem, atau keduanya, maka aplikasi ini harus dapat recover dari penghapusan berkas secara manual.
- ”/var/lib”: Informasi status variabel
  Hirarki ini berisi informasi status suatu aplikasi dari sistem. Yang dimaksud dengan informasi status adalah data yang dimodifikasi program saat program sedang berjalan. Pengguna tidak diperbolehkan untuk memodifikasi berkas di ”/var/lib” untuk mengkonfigurasi operasi package. Informasi status ini digunakan untuk memantau kondisi dari aplikasi, dan harus tetap valid setelah reboot,

tidak berupa output logging atau pun data spool. Sebuah aplikasi harus menggunakan subdirektory "/var/lib" untuk data-datanya. Terdapat satu subdirektori yang dibutuhkan lagi, yaitu "/var/lib/misc", yang digunakan untuk berkas-berkas status yang tidak membutuhkan subdirektori.

- "/var/lock": Lock berkas
  Lock berkas harus disimpan dalam struktur direktori /var/lock. Lock berkas untuk peranti dan sumber lain yang di-share oleh banyak aplikasi, seperti lock berkas pada serial peranti yang ditemukan dalam "/usr/spool/locks" atau "/usr/spool/uucp", sekarang disimpan di dalam "/var/lock". Format yang digunakan untuk isi dari lock berkas ini harus berupa format lock berkas HDB UUCP. Format HDB ini adalah untuk menyimpan pengidentifikasi proses (Process Identifier - PID) sebagai 10 byte angka desimal ASCII, ditutup dengan baris baru. Sebagai contoh, apabila proses 1230 memegang lock berkas, maka HDO formatnya akan berisi 11 karakter: spasi, spasi, spasi, spasi, spasi, spasi, satu, dua, tiga, nol dan baris baru.
- "/var/log": Berkas dan direktori log
  Direktori ini berisi bermacam-macam berkas log. Sebagian besar log harus ditulis ke dalam direktori ini atau subdirektori yang tepat.
- "/var/opt": Data variabel untuk "/opt"
  Data variabel untuk paket di dalam "/opt" harus di-install dalam "/var/opt/¡subdir¿", di mana ¡subdir¿ adalah nama dari subtree dalam "/opt" tempat penyimpanan data statik dari package tambahan perangkat lunak.
- "/var/run": Data variabel run-time
  Direktori ini berisi data informasi sistem yang mendeskripsikan sistem sejak di boot. Berkas di dalam direktori ini harus dihapus dulu saat pertama memulai proses boot. Berkas pengidentifikasi proses (PID), yang sebelumnya diletakkan di "/etc", sekarang diletakkan di "/var/run". Program yang membaca berkas-berkas PID harus fleksibel terhadap berkas yang diterima, sebagai contoh: program tersebut harus dapat mengabaikan ekstra spasi, baris-baris tambahan, angka nol yang diletakkan di depan, dll.
- "/var/spool": Aplikasi data spool

"/var/spool" berisi data yang sedang menunggu suatu proses. Data di dalam "/var/spool" merepresentasikan pekerjaan yang harus diselesaikan dalam waktu depan (oleh program, pengguna atau administrator); biasanya data dihapus sesudah selesai diproses.

- "/var/tmp": Berkas temporer ang diletakkan di dalam reboot sistem

  Direktori "/var/tmp" tersedia untuk program yang membutuhkan berkas temporer atau direktori yang diletakkan dalam reboot sistem. Karena itu, data yang disimpan di "/var/tmp" lebih bertahan daripada data di dalam "/tmp". Berkas dan direktori yang berada dalam "/var/tmp" tidak boleh dihapus saat sistem di-boot. Walaupun data-data ini secara khusus dihapus dalam site-specific manner, tetap direkomendasikan bahwa penghapusan dilakukan tidak sesering penghapusan di "/tmp".

Bab 5

Implementasi Sistem Berkas

Untuk mengimplementasikan suatu sistem berkas biasanya digunakan beberapa struktur on-disk dan in-memory. Struktur ini bervariasi tergantung pada sistem operasi dan sistem berkas, tetapi beberapa prinsip dasar harus tetap diterapkan. Pada struktur on-disk, sistem berkas mengandung informasi tentang bagaimana mem-boot sistem operasi yang disimpan, jumlah blok, jumlah dan lokasi blok yang masih kosong, struktur direktori, dan berkas individu.

Setelah berkas selesai dibuat, mula-mula harus dibuka terlebih dahulu. Perintah open mengirim nama berkas ke sistem berkas. Ketika sebuah berkas dibuka, struktur direktori mencari nama berkas yang diinginkan. Ketika berkas ditemukan, FCB (File Control Block) disalin ke ke tabel system-wide open-file pada memori. Tabel ini juga mempunyai entri untuk jumlah proses yang membuka berkas tersebut.

5.1 Partisi dan mounting

Informasi boot dapat disimpan di partisi yang berbeda. Semuanya mempunyai formatnya masing-masing karena pada saat boot, sistem tidak punya sistem berkas dari perangkat keras dan tidak dapat memahami sistem berkas.

Root partition yang mengandung kernel sistem operasi dan sistem berkas yang lain, di-mount saat boot. Partisi yang lain di-mount secara otomatis atau manual (tergantung sistem operasi). Sistem operasi menyimpan dalam struktur tabel mount dimana sistem berkas di-mount dan jenis dari sistem berkas.

Pada Unix, sistem berkas dapat di-mount di direktori mana pun. Ini diimplementasikan dengan mengatur flag di salinan in-memory dari

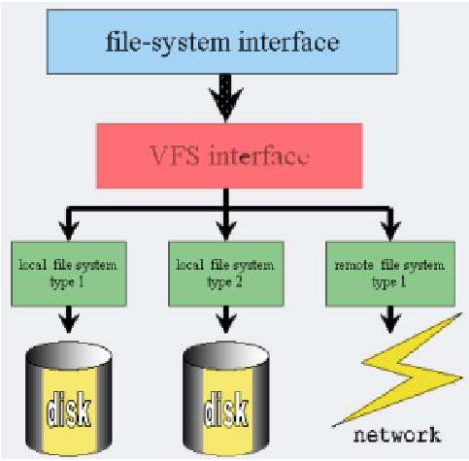
jenis direktori itu. Flag itu mengindikasikan bahwa direktori adalah lokasi mount.

5.2 Sistem file virtual

Sistem operasi Unix menggunakan teknik berorientasi obyek untuk menyederhakan, mengorganisir dan mengelompokkannya sesuai dengan implementasinya. Penggunaan metode ini memungkinkan berkas-berkas yang berbeda jenisnya diimplementasikan dalam struktur yang sama. Implementasi spesifiknya menggunakan struktur data dan prosedur untuk mengisolasi fungsi dasar dari system call.

Virtual File System (VFS) memiliki fungsi yang penting yaitu :

- Memisahkan operasi berkas generic dari implementasinya dengan mendefinisikan VFS antar muka yang masih baru.
- VFS didasarkan pada struktur file-representation yang dinamakan vnode, yang terdiri dari designator numerik untuk berkas unik network-wide.
- Sistem berkas lokal dan sistem berkas remote untuk jaringan.



Gambar 5.1: Schematic View of Virtual File System

Bab 6

Kesimpulan

Di dalam sebuah sistem operasi, salah satu hal yang paling penting adalah sistem berkas. Sistem berkas ini muncul karena ada tiga masalah utama yang cukup signifikan: kebutuhan untuk menyimpan data dalam jumlah yang besar, kebutuhan agar data tidak mudah hilang (non-volatile), dan informasi harus berdiri sendiri tidak bergantung pada proses. Pada sistem berkas ini, diatur segala rupa macam yang berkaitan dengan sebuah berkas mulai dari atribut, tipe, operasi, sampai metoda akses berkas. Banyak sekali berkas-berkas yang disimpan dalam suatu disk, karena itu diperlukan suatu struktur untuk pengorganisasian berkas tersebut.

Setiap sistem file di Linux mempunyai kelebihan dan kekurangan. Pada penjelasan bab tentang sistem file mengindikasikan makin berkembangkannya sistem file maka ada suatu perbaikan seperti sistem file sebelumnya seperti sistem file EXT3 lebih berkembang dibandingkan sistem file EXT2. Reiser lebih unggul dari EXT3. Itu semua dilihat dari perkembangna zaman teknologi waktu itu

Standar Hirarki Sistem Berkas (File Hierarchy Standard) adalah rekomendasi penulisan direktori dan berkas-berkas yang diletakkan di dalamnya. FHS ini digunakan oleh perangkat lunak dan user untuk menentukan lokasi dari berkas dan direktori.

Daftar Pustaka

[1] Masyarakat Digital Gotong Royong. 2005 *Pengantar Sistem Operasi plus ilustrasi kernel linux*, Depok : MDGR Fasilkom Universitas Indonesia

[2] Novriansyah, Nova., 1996, *LINUX*, Jakarta : Gramedia

[3] Samik, Rahman M-Ibrahim. 2004, *Pengantar Sistem Operasi Komputer Plus Studi Kasus kernel Linux*, Depok : MDGR Fasilkom Universitas Indonesia

[4] Suadi, Wahyu. 1995, *Sistem Berkas Terkompres, Implementasi dan Analisa*, Depok : Fasilkom Universitas Indonesia

[5] http://ariya.pandu.org/linux/feature/fileman/fileman.htm

[6] http://www.galihsatria.com/galih/faces/sisop.js

[7] http://bebas.vlsm/v06/kuliah/SistemOperasi/Buku/

[8] http://group.yahoo.com/group/sistemoperasi/