Tugas Besar 2

IF2230 Sistem Operasi

Implementasi User-Space File System dengan FUSE

Versi 1.0

Dipersiapkan oleh:

SISTER Assistants

Knightmare Frame R&D Team IT Division

Didukung oleh:

START: 16 April 2014

END: 7 Mei 2014

A. Story Background

Selamat datang kembali, peserta magang tim R&D Knightmare Frame! Anda telah berhasil menyelesaikan tugas sebelumnya, yaitu implementasi system call. Kali ini, anda akan dikerahkan untuk membantu pemerintah dalam pengembangan filesystem.

Seperti yang anda ketahui, minggu lalu Wakil Gubernur Area 11, Yang Mulia Euphemia li Britannia, telah mencanangkan program kependudukan Britannia Special Administrative District of Area 11. Salah satu aktivitas dalam program ini adalah produksi kartu identitas penduduk dengan memory storage khusus di dalamnya. Oleh karena itu, akan dibuat suatu filesystem untuk mengatur memory storage tersebut, dengan kode nama CCFS.

B. Deskripsi Tugas

Pada tugas kali ini, anda diminta untuk membuat filesystem bernama CCFS. Pembuatan CCFS ini dilakukan dengan FUSE API, yaitu API yang memungkinkan pembuatan filesystem pada user-space. CCFS diimplementasi dalam bahasa C atau C++, dengan compiler gcc / g++ dan sistem operasi Linux.

CCFS sendiri adalah sebuah filesystem yang di-mount dari sebuah file berformat .ccfs (dapat dianalogikan dengan mounting file .iso). Sehingga, setiap file dan folder yang ada pada volume CCFS sebenarnya disimpan dalam sebuah file berformat .ccfs. Deskripsi program dan format file dijelaskan pada bagian berikutnya.

Pembuatan filesystem dengan FUSE dilakukan dengan implementasi beberapa fungsi filesystem, seperti getattr, read, write dan sebagainya. Fungsi-fungsi yang wajib diimplementasikan adalah:

1. getattr
2. open
3. read
4. readdir
5. write
6. truncate
7. mkdir
8. rmdir
9. rename
10. unlink

Adapun jika anda mengimplementasi fungsi-fungsi lainnya akan dinilai sebagai **bonus**.

C. Deskripsi Program

CCFS diimplementasikan dengan membuat sebuah program bernama ccfs-mount. Program ini dapat melakukan dua hal, (1) melakukan mounting file .ccfs, dan (2) membuat file .ccfs baru dan melakukan mounting file baru tersebut. Proses mounting dilakukan dengan command berikut pada terminal.

```
./ccfs-mount hello_dir hello.ccfs
```

Sehingga, setelah command tersebut dijalankan, file hello.ccfs akan di-mount pada folder hello_dir. Proses pembuatan file .ccfs baru (dan otomatis melakukan mounting) dilakukan dengan command berikut pada terminal.

```
./ccfs-mount hello_dir hello.ccfs --new
```

Sehingga, setelah command tersebut dijalankan, sebuah file hello.ccfs baru akan dibuat dan dimount pada folder hello_dir. Proses unmounting dilakukan dengan command berikut.

```
fusermount -u hello_dir
```

Catatan: program fusermount sudah terinstall bersama FUSE.

D. Deskripsi Format File .ccfs

Setiap pembacaan dan penulisan pada file .ccfs dilakukan dengan satuan blok, dengan ukuran 1 blok 512 byte.

Volume Information (1 blok)
Allocation Table (256 blok)
Data Pool (65536 blok)

File .ccfs sendiri terdiri dari 4 bagian utama. Volume information berukuran 1 blok, allocation table berukuran 256 blok (128 KB), dan data pool berukuran 65536 blok (32 MB). Sehingga total blok yang terdapat pada sebuah file .ccfs adalah 65753 blok. Bagian-bagian tersebut dijelaskan pada bagian berikut.

Volume Information

Volume information berisi tentang informasi umum volume / partisi filesystem. Volume information berukuran 1 blok (512 KB). Berikut data yang terdapat pada volume information beserta ukurannya dan offset (jarak) dari awal blok volume information.

Offset	Ukuran (byte)	Deskripsi
0x00	4	Magic string “CCFS” (tanpa kutip). Magic string ini digunakan untuk melakukan validasi bahwa file yang dibaca adalah file .ccfs
0x04	32	Sebuah null-terminated string yang menyimpan nama dari volume. Karakter yang digunakan ditulis dalam format yang dapat dibaca oleh manusia. Jika tak diisikan maka secara default akan berisi “CCFS”.
0x23	4	Kapasitas filesystem dalam blok, ditulis dalam integer 32-bit little endian.
0x28	4	Jumlah blok yang belum terpakai, ditulis dalam integer 32-bit little endian.
0x2C	4	Indeks blok pertama yang bebas, ditulis dalam integer 32-bit little endian.
0x30	460	Tidak digunakan, diisi nilai 0 / karakter null.
0x1FC	4	String “SFCC” (tanpa kutip).

Allocation Table

Allocation table terletak tepat setelah volume information, dengan ukuran 256 blok. Allocation table terdiri dari 65536 word (1 word = 2 byte), dengan masing-masing word menyatakan *pointer to next block* untuk setiap blok pada data pool. Misalnya, next block dari blok 0 pada data pool adalah blok 5, maka pointer ke-0 adalah nilai 5 (dalam integer 16-bit little endian).

Terdapat nilai khusus untuk setiap pointer. Nilai 0x00 menandakan bahwa blok tersebut kosong, sedangkan nilai 0xFF menandakan bahwa blok tersebut tidak memiliki next. Pointer pertama (ke-0) selalu digunakan untuk blok root (blok ke-0 pada data pool) dan pointer terakhir (ke-65535, atau 0xFF) tidak digunakan.

Data Pool

Data pool adalah tempat seluruh file dan direktori disimpan. Setiap file dan direktori yang disimpan akan dipecah-pecah per blok. Untuk file, isi setiap blok adalah potongan isi file tersebut. Untuk direktori, isi setiap blok adalah susunan entri berukuran 32 byte, dengan format dijelaskan pada bagian berikutnya. Jika byte pertama pada suatu entry berisi nilai 0 / karakter null, maka entri itu dianggap kosong dan merupakan akhir dari isi blok direktori tersebut. Blok pertama (indeks ke-0) pada data pool selalu digunakan sebagai blok pertama direktori root, dan blok terakhir (indeks ke-65535, atau 0xFF) tidak digunakan.

Entri blok direktori berukuran 32 byte memiliki format sebagai berikut

Offset	**Ukuran (byte)**	**Deskripsi**
0x00	21	Nama file atau direktori
0x14	1	Atribut file atau direktori
0x16	2	Waktu pembuatan atau modifikasi terakhir
0x18	2	Tanggal pembuatan atau modifikasi terakhir
0x1A	2	Indeks blok data pool pertama penyimpanan file / direktori tersebut, dalam integer 16-bit little endian
0x1C	4	Ukuran file dalam byte, ditulis dalam integer 32-bit little endian

Atribut sebuah file dalam ukuran 1 byte, ditulis dengan format sebagai berikut

Bit	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Data	0	0	0	0	d	r	w	x

dengan,

Data	**Deskripsi**
0	Tidak digunakan, diisi 0
d	Apakah entri berupa file atau direktori, 1 jika direktori 0 jika file.
r	Read permission, 1 jika boleh 0 jika tidak.

w	Write permission, 1 jika boleh 0 jika tidak.
x	Execute permission, 1 jika boleh 0 jika tidak. Jika d diisi 1, maka x juga harus diisi 1 agar direktori dapat dibuka.

Catatan: MSB adalah Most Significant Bit, dalam hal ini bit paling depan. LSB adalah Least Significant Bit, dalam hal ini bit paling akhir.

Waktu pembuatan atau modifikasi terakhir file dikodekan dalam 2 byte sebagai berikut

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	h	h	h	h	h	m	m	m	m	m	m	x	x	x	x	x

Bilangan hhhhhh menyatakan bilangan biner untuk jam (0-23). Bilangan mmmmmm menyatakan bilangan biner untuk menit (0-59). Bilangan xxxxx menyatakan bilangan biner untuk detik dengan ketentuan setiap bilangan merupakan periode 2 detik (0-29) yang merepresentasikan 0 hingga 58. Dengan kata lain, jika file terakhir kali diupdate pada detik ke 57, maka disimpan sebagai 28 (56 mendekati 57 dalam periode 2 detik).

Tanggal pembuatan atau modifikasi terakhir file dikodekan dalam 2 byte sebagai berikut

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	y	y	y	y	y	y	y	m	m	m	m	d	d	d	d	d

Bilangan yyyyyyy menyatakan bilangan biner (0-119) berupa offset tahun dihitung dari 2000, sehingga bilangan 14 menyatakan tahun 2014. Bilangan mmmm menyatakan bilangan biner untuk bulan (0-11). Bilangan ddddd menyatakan bilangan biner untuk hari (1-31).

E. Deliverables

Hal-hal yang perlu dikumpulkan sebagai deliverables tugas besar 2 ini adalah:

1. Source code, beserta makefile dan petunjuk kompilasinya
2. Executable
3. Laporan, yang berisi
   a. Cover
   b. Asumsi-asumsi yang anda gunakan dalam pengembangan beserta alasan penggunaan asumsi tersebut, jika ada
   c. Pembahasan, yang menjelaskan (minimal pilih 2, jika lebih dinilai **bonus**)
      - Algoritma pencarian blok pertama suatu file / direktori jika diberikan path lengkapnya dari root directory
      - Algoritma pembacaan file / direktori yang panjangnya lebih dari 1 blok
      - Algoritma penulisan file / direktori, jika jumlah bloknya tetap, bertambah atau berkurang, dan bagaimana pengaruhnya terhadap informasi seperti jumlah blok bebas, indeks blok kosong pertama, dan sebagainya.
      - Algoritma penghapusan file / direktori, dan bagaimana pengaruhnya seperti pada poin sebelumnya
   d. Penjelasan singkat fungsi-fungsi lainnya yang diimplementasikan, jika ada
   e. Hal yang paling menarik atau paling sulit dalam pengerjaan tugas besar

F. Pengumpulan Tugas

1. Daftarkan kelompok anda pada http://beat.inkubatorit.com/os2014tb2.
2. Buatlah sebuah zip/rar dengan nama TB1_KX_YY, dengan KX nomor kelas (K1 atau K2) dan YY nomor kelompok sesuai daftar pada link di atas (00, 01, 02, dst). File zip/rar ini berisi
   - Folder src, berisi *source code*, makefile dan petunjuk kompilasi.
   - Folder bin, berisi *executable* ccfs-mount
   - Folder doc, berisi laporan
3. Tugas besar dikumpulkan pada **7 Mei 2014** di situs http://oddyseus.if.itb.ac.id. Jika terdapat perubahan prosedur pengumpulan, akan diumumkan paling lambat H-1 deadline.

G. Referensi Tambahan

CS135 FUSE Documentation.
http://www.cs.hmc.edu/~geoff/classes/hmc.cs135.201109/homework/fuse/fuse_doc.html

CS135 FUSE Hello, World!
http://www.cs.hmc.edu/~geoff/classes/hmc.cs135.201109/homework/fuse/fuse_hello.html