**TUGAS 2 IF2230**

**Sistem Operasi**

Diajukan sebagai tugas dari mata kuliah Sistem Operasi

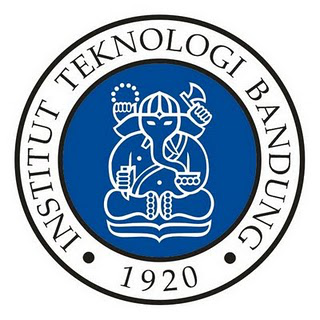
Program Studi Teknik Informatika Institut Teknologi Bandung

Oleh Kelompok 9 :

**Dzar Bela Hanifa (13515007)**

**Robby Syaifullah (13515013)**

**Kevin Erdiza Yogatama (13515016)**

****

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2017**

**Jawaban**

* **Bagaimana program dieksekusi di dalam OS?**

CPU komputer saat ini memiliki 2 mode, yaitu user mode dan kernel mode.

Pada user mode, pengguna tinggal melakukan eksekusi seperti mengklik suatu fungsi pada program, selanjutnya OS akan menjalankan program yang ditunjuk dan programnya tidak bisa berjalan dengan bebas, harus melewati OS dan kernel mode untuk menjalankan suatu job.

Pada kernel mode, ketika kernel mode ingin memulai sebuah proses, pertama akan dialokasikan memori agar bisa memuat (load) program. Termasuk menerima semua instruksi dari binary dan menaymbungkan ke OS.dan biasanya titik masuk harus ada, biasanya int main (int argc, char\*\* argv) dan point point lainnya yang akan dipanggil oleh OS.

Ada beberapa cara untuk menyambungkan program dengan OS menggunakan Application Binary Interface, yakni interface antara dua modul program dimana biasanya adalah library dengan OS dalam level bahasa mesin. Dan ada format binary yang digunakan, diantaranya adalah format PE yang digunakan oleh Microsoft dan format ELF yang digunakan oleh beberapa OS lainnya. Keduanya mendeskripsikan format binary dengan presisi sehingga OS bisa memutuskan program mana yang akan disambungkan ke OS.

* **Apa yang membedakan user program dengan kernel program di OS buatan kalian?**

User program atau aplikasi berjalan di suatu lokasi pada memori yang disebut "user space". User program tidak bisa menngutak-atik memori dan "resource" yang dimiliki oleh program lainnya. Kernel program berjalan di lokasi pada memori yang disebut "kernel space". Kernel program selalu berjalan pada mode super access. Oleh karena itu, wewenang kernel program tidak terbatas. Kernel program dapat mengakses segala hal yang terdapat di OS yang kami buat.

* **Apa yang terjadi jika Anda melakukan execute untuk file yang bukan program? Bagaimana cara OS modern mengatasi hal tersebut?**

Apabila OS mengeksekusi file yang bukan program, maka tidak ada call untuk fungsi terminate, akibatnya hak untuk mengakses shell tidak dimiliki lagi.

Pada modern OS, proses yang dijalankan memiliki lifetime sebagai lamanya program berjalan yang diatur oleh Sistem Operasi. Ketika sebuah file dijalankan, akan diberikan beberapa detik untuk menyimpan state nya lalu pemberian resource dihentikan sampai user memanggil program itu kembali. Inilah yang terjadi pada keaadaan normal apabila sebuah program dieksekusi. Untuk kasus yang dieksekusi ternyata bukan program, sistem operasi tidak akan menerima untuk menjalankan suatu program kecuali sudah lulus verifikasi, yakni akan mencek apakah ada program dengan instruksi binary seperti input user (yang telah diolah).

* **Bagaimana kernel menghindari copy memory dari kernelspace ke user space dan sebaliknya?**

Salah satu cara untuk menghindari terjadinya copy memory dari kernel space ke user space adalah menggunakan fungsionalitas linux yang dinamakan zero copy. Zero copy diimplementasi dengan memanggil system calls map. System call MMap menyebabkan konten suatu file dicopy ke kernel buffer. Buffer ini akan di-share dengan user proses, tanpa melakukan copy antara kernel space dan user space. Dengan menggunakan mmap, jumlah data yang dikopi akan berkurang secara drastis.

* **Apa yang perlu dilakukan untuk meningkatkan file size maximum?**

Batas file size maximum di Linux adalah 2GB. Untuk bisa meningkatkan maximum size tersebut large file support harus diimplementasi. Salah satu cara mengimplementasi large file support adalah meng-compile program c dengan gcc -D\_FILE\_OFFSET\_BITS=64. Implementasi yang umum digunakan adalah menganggap ukuran file sebagai unsigned numbers. Hal tersebut menyebabkan ukuran file akan bertambah dari 2 31-1 (32 GB) menjadi 2 ^ 64 -1 (64 GB)

Modern Operating Systems, 2nd Ed. - Tannenbaum and Bos.

**Kesulitan Saat Pengerjaan**

1. *Debugging*nya cukup sulit.
2. Penanganan karakter null, newline, dan return cukup menyusahkan pengerjaan.
3. *Scope* pengerjaan P4 extended melebihi yang disebutkan di spesifikasi tugas.

**Screenshot**

<isi ya rob>

**Pembagian Tugas**

|  |  |  |  |
| --- | --- | --- | --- |
| Nama | NIM | Apa saja yang dikerjakan | Presentase pengerjaan |
| Dzar Bela Hanifa | 13515007 | P2, P3, P4, laporan | 30 % |
| Robby Syaifullah | 13515013 | P2, P3, P4, laporan | 30 % |
| Kevin Erdiza Yogatama | 13515016 | P4 extended | 40 % |

**Feedback**

1. Formatting *source code* yang disediakin tidak rapi, indentasinya tidak sesuai standar.
2. Materi yang diajarkan di kuliah kurang sesuai dengan materi yang terdapat di tugas besar.

**Referensi**

<http://www.linuxjournal.com/article/6345>