MODUL PRAKTIKUM 01

TOPIK: PENGENALAN LINUX CLI & PELACAKAN SYSTEM CALLS (`strace`)

Mata Kuliah: Sistem Operasi (INF-201) – Alokasi Waktu: 170 Menit

Program Studi Informatika – Fakultas Teknik – Universitas Bengkulu

---

1. CAPAIAN PEMBELAJARAN MODUL (SUB-CPMK 1)

Setelah menyelesaikan Modul Praktikum 01 ini, mahasiswa diharapkan mampu:

1. Mengoperasikan antarmuka *Command Line Interface* (CLI) pada lingkungan sistem operasi Linux.
2. Mengidentifikasi perbedaan mode eksekusi hardware (*User Mode* vs *Kernel Mode*).
3. Menggunakan utilitas pelacak `strace` untuk memantau panggilan sistem (*System Calls*) yang dipanggil oleh aplikasi ke kernel.

2. PRASYARAT TUGAS MEMBACA

Sebelum memulai percobaan laboratorium ini, mahasiswa **wajib** telah membaca bab referensi berikut:

- **[OSTEP]** Chapter 6: *Mechanism: Limited Direct Execution* (hal. 45–58).
- **[TLCL]** Chapter 1: *What Is The Shell?* & Chapter 2: *Navigation* (hal. 1–20).
- **[xv6 RISC-V]** Chapter 1: *Operating System Interfaces* (hal. 7–16).

3. LANDASAN TEORI RINGKAS

3.1 Proteksi Hardware & Dual-Mode Operation

Sistem operasi modern mengimplementasikan proteksi perangkat keras melalui **Dual-Mode Operation**:

- **User Mode (Mode Bit = 1)**: Mode eksekusi tempat aplikasi pengguna berjalan. Aplikasi tidak diizinkan mengakses hardware atau alamat memori kernel secara langsung.
- **Kernel Mode (Mode Bit = 0)**: Mode eksekusi tempat kode kernel Sistem Operasi berjalan, memiliki akses tanpa batas ke seluruh instruksi CPU dan memori utama.

3.2 Limited Direct Execution (LDE) & System Calls

Untuk mencapai efisiensi dan keamanan, OS menggunakan mekanisme **Limited Direct Execution (LDE)**:

1. Program dieksekusi secara langsung di CPU (*Direct Execution*) untuk kecepatan maksimal.
2. Ketika program membutuhkan layanan hardware (misal membaca berkas dari disk), program memicu instruksi *software trap* (**System Call**).
3. CPU beralih dari User Mode ke Kernel Mode, meloncat ke *Trap Handler* kernel yang terdaftar pada *Trap Table*, lalu mengembalikan kendali ke aplikasi setelah selesai.

4. PERINTAH DEMO & KODE ACUAN (EXECUTABLE LISTING)

Utilitas `strace` adalah pelacak panggilan sistem bawaan kernel Linux yang digunakan untuk mencegat dan mencatat panggilan sistem (*system calls*) yang dilakukan oleh suatu proses aplikasi ke kernel.

4.1 Listing 1.1: Merekap Statistik System Calls

Perintah untuk menampilkan tabel statistik panggilan sistem saat menjalankan perintah `ls`:

```
$ strace -c ls -l
```

Listing 1: Rangkuman Statistik System Calls

4.2 Listing 1.2: Filter System Calls Terkait Berkas (I/O)

Perintah untuk melacak secara spesifik panggilan berkas `openat`, `read`, `write`, dan `close`:

```
$ strace -e trace=openat,read,write,close cat /etc/hosts
```

Listing 2: Pelacakan Spesifik System Calls I/O

5. PROSEDUR LANGKAH KERJA PERCOBAAN

Jalankan langkah-langkah percobaan berikut pada terminal Linux (Ubuntu/Debian):

1. Buka terminal emulator Linux.
2. Jalankan perintah `strace -c ls` dan amati output tabel statistik panggilan sistem yang dihasilkan.
3. Buatlah sebuah berkas teks baru bernama `tes.txt` berisi kalimat "Praktikum Sistem Operasi Informatika UNIB":

```
$ echo "Praktikum Sistem Operasi Informatika UNIB" > tes.txt
```

4. Lacak pemanggilan berkas saat membaca `tes.txt` menggunakan `strace` dan simpan keluaran ke berkas log:

```
$ strace -o trace_output.txt cat tes.txt
```

5. Buka berkas log `trace_output.txt` menggunakan `cat` atau `nano`, temukan dan amati baris panggilan `openat`, `read`, dan `write`.

6. PANDUAN TROUBLESHOOTING ERROR UMUM

Pesan Error / Kendala	Penyebab Utama	Solusi Solutif
`strace: command not found`	Utilitas `strace` belum terpasang di sistem Linux.	Jalankan perintah `sudo apt install strace` (Debian/Ubuntu).
`Operation not permitted`	Penebangan ptrace dilarang oleh aturan YAMA PTRACE scope.	Jalankan perintah dengan `sudo strace ...` atau ubah nilai `/proc/sys/kernel/yama/ptrace_scope`.

7. TUGAS PRAKTIKUM & PERTANYAAN ANALISIS LAPORAN

Kerjakan tugas berikut dan cantumkan jawaban analisisnya pada Laporan Resmi Praktikum Anda:

1. **Analisis Kuantitatif**: Berapa jumlah total *system calls* yang dipanggil oleh perintah `ls` dibandingkan dengan perintah `cp`? Mengapa jumlahnya berbeda secara signifikan?
2. **Analisis Alur Mode**: Jelaskan step-by-step apa yang terjadi pada nilai *Mode Bit* CPU dari saat baris kode C `printf("Hello")` dipanggil hingga teks muncul di layar monitor!