| **No** | **Perintah yang Dijalankan** | **Output / Cuplikan Hasil** | **Analisis / Observasi** |
| --- | --- | --- | --- |
| 1 | strace -e trace=open,read,write,close cat /etc/passwd | Menampilkan urutan system call seperti: open("/etc/passwd", O\_RDONLY) read(3, "root:x:0:0:root:/root:/bin/bash\n", 4096) write(1, "root:x:0:0:root:/root:/bin/bash\n", 33) close(3) | Perintah cat membuka file /etc/passwd menggunakan system call open(), membaca isi file dengan read(), menampilkan ke layar melalui write(), lalu menutup file dengan close(). Hal ini menunjukkan interaksi antara program user dengan kernel melalui system call. |
| 2 | `dmesg | tail -n 10` | Menampilkan pesan kernel terakhir, misalnya: [ 7.298721] systemd-journald[43]: Collecting audit messages is disabled. [ 7.526996] systemd-journald[43]: Received client request to flush runtime journal. |
| 3 | strace ls | Menampilkan banyak system call seperti execve(), brk(), mmap(), stat(), openat(), dan close(). | Saat menjalankan ls, kernel memanggil berbagai system call untuk mengakses direktori, membaca metadata file, dan menampilkan daftar file ke layar. Ini menggambarkan proses interaksi intensif antara user space dan kernel. |
| 4 | `dmesg | grep error` | Menampilkan pesan error jika ada, misalnya: [ 15.234512] usb 1-1: device descriptor read/64, error -71 |

System call penting untuk keamanan sistem operasi karena:

Membatasi akses langsung ke hardware — Program user tidak bisa langsung mengakses perangkat keras, semua harus melalui system call yang dikontrol oleh kernel. Ini mencegah program berbahaya merusak sistem.

Menegakkan kontrol hak akses — System call melakukan pemeriksaan izin (permission check) sebelum mengizinkan proses membaca, menulis, atau mengeksekusi file dan sumber daya sistem.

Menjaga stabilitas dan isolasi proses — Dengan system call, kernel memastikan satu program tidak bisa mengganggu atau mengambil alih memori program lain.

OS memastikan transisi dari user mode ke kernel mode berjalan aman melalui beberapa mekanisme penting berikut:

Interrupt dan Trap Terproteksi — Transisi hanya bisa terjadi lewat interrupt atau system call instruction khusus (misalnya int 0x80 atau syscall), bukan sembarang instruksi dari program u\*ser.

Validasi Akses Memori dan Izin — Kernel memeriksa alamat memori dan hak akses proses sebelum menjalankan permintaan. Ini mencegah user mengakses area memori kernel atau proses lain.

Penggunaan Tabel System Call — Hanya fungsi yang terdaftar di tabel system call yang bisa dipanggil. Program user tidak bisa memanggil fungsi kernel lain secara langsung.

Isolasi Mode Eksekusi — CPU memiliki privilege level (user mode & kernel mode). Saat di kernel mode, OS bisa mengakses semua sumber daya; saat di user mode, akses sangat dibatasi.

Jadi, keamanan dijaga dengan pemisahan hak akses, validasi ketat, dan jalur transisi yang terkontrol.

open() – Membuka file atau perangkat untuk dibaca/ditulis.

read() – Membaca data dari file atau input.

write() – Menulis data ke file atau output.

close() – Menutup file yang sudah dibuka.

fork() – Membuat proses baru (child process).

exec() – Menjalankan program lain dalam proses yang sama.

wait() – Menunggu proses anak selesai.

exit() – Mengakhiri proses saat ini.

getpid() – Mengambil ID proses saat ini.

kill() – Mengirim sinyal ke proses lain (misalnya untuk menghentikan).

System call tersebut adalah jembatan utama antara program user dan kernel untuk melakukan operasi dasar sistem.