LAPORAN PRAKTIKUM TUGAS 2
SISTEM OPERASI 2024

Nama : **Muhammad Fadhil Zurani**

NIM : **122140146**

Kelas : **RC**

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI SUMATERA

2024

DAFTAR ISI

BAB 1
DASAR TEORI

1.1. Pengenalan System Call

System call ialah suatu metode program komputer dalam meminta layanan dari kernelsistem operasi di mana program tersebut dijalankan. Hal tersebut memungkinkanprogram untuk berinteraksi dengan sistem operasi, dengan program komputer melakukanpemanggilan sistem ketika memerlukan permintaan kepada kernel sistem operasi.Melalui antarmuka Program Aplikasi (API), System call memberikan layanan sistemoperasi kepada program pengguna. Dengan adanya antarmuka antara proses dan sistemoperasi, pemanggilan sistem memungkinkan proses pengguna untuk meminta layanandari sistem operasi. Satu-satunya titik masuk ke dalam sistem kernel adalah melaluipemanggilan sistem, sehingga semua program yang membutuhkan sumber daya harusmenggunakan metode ini.

Dasar-dasar teoritis tentang pengenalan system call meliputi beberapa konsep kunci:

1. **Interaksi dengan Kernel**: Kernel adalah bagian inti dari sistem operasi yang mengendalikan sumber daya perangkat keras dan memberikan layanan dasar kepada perangkat lunak yang berjalan di atasnya. System call menyediakan antarmuka yang terdefinisi dengan baik antara program pengguna (user-space) dan kernel-space. Program pengguna tidak diperbolehkan untuk secara langsung mengakses sumber daya perangkat keras atau menjalankan instruksi yang bersifat prihatin terhadap keamanan tanpa melewati kernel. Oleh karena itu, system call adalah jembatan yang memungkinkan program pengguna untuk berkomunikasi dengan kernel.

2. **Mekanisme Panggilan**: Setiap system call memiliki nomor identifikasi unik atau "syscall number" yang dipanggil oleh program pengguna untuk merujuk pada layanan tertentu yang disediakan oleh sistem operasi. Biasanya, program pengguna menggunakan fungsi wrapper atau instruksi khusus (misalnya, INT 0x80 di arsitektur x86) untuk memicu system call.

3. **Kategori System Call**: System call dapat dikelompokkan ke dalam beberapa kategori berdasarkan fungsi utamanya. Kategori-kategori ini mungkin termasuk manajemen proses (seperti pembuatan proses baru atau menunggu proses), manajemen file (seperti membuka, menulis, atau menutup file), alokasi memori, manajemen jaringan (seperti membuat atau menerima koneksi jaringan), dan banyak lagi.

4. **Keamanan**: Pengenalan system call juga mencakup aspek keamanan. Sistem operasi memastikan bahwa system call hanya dapat diakses oleh program yang memiliki izin yang sesuai. Hal ini dilakukan dengan menggunakan mekanisme hak akses dan mekanisme otentikasi untuk memeriksa identitas pengguna dan mengizinkan atau menolak akses ke system call tertentu.

5. **Pengelolaan Parameter**: System call biasanya menerima parameter dari program pengguna. Pengenalan system call mencakup definisi parameter yang diperlukan untuk setiap system call tertentu, serta konvensi pemanggilan yang harus diikuti oleh program pengguna dalam menyediakan parameter-parameter ini.

Pengenalan system call merupakan bagian kritis dari pemahaman tentang bagaimana program berinteraksi dengan sistem operasi di tingkat yang paling dasar. Dengan memahami dasar-dasar ini, pengembang perangkat lunak dapat menulis program yang efisien, aman, dan dapat diandalkan yang dapat beroperasi dengan baik di berbagai platform sistem operasi.

BAB 2
ULASAN SOAL

2.1 Membaca File dengan System Call

Pada folder “Data” gunakan perintah nano read.c untuk membuat file baru dengan ekstensi C, pada source code dari read.c terdapat O_RDONLY yang digunakan untuk membuka file dengan membacanya saja dan jalankan perintah gcc read.c -o readfile untuk membuat pemanggil sistem untuk membuka file, terakhir lakukan pemanggil sistem dengan perintah ./readfile

2.2 Implementasi System Call Exec

Pada folder “Data1” gunakan perintah nano programbaru.c untuk membuat file baru dengan ekstensi C yang dimana file ini akan dieksekusi nantinya menggunakan system call exec, dan selanjutnya jalankan perintah gcc programbaru.c -o ProgramTes untuk gcc untuk kode program programbaru.c, tahap berikutnya adalah membuat file exec.c dengan perintah nano exec.c dan pada kode program exec.c yang mengimplementasikan exec berjenis execvp() yang artinya mencari file yang ada di direktori yang sama tanpa memerlukan path lengkap dari program yang akan dijalankan. Setelah itu jalankan perintah gcc exec.c -o ExecTes dan ./ExecTes untuk menjalankan program exec.c

BAB 3
HASIL DAN JAWABAN

3.1 Membaca File dengan System Call

```
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~$ cd data
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data$ nano read.c
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data$ gcc read.c -o readfile
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data$ ls
close.c  closefile  delete.c  deletefile  identitas.txt  open.c  openfile  read.c  readfile  write.c  writefile
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data$ ./readfile
File Berhasil dibuka, isi file:
Muhammad Fadhil Zurani
122140146
RC
```

Pada folder “Data” gunakan perintah nano read.c untuk membuat file baru dengan ekstensi C, pada source code dari read.c terdapat O_RDONLY yang digunakan untuk membuka file dengan membacanya saja dan jalankan perintah gcc read.c -o readfile untuk membuat pemanggil sistem untuk membuka file, terakhir lakukan pemanggil sistem dengan perintah ./readfile

```
 GNU nano 6.2                                                    read.c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>

int main() {
int fd;
char buf[1024];
int coderead;

fd = open("identitas.txt", O_RDONLY);
if (fd == -1) {
printf("error ketika membuka file.\n");
return 1;
}

coderead = read(fd, buf, sizeof(buf));
if (coderead == -1) {
printf("Error ketika membaca file.\n");
return 1;
}

printf("File Berhasil dibuka, isi file:\n", buf);
close(fd);
return 0;
}
```

Gambar diatas adalah Kode Program read.c yang digunakan

3.2 Implementasi System Call Exec

```
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~$ cd data1
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ nano programbaru.c
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ gcc programbaru.c -o ProgramTes
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ ls
fork.c  ForkTes  programbaru.c  ProgramTes  wait.c  WaitTes
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ nano exec.c
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ gcc exec.c -o ExecTes
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ ls
exec.c  ExecTes  fork.c  ForkTes  programbaru.c  ProgramTes  wait.c  WaitTes
fadhil-122140146@fadhil122140146-ROG-Strix-G531GD-G531GD:~/data1$ ./ExecTes
Ini adalah program baru!
```

Pada folder “Data1” gunakan perintah nano programbaru.c untuk membuat file baru dengan ekstensi C yang dimana file ini akan dieksekusi nantinya menggunakan system call exec, dan selanjutnya jalankan perintah gcc programbaru.c -o ProgramTes untuk gcc untuk kode program programbaru.c, tahap berikutnya adalah membuat file exec.c dengan perintah nano exec.c dan pada kode program exec.c yang mengimplementasikan exec berjenis execvp() yang artinya mencari file yang ada di direktori yang sama tanpa memerlukan path lengkap dari program yang akan dijalankan. Setelah itu jalankan perintah gcc exec.c -o ExecTes dan ./ExecTes untuk menjalankan program exec.c

```
 GNU nano 6.2                                                programbaru.c *
#include <stdio.h>

int main() {
printf("Ini adalah program baru!\n");
return 0;
}
```

Gambar diatas adalah kode program untuk programbaru.c

```
 GNU nano 6.2                                                      exec.c *
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main() {
char *args[] = {"./ProgramTes", NULL};
int codeexec = execvp(args[0], args);

if (codeexec == -1) {
printf("Error ketika mengeksekusi file.\n");
return 1;
}

return 0;
}
```

Gambar diatas adalah kode program untuk exec.c

BAB 4
KESIMPULAN DAN SARAN

Berdasarkan latihan praktikum kali ini, didapatkan kesimpulan sebagai berikut :

1. System call membantu mahasiswa memahami bagaimana program berinteraksi dengan sistem operasi melalui system call untuk melakukan tugas-tugas tertentu..
2. Mengimplementasikan system call pada level kernel, termasuk bagaimana system call didefinisikan, dipanggil, dan dieksekusi oleh kernel.
3. Dapat menganalisa source code sistem operasi terkait system call, memahami struktur internal sistem operasi, dan mempelajari cara kerja system call secara mendalam.
4. Dapat memahami aspek keamanan dan kinerja dalam konteks system call, termasuk praktik terbaik untuk memastikan keamanan program dan optimalisasi kinerja system call.