**LAPORAN SISTEM OPERASI**

**Praktikum 7**

**“ Unix System Call dan Manajemen Memory”**

****

**FARID AZIZ WICAKSONO**

**1841720094**

**POLITEKNIK NEGERI MALANG**

**MEI 2019**

**Tugas Pendahuluan**

Jawablah pertanyaan-pertanyaan berikut ini:

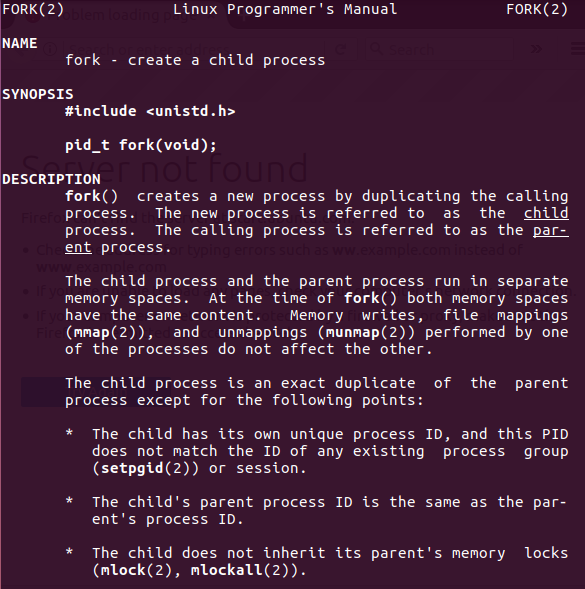
1. Apa yang dimaksud dengan system call?

Jawab: System call adalah tata cara pemanggilan di program aplikasi untuk memperoleh layanan yang disediakan oleh sistem operasi. System call berupa rutinitas sistem operasi untuk keperluan tertentu yang spesifik. System Call untuk manajemen proses diperlukan untuk mengatur proses-proses yang sedang berjalan.

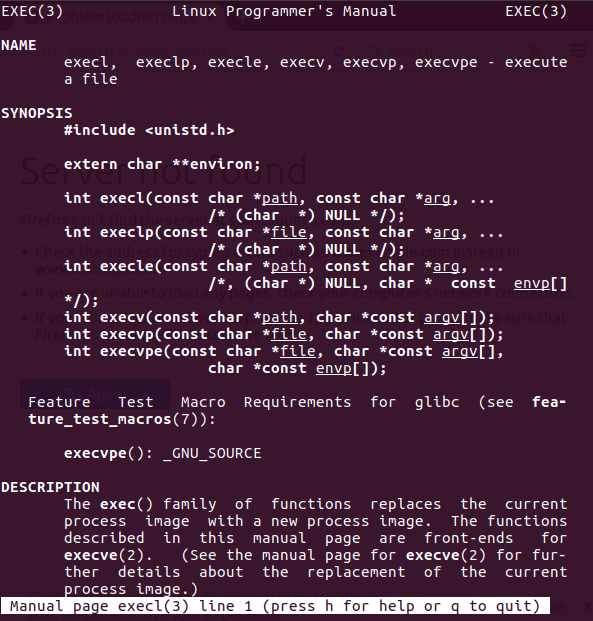
1. Apa yang dimaksud dengan system call fork(), execl() dan wait(). Jawablah dengan menggunakan perintah man (contoh : man 2 fork, man 2 execl dan man 2 wait)?

Jawab:

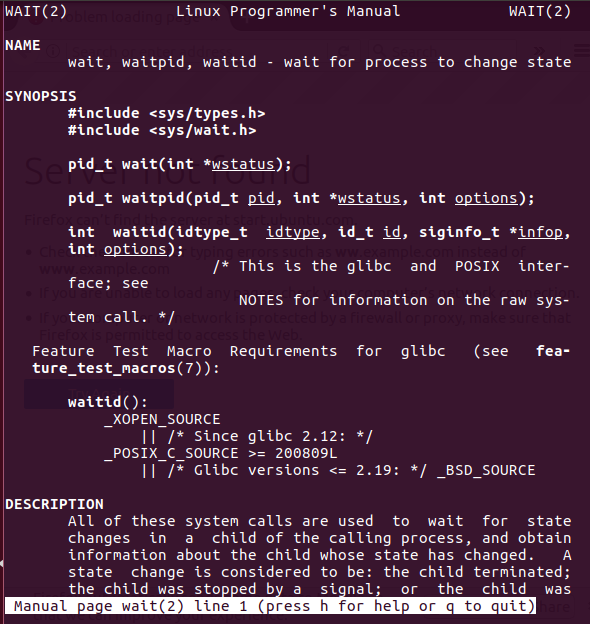
* System call fork() digunakan untuk membuat proses. Tidak perlu ada argumen dan mengembalikan ID proses. Tujuan fork () adalah untuk menciptakan proses baru, yang menjadi proses anak dari pemanggil. Setelah proses anak baru dibuat, kedua proses akan menjalankan instruksi berikutnya.



* System call execl() adalah fungsi dari sistem operasi yang menjalankan file yang dapat dieksekusi dalam konteks proses yang sudah ada, menggantikan eksekusi sebelumnya. Tindakan ini juga disebut sebagai overlay.



* System call wait() adalah fungsi dari sistem operasi yang digunakan untuk memblokir proses pemanggilan sampai salah satu proses turunannya keluar atau sinyal diterima.



1. Apa yang dimaksud system virtual memory, proses swapping dan buffer cache pada manajemen memory?

Jawab:

* Virtual Memory adalah teknik manajemen memori yang dikembangkan untuk kernel multitugas. Teknik ini divirtualisasikan dalam berbagai bentuk arsitektur komputer di dalam penyimpanan data (seperti RAM dan Harddisk), yang memungkinkan sebuah program harus dirancang seolah-olah hanya ada satu jenis memori yang digunakan yaitu memori “virtual”, yang bertindak secara langsung pada alamat memori baca/tulis (RAM)
* Swapping adalah Suatu metode pengalihan proses yang bersifat sementara dari memori utama ke suatu tempat penyimpanan sementara (disk) dan dipanggil kembali ke memori jika akan melakukan eksekusi. Adapun proses pindahkan yaitu proses yang di blocked ke disk dan hanya memasukkan proses-proses ready ke memory utama.
* Buffer Cache adalah buffer cache adalah proses yang terjadi pada daerah memeori yang meyimpan data ketika data sedang dipindahkan dari perangkat satu ke perangkat lainnya atau dari perangkat ke aplikasi yang memiliki akses data yang lebih cepat dan efisien.

1. Apa yang dimaksud perintah free dan cat/proc/meminfo?

Jawab:

* Free adalah perintah yang digunakan untuk melihat memori yang terpakai oleh suatu proses dan memperlihatkan swap space dan banyaknya swapping yang terjadi.
* Cat /proc/meminfo adalah perintah yang digunakan untuk melihat penggunaan memori dan swap-nya

1. Apa yang dimaksud perintah ps?

Jawab: Ps adalah perintah yang digunakan untuk melihat status proses yang terdiri dari attribut PID (Process Identifier), TTY, Time dan CMD

**Percobaan 1: Melihat proses parent dan proses child**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:    Analisa: listing program diatas berguna untuk menampilkan nomor PID dari proses dan parent serta user id dari user yang sedang aktif ketika program dieksekusi |
|  | Gunakan g++ compiler untuk menjalankan program diatas    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari proses, nomor PID dari parent dan user id dari user yang sedang aktif |

**Percobaan 2: Membuat dua proses terus menerus dengan sebuah system call fork()**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:    Analisa: listing program diatas berguna untuk menampilkan nomor PID dari proses beserta urutan foreground-nya di dalam memory ketika program dieksekusi |
|  | Gunakan g++ compiler untuk menjalankan program diatas. Pada saat dijalankan, program tidak akan pernah berhenti. Untuk menghentikan program tekan Ctrl+C.    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari proses beserta urutan foreground-nya di dalam memory yang dilakukan secara berulang-ulang hingga user menghentikannya sendiri dengan perintah ctrl+c |

**Percobaan 3: Membuat dua proses sebanyak lima kali**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:    Analisa: listing program diatas berguna untuk menampilkan nomor PID dari 2 buah proses yang dilakukan secara berulang sebayak 5 kali perulangan. |
|  | Gunakan g++ compiler untuk menjalankan program diatas    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari 2 buah proses yang dilakukan secara berulang sebayak 5 kali perulangan |

**Percobaan 4: Proses parent menunggu sinyal dari proses child dengan system call wait**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:    Analisa: listing program di atas berguna untuk menampilkan nomor PID dari parent dan child dimana nomor PID ini nantinya akan dipanggil kembali pada pendeklarasian string berikutnya dan perulangan terjadi pada program ini sebanyak 2 perulangan dengan menampilkan nomor PID parent, nomor PID child, String parent dan String Child |
|  | Gunakan g++ compiler untuk menjalankan program diatas    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari parent dan child dimana nomor PID ini nantinya akan dipanggil kembali pada pendeklarasian string berikutnya dan perulangan terjadi pada program ini sebanyak 2 perulangan dengan menampilkan nomor PID parent, nomor PID child, String parent dan String Child. |

**Percobaan 5: System call fork/exec dan wait mengeksekusi program bernama ls, menggunakan file executable /bin/ls dengan satu parameter –l yang ekuivalen dengan ls –l**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:      Analisa: listing program di atas berguna untuk menampilkan nomor PID dari parent dan child dimana PID tersebut nantinya akan dipanggil kembali pada pendeklarasian string berikutnya kemudian program menampilkan informasi tentang user yang sedang aktif dan mengakhiri eksekusi dengan tampilan “I am a parent and I am quitting.” |
|  | Gunakan g++ compiler untuk menjalankan program diatas    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari parent dan child dimana PID tersebut nantinya akan dipanggil kembali pada pendeklarasian string berikutnya kemudian program menampilkan informasi tentang user yang sedang aktif dan mengakhiri eksekusi dengan tampilan “I am a parent and I am quitting.” |

**Percobaan 6: System call fork/exec dan wait mengeksekusi program lain**

|  |  |
| --- | --- |
| No | Keterangan |
|  | Dengan menggunakan editor vi, buatlah file cpp dan ketikkan program berikut:      Analisa: listing program di atas berguna untuk menampilkan nomor PID dari parent dan child dimana nomor PID tersebut nantinya akan dipanggil kembali pada pendeklarasian string berikutnya dan program juga menampilkan nomor PID dari 2 buah proses dan mengulangnya sebanyak 5 kali pengulangan |
|  | Gunakan g++ compiler untuk menjalankan program diatas    Analisa: perintah di atas digunakan untuk mengcompile file yang telah dibuat sebelumnya ke dalam bentuk file shell script sehingga dapat dieksekusi |
|  | Amati output yang dihasilkan    Analisa: Hasil output menampilkan nomor PID dari parent dan child dimana nomor PID tersebut nantinya akan dipanggil kembali pada pendeklarasian string berikutnya dan program juga menampilkan nomor PID dari 2 buah proses dan mengulangnya sebanyak 5 kali pengulangan |

**Percobaan 7: Melihat Manajemen Memory**

|  |  |
| --- | --- |
| No | Soal |
|  | Perhatikan dengan perintah dmesg jumlah memory tersedia dan proses swapping    Analisa: Pada gambar diatas terlihat bahwa jumah memory yang tersedia adalah 154840 Kb dan jumlah memory swapping yang tersedia adalah 444072 Kb |
|  | Dengan perintah free perhatikan jumlah memory ”free”, ”used”, “share” dan “buffer” .    Analisa: Pada gambar di atas terlihat bahwa perintah free menyediakan layanan berupa menampilkan informasi mengenai penggunaan aktivitas memory yang terdiri dari total, used, free, shared, buffer/cache dan available |
|  | Dengan perintah dibawah ini apakah hasilnya sama dengan no 2?    Analisa: Ya, hasilnya sama, namun pada perintah cat /proc/meminfo lebih banyak menampilkan informasi secara detail dan rinci mengenai aktivitas penggunaan memory |
|  | Gunakan perintah dibawah ini    Analisa: perintah diatas diguanakan untuk menampilkan isi dari direktori root secara rinci |
|  | Perhatikan perubahan manajemen memory    Analisa: Perintah diatas digunakan untuk menampilkan aktivitas dari penggunaan memory dan swap secara general |
|  | Jalankan sebuah program, misalnya open Office. Perhatikan perubahan manajemen memory    Analisa: Pada saat kita menjalankan libre office maka memory akan menggunakan bagian pada dirinya yang masih free dan hal ini akan mengurangi jumlah total memory bebas yang tersedia, dimana sebelumnya (sebelum menjalankan libre office) memory yang tersedia sebanyak 145708 Kb dan setelah menjalankan libre office memory free berkurang menjadi 70424 Kb |
|  | Dengan perintah ps bagaimana penggunaan memory untuk setiap proses diatas?    Analisa: Dengan dijalankannya aplikasi libre office, tentunya penggunaan memory menjadi semakin besar dan menambah daftar proses yang terdaftar pada ps –uax. Melalui perintah ini kita bisa melihat berbagai proses yang sedang berjalan di memory, baik itu proses background maupun foreground secara detail. Pada gambar terlihat bahwa aplikasi libre office menggunakan sumber daya memory sebanyak 14.9% dan CPU 4.1% |

**Latihan:**

|  |  |
| --- | --- |
| No | Soal |
|  | Ubahlah program fork5.cpp pada percobaan 5 untuk mengeksekusi perintah yang ekuivalen dengan   1. ls –al /etc.      1. cat fork2      1. ./fork2     Analisa: Pada program diatas setelah listing program di buat maka program terlebih dahulu harus di compile ke dalam betuk shell script kemudian di eksekusi dan program menampilkan hasil output berupa semua isi (beserta file hidden) dari direktori /etc setelah itu program membuka fork2 dengan perintah cat dan kemudian program diakhiri dengan menampilkan 2 buah proses beserta PID nya yang dilakukan secara berulang-ulang |
|  | Informasi apa saja mengenai manajemen memory yang ditampilkan pada perintah dmesg pada percobaan Anda?    Analisa: Perintah diatas digunakan untuk menampilkan informasi berupa hardware yang sedang beraktivitas beserta file yang sedang dieksekusinya |
|  | Bagaimana informasi yang ditampilkan dengan perintah free pada percobaan Anda?    Analisa: Informasi yang ditampilkan pada percobaa saya adalah informasi berupa total memory dan swap, penggunaan memory dan swap, total memory dan swap yang sedang free, memory yang bisa dibagi, besaran buffering cache dan available memory |
|  | Apa isi file /proc/meminfo pada percobaan yang Anda lakukan?      Analisa: Isi file /proc/meminfo adalah informasi berupa aktivitas penggunaan memory secara detail yang diantaranya berupa total memory, penggunaan memory, memory yang sedang free, besaran buffering dan cache dan available memory |
|  | Berapa besar memory yang digunakan setelah percobaan 7 dengan perintah ps – uax?    Analisa: besarnya memory yang digunakan setelah percobaan 7 adalah 303648 Kb dengan penggunaan CPU sebesar 20.2% |
|  | Lakukan hal yang sama dengan percobaan 7 untuk melihat perubahan memory setelah dilakukan beberapa proses pada shell. Tentukan perintah yang dilakukan misalnya membuka browser dan perhatikan hal-hal berikut:   1. Informasi apa saja yang ditampilkan dengan perintah free?     Analisa: Informasi yang ditampilkan pada percobaa saya adalah informasi berupa total memory dan swap, penggunaan memory dan swap, total memory dan swap yang sedang free, memory yang bisa dibagi, besaran buffering cache dan available memory   1. Informasi apa saja yang disimpan file /proc/meminfo?     Analisa: Isi file /proc/meminfo adalah informasi berupa aktivitas penggunaan memory secara detail yang diantaranya berupa total memory, penggunaan memory, memory yang sedang free, besaran buffering dan cache dan available memory   1. Berapa besar kapasitas memory total?      1. Berapa kapasitas memory yang sudah terpakai?      1. Berapa kapasitas memory yang belum terpakai?      1. Berapa kapasitas memory yang digunakan sharing beberapa proses?      1. Berapa kapasitas buffer cache? |

**Kesimpulan**

Kesimpulan yang saya dapat setelah mengikuti praktikum ini dengan baik adalah:

* System call adalah tata cara pemanggilan di program aplikasi untuk memperoleh layanan yang disediakan oleh sistem operasi. System call berupa rutinitas sistem operasi untuk keperluan tertentu yang spesifik. System Call untuk manajemen proses diperlukan untuk mengatur proses-proses yang sedang berjalan.
* Manajemen memory adalah proses mengelola memori computer dengan efisien. Kebutuhan utama manajemen memory adalah untuk menyediakan cara secara dinamis dalam mengalokasikan bagian-bagian dari memori untuk program atas request yang dilakukan dan membebaskan memori untuk digunakan kembali ketika tidak lagi diperlukan.