Weekly Log 4 System Programming

Nama: Mahardika Krisna Ihsani

Kelas: A

NPM: 1806141284

Hal yang Sejauh Ini Masih bisa Dipahami

Pada week ini, Saya mempelajari mengenai process serta keterkaitannya dengan program dan thread. Jadi secara sederhananya suatu program terdiri atas beberapa process yang mengsekusi beberpa instruksi, sedangkan suatu process dapat mempunyai lebih dari satu thread. Process ini sendiri mempunyai ID unik yang di-assign oleh operating system yang dikenal dengan PID. Suatu process juga mempunyai hierarki yaitu adanya parent-children relation. Hubungan ini menunjukkan bahwa child process berjalan di atas parent process, dan child process mempunyai PID yang berbeda dengan PID parent-nya. Untuk memperoleh PID dari process bisa dilakukan dengan memasukkan command get_pid() dan untuk memperoleh PID parent process dari process yang ditunjuk dapat dilakukan dengan mengeksekusi command get_ppid(). PID dari UNIX akan melakukan reset ke 300 karena untuk PID di bawah 300 di-reserve untuk kernel process secara default. Untuk membuat child process yang merupakan duplikat dari process, bisa dilakukan command fork(). Command ini menerapkan copy-on-write. Ini artinya resource yang diakses oleh parent process dan child process akan bersifat shared resources selama salah satunya belum melakukan modifikasi pada resource tersebut. Bila hal tersebut terjadi, maka akan dibuatkan salinan resource yang ditujukan untuk process yang memodifikasi resource tersebut. Untuk mengecek status process, bisa dilakukan dengan cat /proc/<pid>/status. Untuk mengakses informasi informasi yang berkaitan dengan process, kita bisa menggunakan command ps. Contoh dari penggunaan command ini adalah ps -e. Command ini akan menampilkan processes yang berjalan pada saat ini. Untuk mengambil process spesifik, kita bisa mengeksekusi ps -e | grep <keyword-dari-process> (bila kita tidak tahu PID-nya) atau ps –pid <PID number>. Process ini mempunyai status yang terdiri atas created(process baru dibuat), running/runnable(process sedang dieksekusi atau siap untuk dieksekusi), sleeping(process menunggu resource yang diperlukan tersedia atau menunggu suatu event), stopped(process berhenti mengeksekusi instruksi), dan zombie(process sudah menyelesaikan instruksi namun masih tersimpan di process table). Dalam process cycle, terdapat 3 istilah yakni daemon process, orphaned process, dan zombie process. Daemon process adalah process yang berjalan di background dan mempunyai adoptive parent process yakni init (process dengan PID 1) dan bertanggung jawab untuk mematikan parent processes. Orphaned process adalah process yang parent process-nya sudah terminated sebelum process itu sendiri terminated. Zombie process adalah process yang sudah berhasil menyelesaikan instruksi namun tidak terhapus pada process table karena parent process masih membutuhkan exit status dari process tersebut.

Pada memory, linux menggunakan virtual memory management system. Ini artinya process akan me-refer ke virtual memory daripada langsung me-refer ke address di physical memory. Dalam linux maupun windows, terdapat Namanya environment variable. Environment variable dapat diartikan sebagai suatu variable yang dapat dipakai oleh beberapa process yang berjalan. Variable ini tersimpan dalam operating system daripada dalam suatu program sehingga bisa diakses oleh berbagai process. Contoh environment di sini adalah PATH yang me-refer ke beberapa path binary di linux sehingga kita bisa mengeksekusi langsung suatu alias command di terminal tanpa harus menuliskan path lengkapnya. Untuk menambahkan environment variables secara permanen, kita bisa menambahkan line “export <nama-variabel>=<suatu-value>” ke file “~/.profile”. Setelah di-save, maka bisa lakukan relogin atau source “~/.profile” untuk menyimpan export env variable tersebut secara permanen. Terdapat command untuk mengambil environment variable, yaitu char *getenv(const char *name). Untuk menambahkan/mengubah nilai environment bisa dilakukan dengan setenv atau putenv. Di linux suatu file atau folder terdapat group dan user untuk mengatur kepemilikan serta permission dalam pengaksesannya.

Hal yang Belum Dipahami

Saya masih kurang memahami bagaimana sbrk(), malloc(), brk() sebenarnya bekerja.