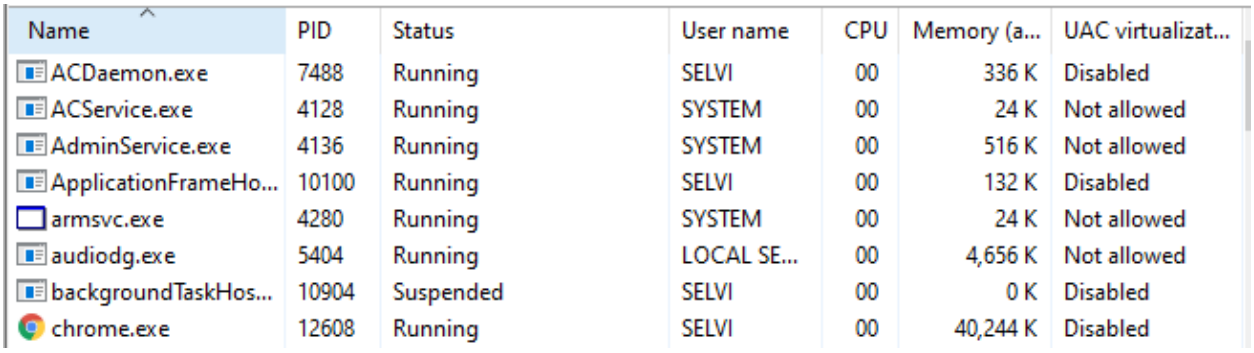
Log Week 4 Sysprog: Process & Memory

A. Proses

Proses adalah bagian dari entitas suatu program sementara **memori** adalah bagian yang dipakai oleh proses untuk melaksanakan tugasnya.

- Ketika kita menjalankan sebuah proses maka akan Kernel harus menyediakan *resource* untuk proses tersebut.
- Contoh beberapa proses yang sedang berjalan di computer saya (untuk pengguna Windows, bisa dilihat di Task Manager > Details



Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
ACDaemon.exe	7488	Running	SELVI	00	336 K	Disabled
ACService.exe	4128	Running	SYSTEM	00	24 K	Not allowed
AdminService.exe	4136	Running	SYSTEM	00	516 K	Not allowed
ApplicationFrameHo...	10100	Running	SELVI	00	132 K	Disabled
armsvc.exe	4280	Running	SYSTEM	00	24 K	Not allowed
audiodg.exe	5404	Running	LOCAL SE...	00	4,656 K	Not allowed
backgroundTaskHos...	10904	Suspended	SELVI	00	0 K	Disabled
chrome.exe	12608	Running	SELVI	00	40,244 K	Disabled

Masing-masing proses tersebut memiliki Process ID (PID) yang berguna sebagai *identifier* untuk sebuah proses. Selain itu, sebuah proses harusnya memiliki PPID (Parent Process ID), namun pada Windows saya belum menemukan letaknya dibagian mana.

Sementara pada Linux, PID dan PPID semuanya ditampilkan. Kolom kedua dari kiri adalah PID, sementara kolom ketiga dari kiri adalah PPID. PPID adalah ID dari sebuah proses yang sudah ada dan menjadi *parent* dari proses yang sekarang dilakukan.

```
user@sysprog-ova:~$ ps -ef | grep selvy.txt
user      1575  1329  0 03:03 pts/0    00:00:00 vi selvy.txt
user      1657  1644  0 03:04 pts/1    00:00:00 grep --color=auto selvy.txt
```

Untuk melihat informasi tentang PID dan PPID, ketikkan perintah ini:

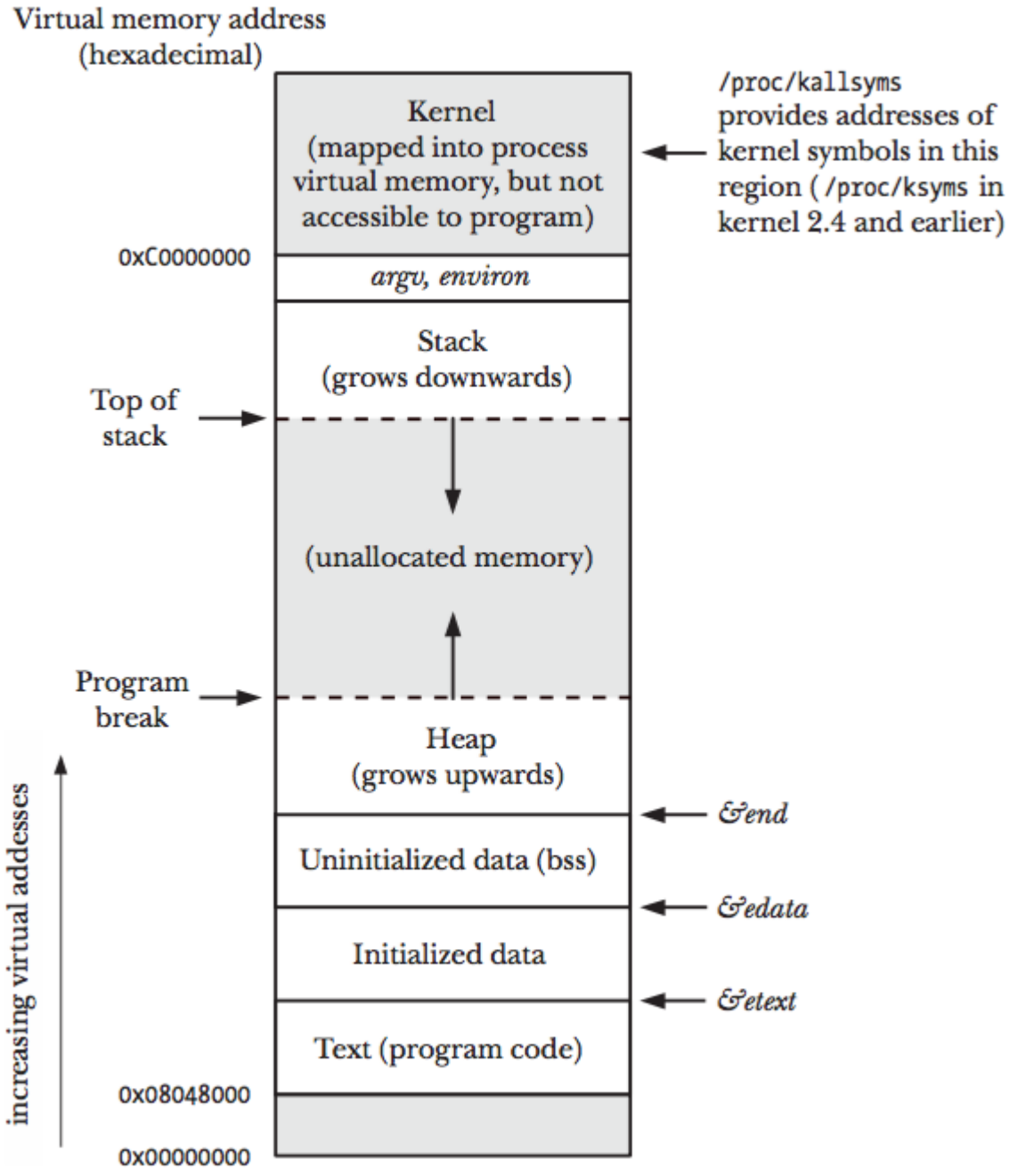
```
user@sysprog-ova:/proc/1329$ ps -ef | grep 1329
user      1329  1328  0 02:20 pts/0    00:00:00 -bash
```

Lalu, bagaimana kalau *parent* dari proses kita kirimkan sinyal **-9 (dead)** menggunakan *kill?*

```
user@sysprog-ova:/$ sudo kill -9 1329
```

Ternyata, proses dengan PPID tersebut akan otomatis *diselesaikan*.

B. Memory Layout of Memory



Sumber gambar: https://notes.shichao.io/tlpi/figure_6-1.png

C. Materi lainnya

- Dari pengerjaan *worksheet*, saya me-recall kembali konsep fork() yang telah saya pelajari di OS. Fork() adalah system call yang membuat duplikat dari X menjadi proses Y, X dinamakan *parent process*, sementara Y dinamakan *child process*. Kedua

proses ini menggunakan konsep Copy-on-write apabila ingin mengakses file yang sama dalam waktu bersamaan.