LABORATORIUM JARINGAN KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

PRAKTIKUM SISTEM OPERASI

SEMESTER : GENAP **TAHUN : 2015/2016**

BAB II

JUDUL BAB : PROSES

DISUSUN OLEH : MOH ARIF ANDRIAN

NIM : 156150600111002

ASISTEN : SISKA PERMATASARI
ZAENAL KURNIAWAN

KOORDINATOR ASISTEN : DANY RAHMANA

LAPORAN PRAKTIKUM SISTEM OPERASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

Nama : Moh. Arif Andrian
NIM : 156150600111002
Laporan : BAB II
Asisten : Siska Permatasari
Zaenal Kurniawan

BAB II
PROSES

I. Dasar Teori

Proses pada Linux merupakan aktifitas permintaan user terhadap sistem operasi. Model proses pada Linux mirip dengan UNIX, dimana prinsip dasar keduanya menggunakan fungsi fork() dan exec(). fork() digunakan untuk membuat proses baru sedangkan exec() digunakan untuk memanggil program.

Kedua pendekatan diatas merupakan dua hal yang berbeda, dimana proses (child) bisa diciptakan tanpa membuka program baru, dan secara sederhana akan meneruskan program awal (parent) untuk mengeksekusi perintah yang sama pada program awal.

Untuk membuat proses baru, bisa dengan mengetikan perintah langsung pada shell Linux.

TIPE PROSES:

Terdapat beberapa tipe proses yang dikenal dalam OS berbasis Linux pada umumnya, antara lain:

- Interactive
  Proses yang dimulai (dan dikontrol oleh) shell . Bisa tampak di luar (foreground) ataupun hanya di dalam (background).
- Batch
  Proses yang tidak berhubungan dengan terminal, tetapi menunggu untuk dieksekusi secara berurutan (sekuensial).
- Daemon
  Proses yang dimulai ketika Linux booting dan berjalan secara background. Proses ini menunggu permintaan dari proses lainnya, bila tidak ada request, maka berada dalam keadaan ‘idle’.

 # Tipe-tipe proses dalam Linux, dibagi ke dalam 3 bagian. Sebutkan dan jelaskan!

1. Interactive : proses yang dilakukan oleh pemakai langsung pada terminal. Bisa tampak di luar atau hanya di dalam.
2. Batch : Proses yang tidak berhubungan dengan terminal. Proses ini dijalankan secara sekuensial (satu persatu).
3. Daemon : Proses yang menunggu permintaan dari proses lainnya dan

menjalankan tugas sesuai dengan permintaan tersebut. Bila tidak ada proses, kondisinya menjadi “idle”.

Process Environment

Pada sistem operasi Linux process environment terdiri dari dua komponen, argumen dan environment. **Argumen** adalah daftar opsi tambahan pada CLI yang berkaitan dengan perilaku program ketika dijalankan, sedangkan **environmen**t adalah daftar parameter, baik berupa variabel, direktori home yang secara tekstual dibutuhkan oleh program.

Environment variable biasanya terdiri dari beberapa informasi seperti:

- PATH, daftar lokasi direktori dimana file executable berada.
- HOME, lokasi direktori home.
- CPPLIBS, lokasi dimana library yang berkaitan dengan program disimpan.
- HOSTNAME digunakan untuk penamaan mesin.
- USER user yang digunakan pada saat login pada sistem operasi.

 # Untuk itu kita harus mempelajari Manajemen Proses pada Linux. Perintah INTI dari proses manajemen proses di Linux ada 2. Sebutkan!

Pertama:

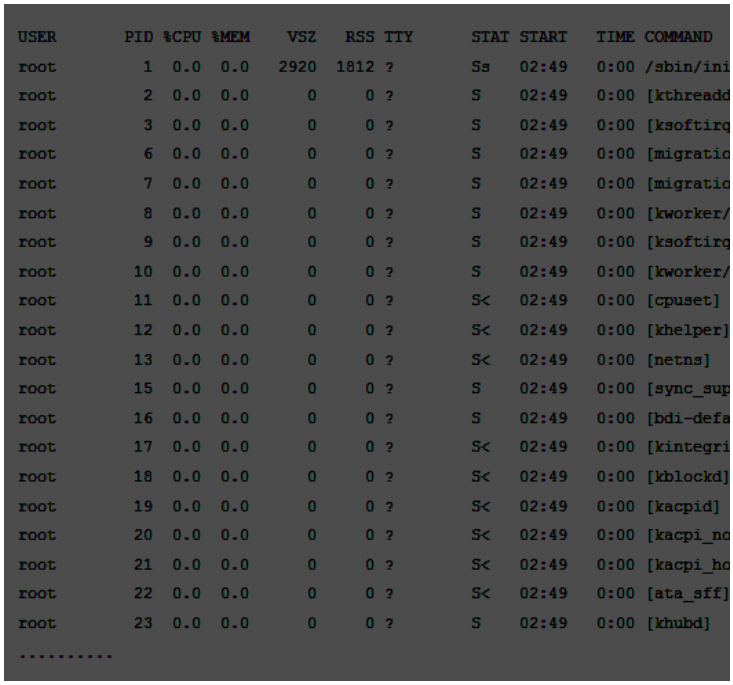
```
$ ps
```

PID adalah nomor identitas proses. TTY adalah nama terminal di mana proses tersebut sedang berlangsung.

Kedua:

```
$ kill
```

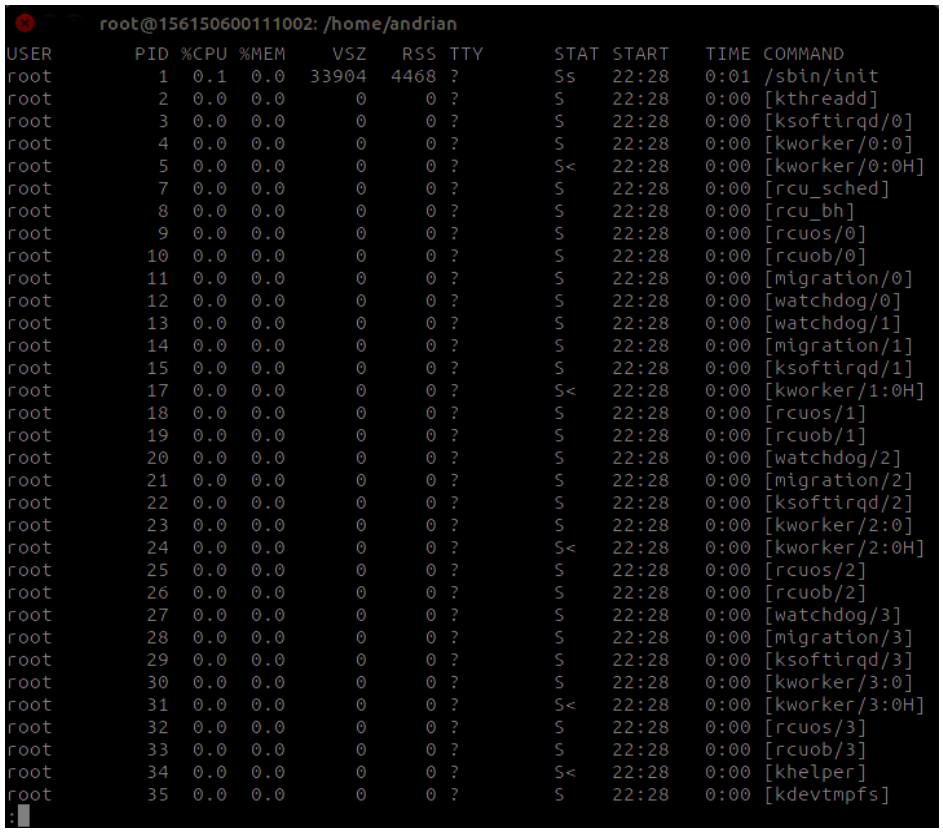
1. **Memulai menggunakan perintah ps**



```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   2920  1812 ?        Ss   02:49   0:00 /sbin/ini
root         2  0.0  0.0      0     0 ?        S    02:49   0:00 [kthreadd
root         3  0.0  0.0      0     0 ?        S    02:49   0:00 [ksoftirq
root         6  0.0  0.0      0     0 ?        S    02:49   0:00 [migratio
root         7  0.0  0.0      0     0 ?        S    02:49   0:00 [migratio
root         8  0.0  0.0      0     0 ?        S    02:49   0:00 [kworker/
root         9  0.0  0.0      0     0 ?        S    02:49   0:00 [ksoftirq
root        10  0.0  0.0      0     0 ?        S    02:49   0:00 [kworker/
root        11  0.0  0.0      0     0 ?        S<   02:49   0:00 [cpuset]
root        12  0.0  0.0      0     0 ?        S<   02:49   0:00 [khelper]
root        13  0.0  0.0      0     0 ?        S<   02:49   0:00 [netns]
root        15  0.0  0.0      0     0 ?        S    02:49   0:00 [sync_sup
root        16  0.0  0.0      0     0 ?        S    02:49   0:00 [bdi-defa
root        17  0.0  0.0      0     0 ?        S<   02:49   0:00 [kintegri
root        18  0.0  0.0      0     0 ?        S<   02:49   0:00 [kblockd]
root        19  0.0  0.0      0     0 ?        S<   02:49   0:00 [kacpid]
root        20  0.0  0.0      0     0 ?        S<   02:49   0:00 [kacpi_no
root        21  0.0  0.0      0     0 ?        S<   02:49   0:00 [kacpi_ho
root        22  0.0  0.0      0     0 ?        S<   02:49   0:00 [ata_sff]
root        23  0.0  0.0      0     0 ?        S    02:49   0:00 [khubd]
.........
```

 # Apa perintah yang seharusnya Anda tulis agar menampilkan hasil output seperti di atas?

```
$ ps -aux | less
```



```
root@1561506001110O2: /home/andrian
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.1  0.0  33904  4468 ?        Ss   22:28   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S    22:28   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    22:28   0:00 [ksoftirqd/0]
root         4  0.0  0.0      0     0 ?        S    22:28   0:00 [kworker/0:0]
root         5  0.0  0.0      0     0 ?        S<   22:28   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    22:28   0:00 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    22:28   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuos/0]
root        10  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuob/0]
root        11  0.0  0.0      0     0 ?        S    22:28   0:00 [migration/0]
root        12  0.0  0.0      0     0 ?        S    22:28   0:00 [watchdog/0]
root        13  0.0  0.0      0     0 ?        S    22:28   0:00 [watchdog/1]
root        14  0.0  0.0      0     0 ?        S    22:28   0:00 [migration/1]
root        15  0.0  0.0      0     0 ?        S    22:28   0:00 [ksoftirqd/1]
root        17  0.0  0.0      0     0 ?        S<   22:28   0:00 [kworker/1:0H]
root        18  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuos/1]
root        19  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuob/1]
root        20  0.0  0.0      0     0 ?        S    22:28   0:00 [watchdog/2]
root        21  0.0  0.0      0     0 ?        S    22:28   0:00 [migration/2]
root        22  0.0  0.0      0     0 ?        S    22:28   0:00 [ksoftirqd/2]
root        23  0.0  0.0      0     0 ?        S    22:28   0:00 [kworker/2:0]
root        24  0.0  0.0      0     0 ?        S<   22:28   0:00 [kworker/2:0H]
root        25  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuos/2]
root        26  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuob/2]
root        27  0.0  0.0      0     0 ?        S    22:28   0:00 [watchdog/3]
root        28  0.0  0.0      0     0 ?        S    22:28   0:00 [migration/3]
root        29  0.0  0.0      0     0 ?        S    22:28   0:00 [ksoftirqd/3]
root        30  0.0  0.0      0     0 ?        S    22:28   0:00 [kworker/3:0]
root        31  0.0  0.0      0     0 ?        S<   22:28   0:00 [kworker/3:0H]
root        32  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuos/3]
root        33  0.0  0.0      0     0 ?        S    22:28   0:00 [rcuob/3]
root        34  0.0  0.0      0     0 ?        S<   22:28   0:00 [khelper]
root        35  0.0  0.0      0     0 ?        S    22:28   0:00 [kdevtmpfs]
:
```

 # Apa fungsi perintah tersebut?

Untuk menampilkan semua proses yang sedang berjalan. %CPU adalah presentasi CPU time yang digunakan oleh proses tersebut, %MEM adalah presentasi system memori yang digunakan proses, SIZE adalah jumlah memori yang digunakan, RSS (Real System Storage) adalah jumlah memori yang digunakan, START adalah kapan proses tersebut diaktifkan.

2. Menampilkan semua proses yang sedang berjalan pada sistem

```
  PID TTY          TIME CMD
    1 ?        00:00:00 init
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 ksoftirqd/0
    6 ?        00:00:00 migration/0
    7 ?        00:00:00 migration/1
    9 ?        00:00:00 ksoftirqd/1
   10 ?        00:00:00 kworker/0:1
   11 ?        00:00:00 cpuset
   12 ?        00:00:00 khelper
   13 ?        00:00:00 netns
   15 ?        00:00:00 sync_supers
   16 ?        00:00:00 bdi-default
   17 ?        00:00:00 kintegrityd
   18 ?        00:00:00 kblockd
   19 ?        00:00:00 kacpid
   20 ?        00:00:00 kacpi_notify
   21 ?        00:00:00 kacpi_hotplug
   22 ?        00:00:00 ata_sff
   23 ?        00:00:00 khubd
```

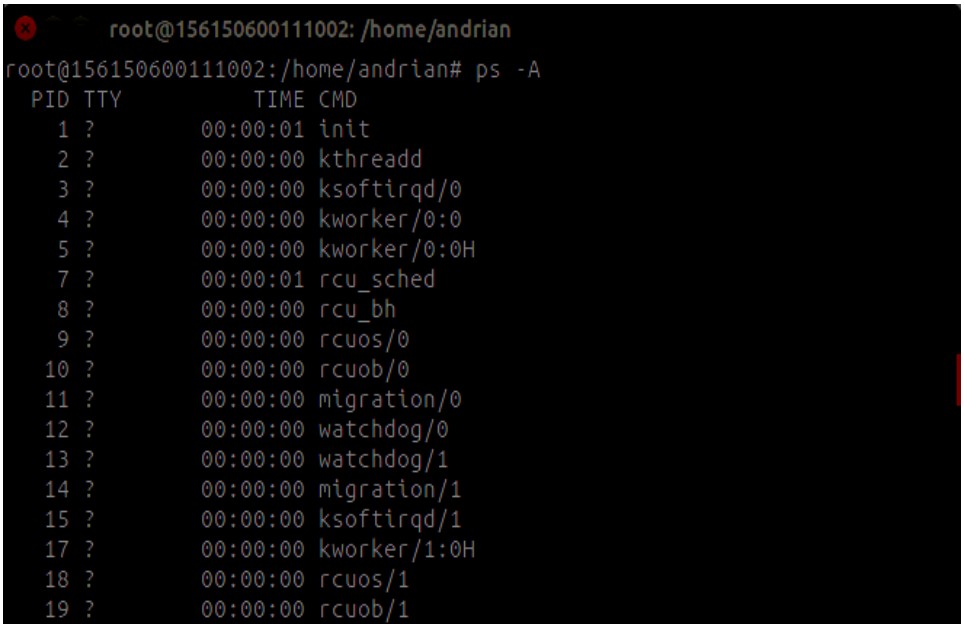
 # Apa perintah yang seharusnya Anda tulis agar menampilkan hasil output seperti di atas?

```
$ ps -e
```

```
root@156150600111002: /home/andrian
root@156150600111002:/home/andrian# ps -e
  PID TTY          TIME CMD
    1 ?        00:00:01 init
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 ksoftirqd/0
    4 ?        00:00:00 kworker/0:0
    5 ?        00:00:00 kworker/0:0H
    7 ?        00:00:01 rcu_sched
    8 ?        00:00:00 rcu_bh
    9 ?        00:00:00 rcuos/0
   10 ?        00:00:00 rcuob/0
   11 ?        00:00:00 migration/0
   12 ?        00:00:00 watchdog/0
   13 ?        00:00:00 watchdog/1
   14 ?        00:00:00 migration/1
   15 ?        00:00:00 ksoftirqd/1
   17 ?        00:00:00 kworker/1:0H
   18 ?        00:00:00 rcuos/1
   19 ?        00:00:00 rcuob/1
```

 # atau Anda bisa menggunakan perintah lain, seperti?

```
$ ps-A
```



 # option -a **akan menampilkan semua user yang sedang menjalankan proses, option** -u **berfungsi untuk menampilkan semua proses yang lain yang sedang berjalan, option** -x **berfungsi untuk menampilkan proses yang tidak dikontrol oleh terminal (tty) seperti daemon yang dijalankan saat booting.**

 # Dan apa perbedaan perintah di atas dengan perintah ps -e |more?

ps –e : untuk menampilkan semua proses dalam bentuk 4 kolom : PID, TTY, TIME, dan CMD.

➔ PID : proses ID untuk menghentikan atau mengunci suatu proses dari aplikasi.

➔ TTY : menampilkan nomor terminal tempat user bekerja.

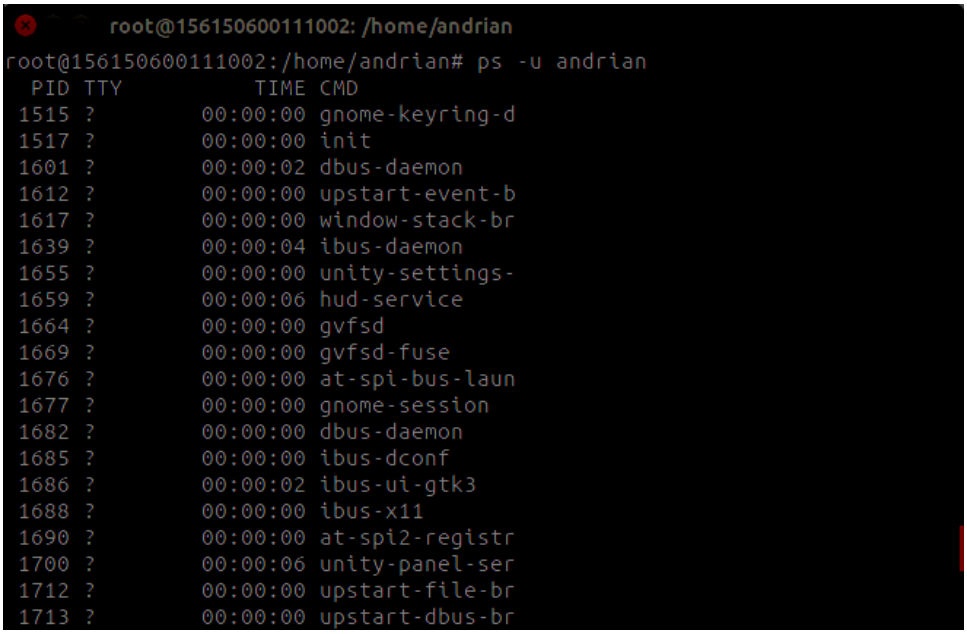
➔ TIME : lama waktu proses yang dijalankan.

➔ CMD : perintah yang dijalankan/sedang proses.

ps –e | more : untuk menampilkan semua proses dalam format penuh.

 # Dan perintah apa yang berfungsi menampilkan semua proses dalam format sesuai definisi user yaitu terdiri dari kolom PID dan CMD?

```
$ ps -<username>
```

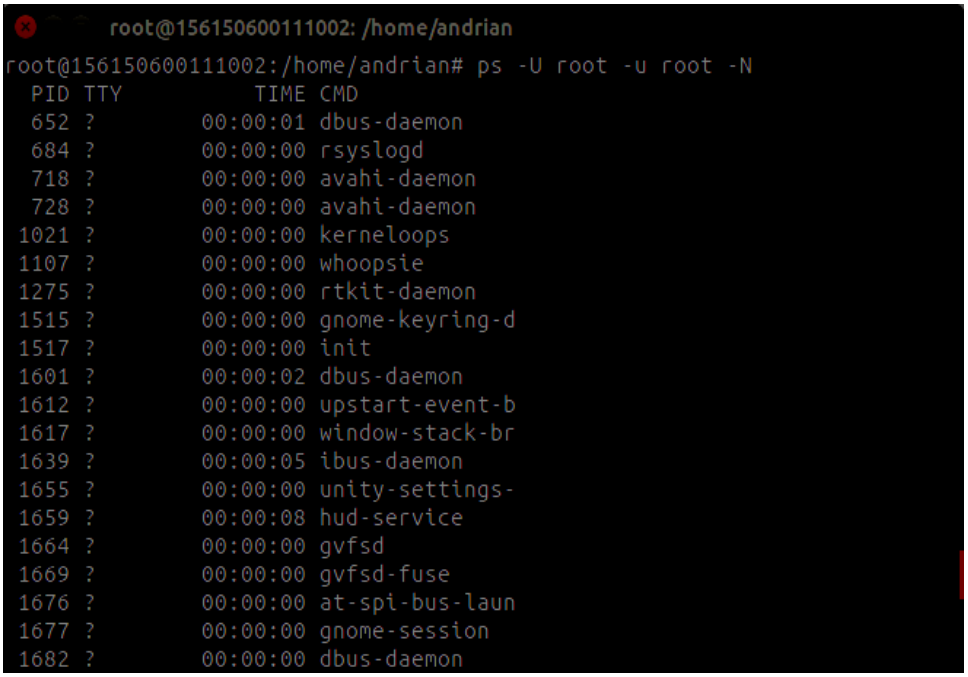


3. **Melihat semua proses yang berjalan kecuali root**

 # Pada poin ketiga ini, Anda dapat menggunakan perintah seperti di bawah ini:

```
ps -U root -u root -N
```

#Bagaimana hasil ouputnya?

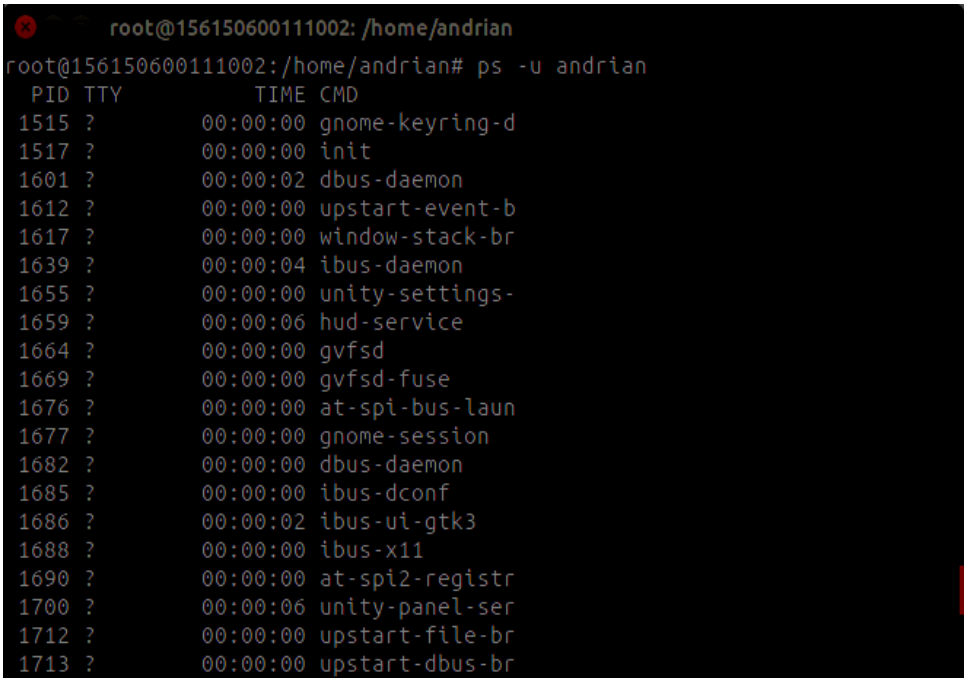


Proses yang dijalankan root tidak akan ditampilkan.

4. **Menampilkan proses yang sedang dijalankan oleh user tertentu**
   # Untuk menjalankan fungsi pada poin empat, Anda dapat menggunakan perintah seperti di bawah ini :

```
ps -u <user>
```

#Bagaimana hasil outputnya?



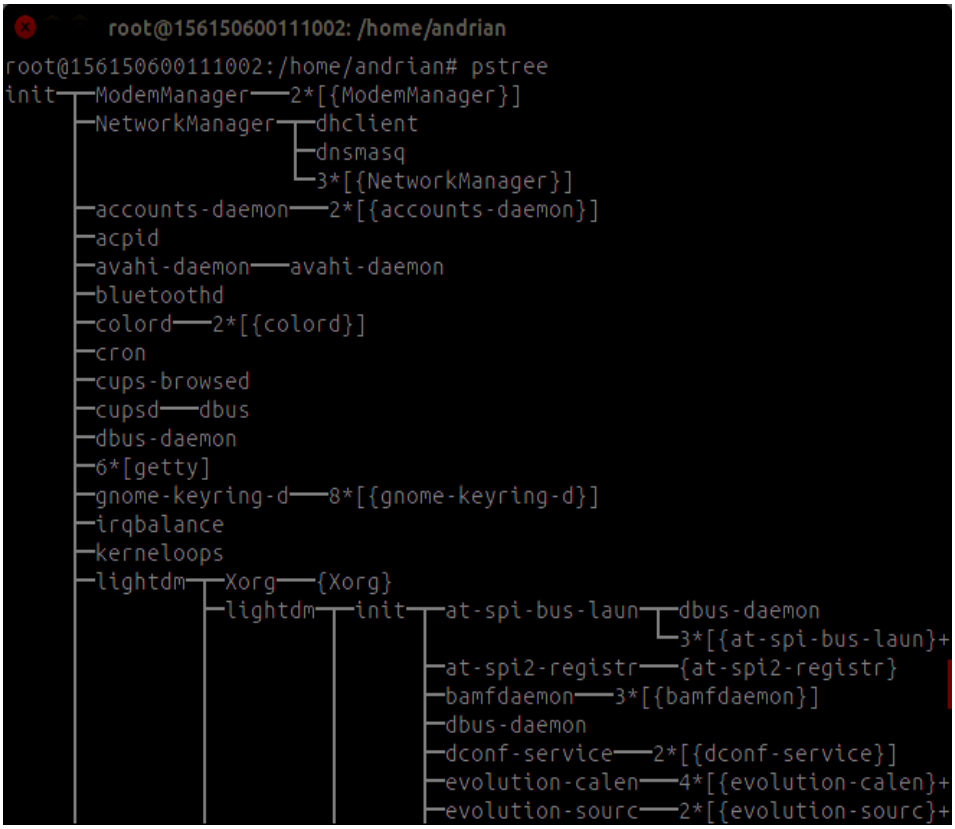
 # Bagaimana contoh sintaksnya?

```
$ ps –u andrian
```

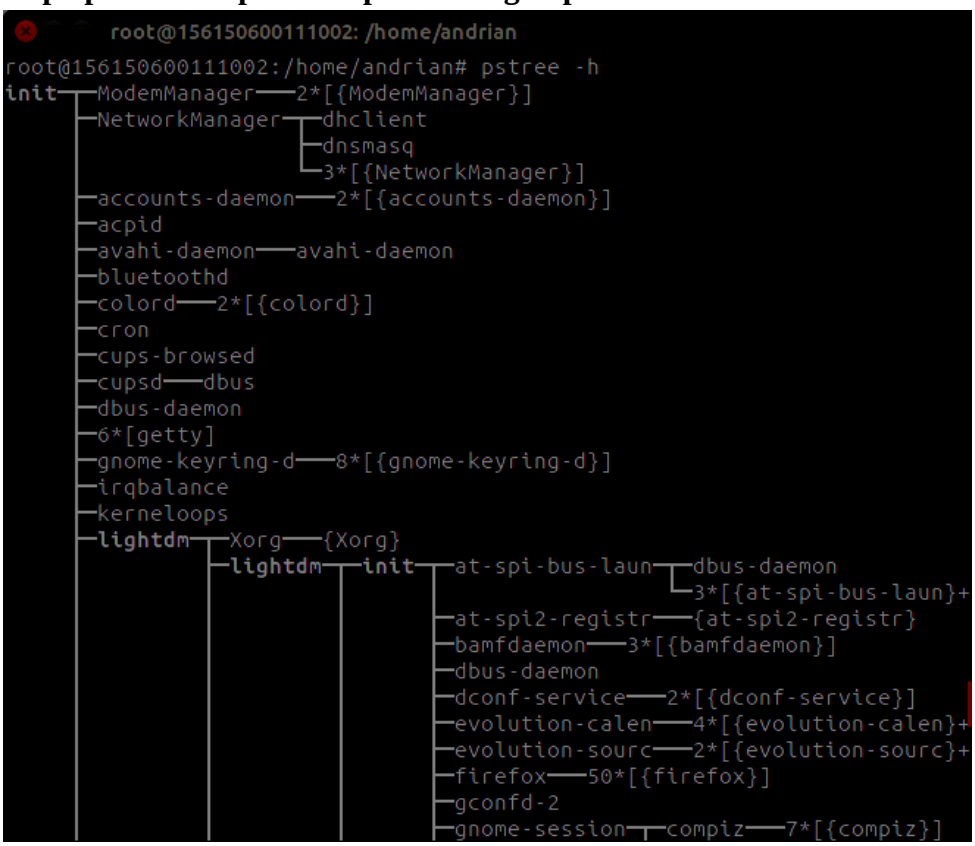
5. **Menampilkan proses yang sedang berjalan dalam bentuk pohon**
   # Untuk menjalankan fungsi pada poin lima, Anda dapat menggunakan perintah seperti di bawah ini:

```
pstree
```

 # Bagaimana hasil ouputnya?



#Apa perbedaan perintah pstree dengan pstree –h?



Pada pstree –h, percabangan yang ditampilkan akan lebih mendetail hingga ke cabang terluar.

CONTROLLING PROSES

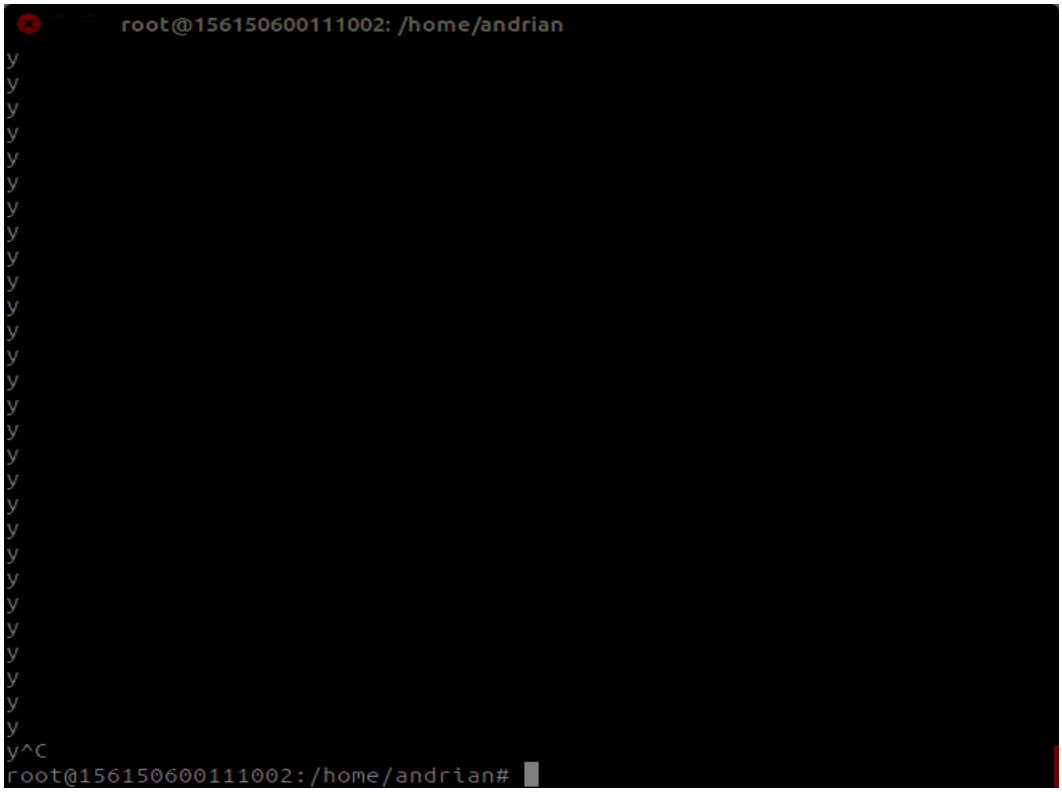
Dalam sistem operasi linux semua proses dikontrol oleh shell yaitu sebuah antar muka antara sistem operasi dengan user. Sebagai contohnya kita akan mengirim output y yang tidak akan pernah selesai. Dengan menggunakan sintak sebagai berikut:

```
$ yes
```

Bagaimana untuk menghentikan perintah $yes di atas?

```
Ctrl+C
```

Hasil output:



Bagaimana cara agar perintah yes tetap dijalankan tetapi shell tetap digunakan untuk hal yang lain?

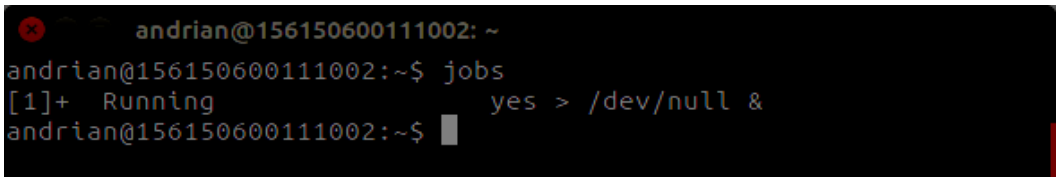
```
$ yes > /dev/null &
```

Hasil output:

Bagaimana umelihat status proses yang sedang berjalan?

```
$ jobs
```

Hasil output:



PERINTAH KILL

Perintah kill adalah salah satu perintah daasar Linux yang digunakan untuk menghentikan atau mematikan proses yang sedang berjalan pada Sistem Operasi Linux / UNIX. perintah ini sangat penting karena dengan memahami perintah ini kita bisa mengetahui mana proses yang mengganggu performa, tidak dibutuhkan, dll.

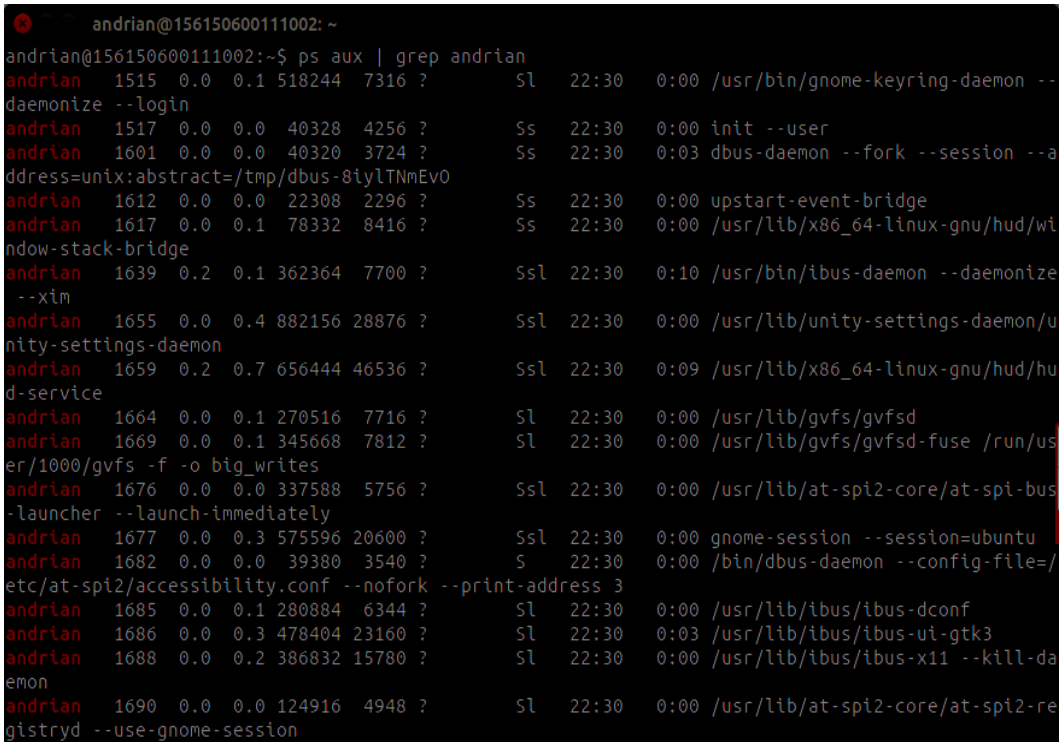
#Bagaimana contoh perintah kill?

```
$ kill 3053
```

 # PID adalah nomor proses yang akan dihentikan. Tidak tahu PID proses mana yang akan dihentikan? Cobalah bereksperimen dengan perintah:

```
ps aux | grep <myusername>
```

 # Lalu tempelkan hasil output pada kolom di bawah ini!



Daemons

Daemons adalah sebuah proses yang bekerja pada background karena proses ini tidak memiliki terminal pengontrol. Dalam sistem operasi Windows biasanya lebih dikenal dengan sebutan service. Daemon adalah sebuah proses yang didesain supaya proses tersebut tidak mendapatkan intervensi dari user. Daemon biasanya bekerja dalam jangka waktu yang sangat lama dan bertugas menerima request dan menjalankan responsnya.

#Apa yang membedakan Daemons dengan proses lain?

1. Daemon tidak memiliki parent proses ID.
2. Daemon tidak memiliki pengontrol baik itu STDOUT, STDIN, maupun STDERR.
3. Daemon berjalan dalam previlege super user.

Berikut ini adalah beberapa cara untuk membuat daemon:

a. Forking dan pembunuhan Proses induk

Langkah pertama dari pembuatan daemon adalah dengan menspawn proses menjadi induk dan anak dengan melakukan forking, kemudian membunuh proses induk. Proses induk yang mati akan menyebabkan sistem operasi mengira bahwa proses telah selesai sehingga akan kembali ke terminal user.

 # Contoh Script:

*script.sh ×

```
pid_t pid,sid;
pid=fork();
if (pid<0){
exit(EXIT_FAILURE);
}
if (pid>0){
exit(EXIT_SUCCESS);
}
```

sh ▾ Tab Width: 8 ▾ Ln 10, Col 1 INS

b. Membuat proses bekerja secara independen

Daemon harus bekerja secara independen daripada proses-proses lain, termasuk juga proses yang menjalankannya. Langkah bisa dilakukan dengan memanggil fungsi setsid(), sehingga proses akan mendapatkan sebuah session ID yang baru.

 # Contoh Script:

*script.sh ×

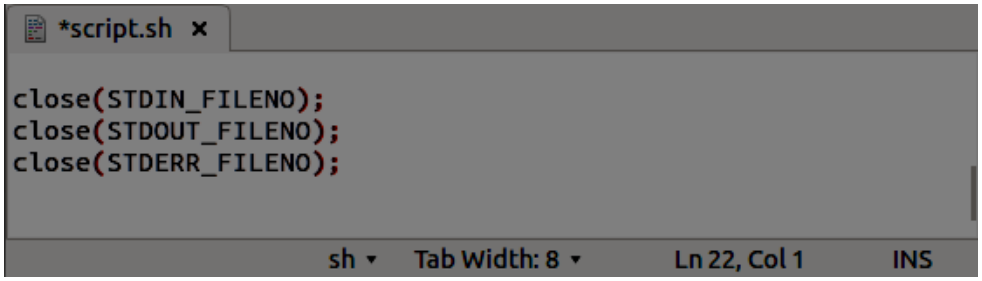
```
sid = setsid();
if (pid<0){
exit (EXIT_FAILURE);
}
```

sh ▾ Tab Width: 8 ▾ Ln 22, Col 1 INS

c. Menutup standar I/O deskriptor yang diwarisi

Untuk mencegah terjadinya intervensi dari user serta untuk pengamanan, maka standar I/O descriptor dan descriptor yang diwarisi dari proses induk harus ditutup. Ada 3 jenis standar I/O descriptor : STDIN (standar input), STDOUT (standar output), STDERR (standar error).

 # Contoh Script:



d. Melakukan masking pada File Creation

Sebagian besar daemon bekerja dalam previlege super user. Daemon biasanya memproteksi setiap file yang dibuat, dengan alasan keamanan. Fungsi **umask()** akan mencegah file-file previleges yang tidak aman dalam setiap pembuatan file.

 # Contoh Script:

e. Running Directory

Directory kerja daemon haruslah sebuah directory yang selalu hidup. Bisa saja pada saat starting working directorynya pada saat itu berada pada user home. Karena daemon bekerja sampai sistem reboot, maka file sistem user directorynya takkan pernah bisa di unmount.

 # Contoh Script:

f. Mendengarkan signal

 # Tulislah pada kolom di bawah ini, apakah maksud dari proses ini?

Tugas utama dari daemon adalah mendengarkan requst. Maka didalam daemon haruslah terdapat pendangar signal yang dapat merespon ketika daemon dikirimi signal tertentu, hal ini dapat dilakukan dengan meamnggil fungsi signal() untuk menginstall sebuah signal listener. Perlu kita ketahui bahwa signal 15 (SIGTERM) dan signal 9 (SIGKILL) tidak dapat ditangkap oleh signal handler.

g. Logging

 # Sebutkan beberapa cara untuk melakukan proses logging dan berikan penjelasan?

1. Metode log file : Semua file ditulis kedalam sebuah file tertentu yang diatur oleh file komfigurasi daemon tersebut menggunakan fungsi fopen(). 2. Metode log server : Untuk keluarga UNIX mereka meimiliki daemon khusus yang digunakan untuk logging yaitu syslogd. Daemon ini mengelompokkan pesan-pesan menjadi beberapa kelompok yang disebut facility, kemudian kelompok-kelompok ini dapat dikirim ke tempat-tempat yang berbeda, misalnya langsung dikirim ke sysadmin lewat e-mail, dikirimkan kepada semua console terminal yang sedang log in, atau ditulis dalam suatu file logger.konfigurasi dari daemon syslogd ini dapat ditulis dalam sebuah file /etc/syslog.conf.

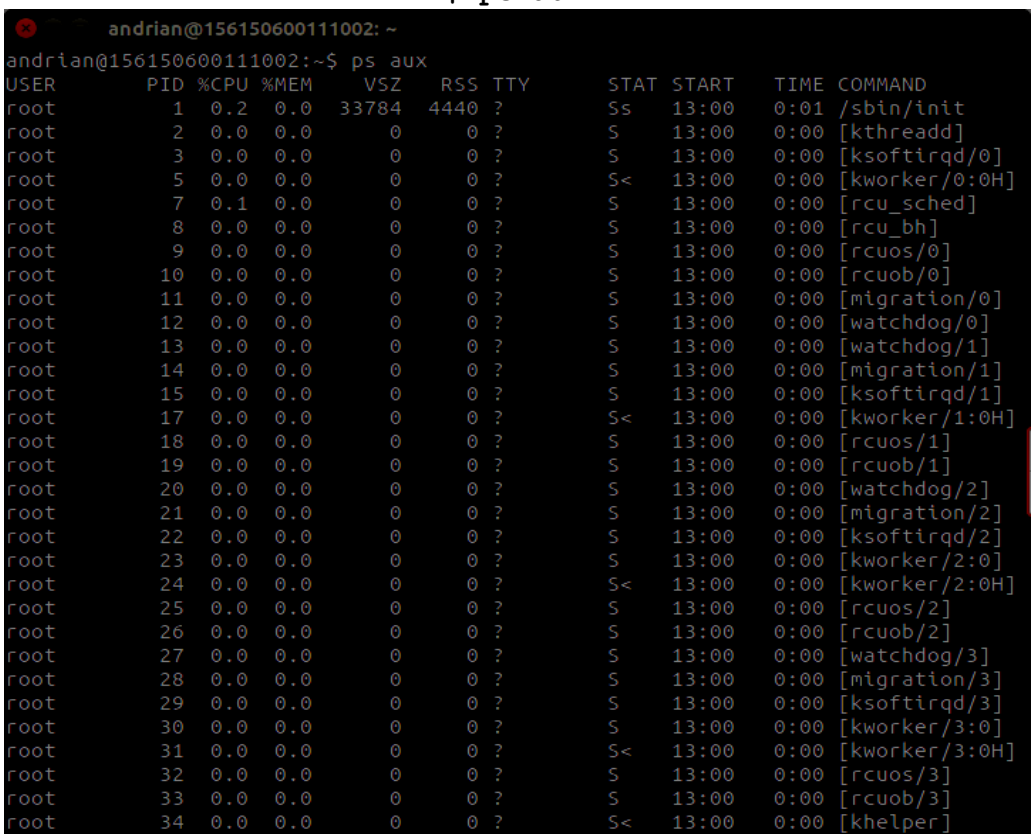
LAPORAN PRAKTIKUM SISTEM OPERASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

Nama	: Moh. Arif Andrian
NIM	: 156150600111002
Tugas	: BAB II
Asisten	: Siska Permatasari\
	Zaenal Kurniawan

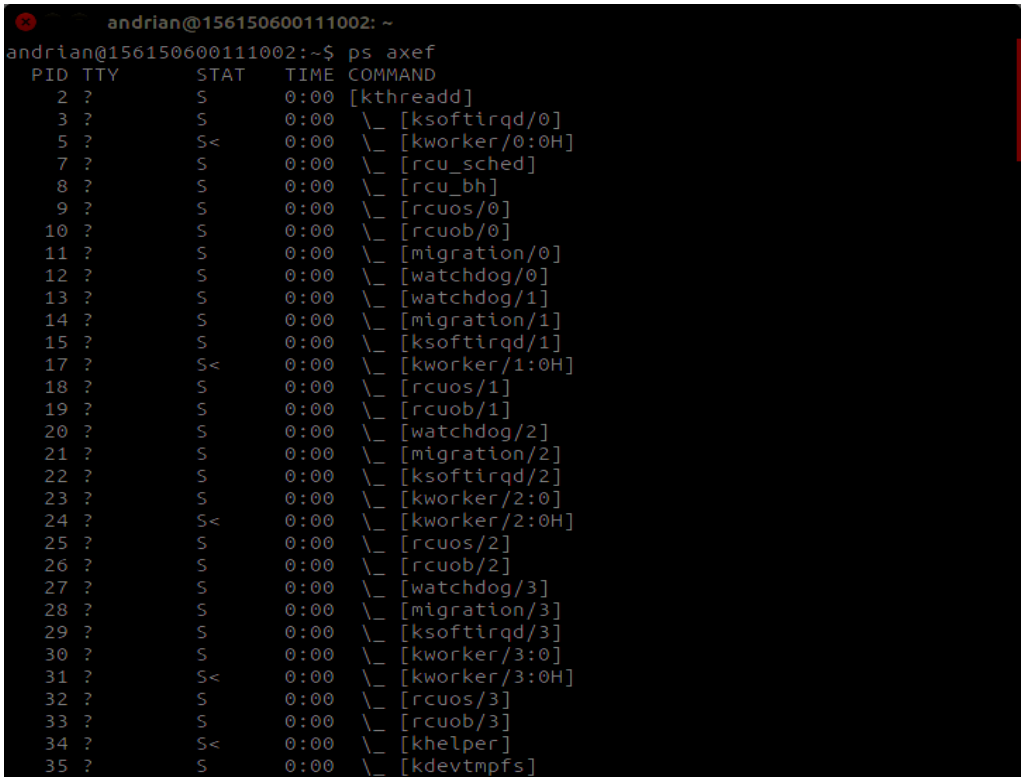
TUGAS PRAKTIKUM

1. Carilah defunct process pada sistem operasi linux anda dengan cara mengetikkan script berikut pada terminal :

$ ps aux

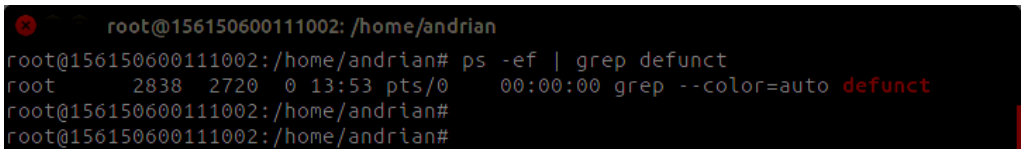


atau juga bisa dengan mengetikkan perintah " **ps axef** " pada terminal, tapi untuk yang lebih compleks gunakan " **ps aux** " .



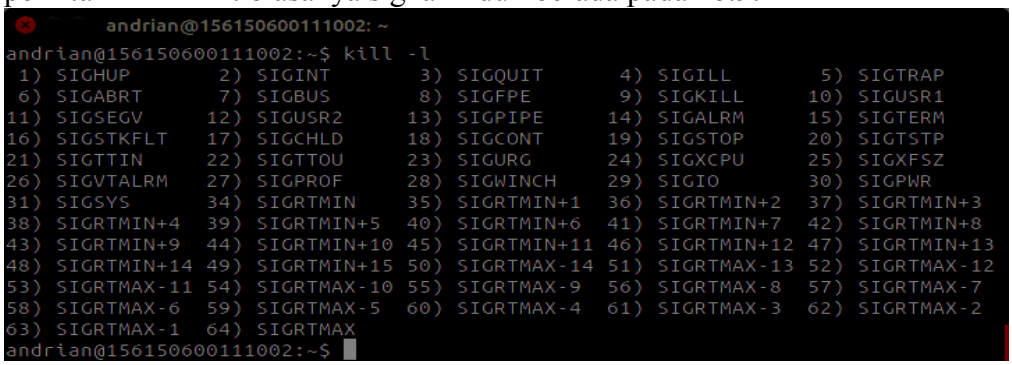
untuk lebih mudahnya mencari defunct proses langsung ketikkan saja.

" **ps -ef | grep defunct** ".

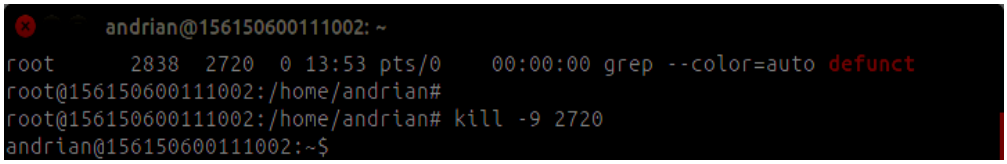


2. Matikan defunct process yang telah ditemukan

pertama, cari signal defunct proses terlebih dahulu, dengan mengetikkan perintah " **kill -l** ". biasanya signal induk berada pada no.9.



kedua, matikan defunct proses dengan cara mengetikkan perintah berikut pada terminal. " **$ kill -9 <no. PID induk>**



LAPORAN PRAKTIKUM SISTEM OPERASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

Nama : Moh. Arif Andrian
NIM : 156150600111002
Laporan : BAB II
Asisten : Siska Permatasari
Zaenal Kurniawan

KESIMPULAN

Proses adalah program yang sedang diekseskusi. Setiap kali menjalankan program, sistem UNIX akan menjalankan fork, yaitu menjalankan beberapa urutan operasi untuk membuat suatu proses konteks, dan menjalankannya pada konteks yang telah dibuat.

Pada Linux Ubuntu Terdapat beberapa tipe proses yang dikenal dalam OS berbasis Linux pada umumnya, antara lain :

1. **Interactive** : proses yang dilakukan oleh pemakai langsung pada terminal. Bisa tampak di luar atau hanya di dalam.
2. **Batch** : Proses yang tidak berhubungan dengan terminal. Proses ini dijalankan secara sekuensial (satu persatu).
3. **Daemon** : Proses yang menunggu permintaan dari proses lainnya dan menjalankan tugas sesuai dengan permintaan tersebut. Bila tidak ada proses, kondisinya menjadi "idle".

Perbedaan Process dan Thread

Process	Thread
• Program yang dieksekusi • Mencakup program counter,yaitu sebuah stack untuk menyimpan alamat dari instruksi yang selanjutnya akan dieksekusi • Memiliki IP adress masing-masing • Dari satu process dengan process lainnya harus menggunakan komunikasi • Perubahan pada parent proses tidak mempengaruhi turunannya • Waktu yang dibutuhkan untuk mengakhiri process lebih lama.	• Adalah alur kontrol dari suatu proses • Merupakan unit dasar dari pengguna CPU dan sering disebut dengan lightwight process • IP adress digunakan secara bersamasama • Dapat berkomunikasi dengan thread lain dalam satu process • Hampir tidak memiliki overhead • Perubahan pada thread utama dapat mempengaruhi tingkah laku trhead lain dalam satu waktu • Waktu yang dibutuhkan untuk mengakhiri thread lebih sedikit • Lebih mudah melakukan switch antar thread dari pada switch antar process.

- Untuk menghentikan suatu proses dapat menggunakan syntak $kill. Perintah ini sangat penting karena dengan memahami perintah ini kita bisa mengetahui mana proses yang mengganggu performa, tidak dibutuhkan dll.
- Pstree adalah syntak yang akan menampilkan semua prosess pada sistem dalam bentuk hirarki parent/child dimana parent berada di sebelah kiri proses child.