Nama: Chandra Bagus Sulaksono

NIM: 2241760079

**Praktikum 4 B**

Proses dan Manajemen Proses

# POKOK BAHASAN:

* Proses pada Sistem Operasi Linux
* Manajemen Proses pada Sistem Operasi Linux

# TUJUAN BELAJAR:

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

* Memahami konsep proses pada sis tem operasi Linux.
* Menampilkan beberapa cara menampilkan hubungan proses parent dan child.
* Menampilkan status proses dengan beberapa format berbeda.
* Melakukan pengontrolan proses pada shell.
* Memahami penjadwalan prioritas.

# DASAR TEORI:

1. **KONSEP PROSES PADA SISTEM OPERASI LINUX**

Proses adalah program yang sedang dieksekusi. Setiap kali menggunakan utilitas sistem atau program aplikasi dari shell, satu atau lebih proses ”child” akan dibuat oleh shell sesuai perintah yang diberikan. Setiap kali instruksi dibe rikan pada Linux shell, maka kernel akan menciptakan sebuah proses-id. Proses ini disebut juga dengan terminology Unix sebagai sebuah Job. Proses Id (PID) dimulai dari 0, yaitu proses INIT, kemudian diikuti oleh proses berikutnya (terdaftar pada /etc/inittab).

Beberapa tipe proses :

## Foreground

Proses yang diciptakan oleh pemakai langsung pada terminal (interaktif, dialog)

31

## Batch

Proses yang dikumpulkan dan dijalankan secara sekuensial (satu persatu). Prose Batch tidak diasosiasikan (berinteraksi) dengan terminal.

## Daemon

Proses yang menunggu permintaan (request) dari proses lainnya dan menjalankan tugas sesuai dengan permintaan tersebut. Bila tidak ada request, maka program ini akan berada dalam kondisi “idle” dan tidak menggunakan waktu hitung CPU. Umumnya nama proses daemon di UNIX berakhiran d, misalnya inetd, named, popd dll

# SINYAL

Proses dapat mengirim dan menerima sinyal dari dan ke proses lainnya. Proses mengirim sinyal melalui instruksi “kill” dengan format

kill [-nomor sinyal] PID

Nomor sinyal : 1 s/d maksimum nomor sinyal yang didefinisikan system Standar nomor sinyal yang terpenting adalah :

|  |  |  |
| --- | --- | --- |
| **No Sinyal** | **Nama** | **Deskripsi** |
| 1 | SIGHUP | Hangup, sinyal dikirim bila proses terputus,  misalnya melalui putusnya hubungan modem |
| 2 | SIGINT | Sinyal interrupt, melalui ^C |
| 3  9 | SIGQUIT  SIGKILL | Sinyal Quit, melalui ^\  Sinyal Kill, menghentikan proses |
| 15 | SIGTERM | Sinyal terminasi software |

# MENGIRIM SINYAL

Mengirim sinyal adalah satu alat komunikasi antar proses, yaitu memberitahukan proses yang sedang berjalan bahwa ada sesuatu yang harus dikendalikan. Berdasarkan sinyal yang dikirim ini maka proses dapat bereaksi dan

administrator/programmer dapat menentukan reaksi tersebut. Mengirim sinyal menggunakan instruksi

kill [-nomor sinyal] PID

Sebelum mengirim sinyal PID proses yang akan dikirim harus diketahui terlebih dahulu.

# MENGONTROL PROSES PADA SHELL

Shell menyediakan fasilitas job control yang memungkinkan mengontrol beberapa job atau proses yang sedang berjalan pada waktu yang sama. Misalnya bila melakukan pengeditan file teks dan ingin melakukan interrupt pengeditan untuk mengerjakan hal lainnya. Bila selesai, dapat kembali (*switch*) ke editor dan melakukan pengeditan file teks kembali.

Job bekerja pada **foreground** atau **background**. Pada *foreground* hanya diper untukkan untuk satu job pada satu waktu. Job pada *foreground* akan mengontrol shell - menerima input dari keyboard dan mengirim output ke layar. Job pada background tidak menerima input dari terminal, biasanya berjalan tanpa memerlukan interaksi.

Job pada *foreground* kemungkinan dihentikan sementara (suspend), dengan menekan [Ctrl-Z]. Job yang dihentikan sementara dapat dijalankan kembali pada *foreground* atau *background* sesuai keperluan dengan menekan ”**fg**” atau ”**bg** ”. Sebagai catatan, menghentikan job seme ntara sangat berbeda dengan melakuakan interrupt job (biasanya menggunakan [Ctrl-C]), dimana job yang diinterrup akan dimatikan secara permanen dan tidak dapat dijalankan lagi.

# MENGONTROL PROSES LAIN

Perintah ps dapat digunakan untuk menunjukkan semua proses yang sedang berjalan pada mesin (bukan hanya proses pada shell saat ini) dengan format :

ps –fae atau

ps -aux

Beberapa versi UNIX mempunyai utilitas sistem yang disebut top yang menyediakan cara interaktif untuk memonitor aktifitas sistem. Statistik secara detail

dengan proses yang berjalan ditampilkan dan secara terus-menerus di-*refresh* . Proses ditampilkan secara terurut dari utilitas CPU. Kunci yang berguna pada top adalah

s – set update frequency

u – display proses dari satu user

k – kill proses (dengan PID)

q – quit

Utilitas untuk melakukan pengontrolan proses dapat ditemukan pada sistem UNIX adalah perintah killall. Perintah ini akan menghentikan proses sesuai PID atau job number proses.

# Percobaan 5 : Menghentikan dan memulai kembali job

1. Cara lain meletakkan job pada *background* dengan memulai job secara normal (pada *foreground*), **stop** job dan memulai lagi pada *background*

$ *yes > /dev/null*

Hentikan sementara job (*suspend* ), bukan menghentikannya (*terminate* ), tetapi menghentikan sementara job sampai di restart. Untuk menghentikan sementara job gunakan **Ctrl-Z**.

1. Untuk restart job pada *foreground* , gunakan perintah fg.

$ *fg*

1. Shell akan menampilkan nama perintah yang diletakkan di *foreground* . Stop

job lagi dengan **Ctrl-Z**. Kemudian gunakan perintah bg untuk meletakkan job pada *background* .

$ *bg*

Job tidak bisa dihentikan dengan **Ctrl-Z** karena job berada pada *background*.

Untuk menghentikannya, letakkan job pada *foreground* dengan fg dan kemudian hentikan sementara dengan **Ctrl-Z**.

$ *fg*

1. Job pada *background* dapat digunakan untuk menampilkan teks pada terminal, dimana dapat diabaikan jika mencoba mengerjakan job lain.

$ *yes &*

Untuk menghentikannya tidak dapat menggunakan **Ctrl-C.** Job harus dipindah ke *foreground*, baru dihentikan dengan cara tekan **fg** dan tekan **Enter**, kemudian dilanjutkan dengan **Ctrl-Z** untuk menghentikan sementara.

1. Apabila ingin menjalankan banyak job dalam satu waktu, letakkan job pada

*foreground* atau *background* dengan memberikan job ID

$ *fg %2*

$ *bg %2*

atau

$ *%2*

1. tekan **fg** dan tekan **Enter**, kemudian dilanjutkan dengan **Ctrl -Z** untuk menghentikan sementara.
2. Lihat job dengan perintah **ps -fae** dan tekan **Enter**. Kemudian hentikan proses dengan perintah kill.

$ *ps -fae*

$ *kill -9 <NomorPID>*

1. **Logout** dan tekan **Alt+F7** untuk kembali ke mode grafis

**Percobaan 6 : Percobaan dengan Penjadwalan Prioritas**

1. Login sebagai root.
2. Buka 3 terminal, tampilkan pada screen yang sama.
3. Pada setiap terminal, ketik **PS1 = ” \w:”** diikuti **Enter**. **\w** menampilkan path pada direktori home.
4. Karena login sebagai root, maka akan ditampilkan **~:** pada setiap terminal. Untuk setiap terminal ketik **pwd** dan tekan **Enter** untuk melihat bahwa Anda sedang berada pada direktori /root.
5. Buka terminal lagi (keempat), atur posisi sehingga keempat terminal terlihat

pada screen.

* 1. Pada terminal keempat, ketik **top** dan tekan **Enter.** Maka program top akan muncul. Ketik **i**. **Top** akan menampilkan proses yang aktif. Ketik **lmt**. **Top** tidak lagi menampilkan informasi pada bagian atas dari screen. Pada percobaan ini, terminal ke empat sebagai je ndela **Top**.
  2. Pada terminal 1, bukalah program executable C++ dengan mengetik program

**yes** dan tekan **Enter.**

* 1. Ulangi langkah 7 untuk terminal 2.
  2. Jendela **Top** akan menampilkan dua program yes sebagai proses yang berjalan. Nilai %CPU sama pada keduanya. Hal ini berarti kedua proses mengkonsumsi waktu proses yang sama dan berjalan sama cepat. PID dari kedua proses akan berbeda, misalnya 3148 dan 3149. Kemudian gunakan terminal 3 (yang tidak menjalankan primes maupun Jendela **Top**) dan ketik **renice 19 <PID terimnal 1>** (contoh : **renice 19 3148**) dan diikuti **Enter**. Hal ini berarti mengganti penjadwalan prioritas dari proses ke 19.
  3. Tunggu beberapa saat sampai program top berubah dan terlihat pada jendela

**Top**. Pada kolom STAT memperlihatkan N untuk proses 3148. Hal ini berarti bahwa penjadwalan prioritas untuk proses 3148 lebih besar (lebih lambat) dari 0. Proses 3149 berjalan lebih cepat.

* 1. Program top juga mempunyai fungsi yang sama dengan program renice. Pilih Jendela **Top** dan tekan **r**. Program top terdapat prompt **PID to renice:** tekan **3148** (ingat bahwa Anda harus mengganti 3148 dengan PID

Anda sendiri) dan tekan **Enter**. Program top memberikan prompt **Renice PID 3148 to value:** tekan -**19** dan tekan **Enter**.

* 1. Tunggu beberapa saat sampai top berubah dan lihat nilai %CPU pada kedua proses. Sekarang proses 3148 lebih cepat dari proses 3149. Kolom status menunjukkan < pada proses 3148 yang menunjukkan penjadwalan

prioritas lebih rendah (lebih cepat) dari nilai 0.

1. Pilih terminal 3 (yang sedang tidak menjalankan yes atau program top) dan ketik **nice –n -10 yes** dan tekan **Enter.** Tunggu beberapa saat agar

program top berubah dan akan terlihat proses primes ketiga. Misalnya PID nya 4107. Opsi -10 berada pada kolom NI (penjadwalan prioritas).

1. Jangan menggunakan mouse dan keyboard selama 10 detik. Program top menampilkan proses yang aktif selain program yes. Maka akan terlihat proses top terdaftar tetapi %CPU kecil (dibawah 1.0) dan konsisten. Juga terlihat proses berhubungan dengan dekstop grafis seperti X, panel dll.
2. Pindahkan mouse sehingga kursor berubah pada screen dan lihat apa yang

terjadi dengan tampilan top. Proses tambahan akan muncul dan nilai

%CPU berubah sebagai bagian grafis yang bekerja. Satu alasan adalah bahwa proses 4107 berjalan pada penjadwalan prioritas tinggi. Pilih jendela **Top**, ketik **r**. **PID to renice** : muncul prompt. Ketik **4107** (ubahlah 4107 dengan PID Anda) dan tekan **Enter**. **Renice PID 4107 to value:** muncul prompt. Ketik **0** dan tekan **Enter**. Sekarang pindahkan mouse ke sekeliling screen. Lihat perubahannya.

1. Tutup semua terminal window.
2. Logout dan login kembali sebagai user.

# LATIHAN:

1. Masuk ke tty2 dengan **Ctrl+Alt+F2**. Ketik **ps – au** dan tekan **Enter.** Kemudian perhatikan keluaran sebagai berikut :
   1. Sebutkan nama -nama proses yang bukan root
   2. Tulis PID dan COMMAND dari proses yang paling banyak menggunakan CPU time
   3. Sebutkan buyut proses dan PID dari proses tersebut
   4. Sebutkan beberapa proses daemon
   5. Pada prompt login lakukan hal- hal sebagai berikut :

$ *csh*

$ *who*

$ *bash*

$ *ls*

$ *sh*

$ *ps*

* 1. Sebutkan PID yang paling besar dan kemudian buat urut-urutan proses sampai ke PPID = 1.

1. Cobalah format tampilan ps dengan opsi berikut dan perhatikan hasil tampilannya :

* **-f** daftar penuh
* **-j** format job
* **j** format job control
* **l** daftar memanjang
* **s** format sinyal
* **v** format virtual memory
* **X** format register i386

1. Lakukan urutan pekerjaan berikut :
   1. Gunakan perintah find ke seluruh direktory pada sistem, belokkan output sehingga daftar direktori dialihkan ke file directories.txt dan daftar pesan error dialihkan ke file errors.txt
   2. Gunakan perintah sleep 5. Apa yang terjadi dengan perintah ini ?
   3. Jalankan perintah pada *background* menggunakan &
   4. Jalankan sleep 15 pada *foreground* , hentikan sementara dengan Ctrl- Z dan kemudian letakkan pada *background* dengan bg. Ketikkan jobs. Ketikkan ps. Kembalikan job ke *foreground* dengan perintahfg.
   5. Jalankan sleep 15 pada *background* menggunakan & dan kemudian gunakan perintah kill untuk menghentikan proses diikuti *job number*.
   6. Jalankan sleep 15 pada *background* menggunakan & dan kemudian gunakan kill untuk menghentikan sementara proses. Gunakan bg untuk melanjutkan menjalankan proses.
   7. Jalankan sleep 60 pada *background* 5 kali dan terminasi semua pada dengan menggunakan perintah killall.
   8. Gunakan perintah ps, w dan top untuk menunjukkan semua proses yang sedang dieksekusi.
   9. Gunakan perintah ps –aeH untuk menampilkan hierarki proses. Carilah init proses. Apakah Anda bisa identifikasi sistem daemon yang penting ? Dapatkan Anda identifikasi shell dan subprose s ?
   10. Kombinasikan ps –fae dan grep, apa yang Anda lihat ?
   11. Jalankan proses sleep 300 pada *background*. Log off komputer dan log in kembali. Lihat daftar semua proses yang berjalan. Apa yang terjadi pada proses sleep ?

# LAPORAN RESMI:

1. Analisa hasil percobaa n yang Anda lakukan.
2. Kerjakan latihan diatas dan analisa hasil tampilannya.
3. Berikan kesimpulan dari praktikum ini.
4. Cara lain meletakkan job pada *background* dengan memulai job secara normal (pada *foreground*), **stop** job dan memulai lagi pada *background*

$ *yes > /dev/null*

Hentikan sementara job (*suspend* ), bukan menghentikannya (*terminate* ), tetapi menghentikan sementara job sampai di restart. Untuk menghentikan sementara job gunakan **Ctrl-Z**.

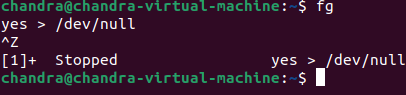
Hasil:

Analisis:

Perintah $ yes > /dev /null di pergunakan untuk memulai job, job dapat dihentikan sementara atau sering di sebut (suspend) dengan menggunakan Ctrl – Z.

1. Untuk restart job pada foreground , gunakan perintah fg.

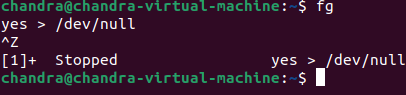
$ fg

Hasil:

1. $ bg

$ fg

Hasil :

Analisis:

Hal pertama kali yang dilakukan adalah mensuspend job dengan yes >/dev/null. Kemudian memindahkannya ke background proses, yang berarti proses atau job tersebut tidak berhenti,akan tetapi terus berjalan di belakang layar. Selanjutkan job yang berada pada background tidak bisa langsung dihentikan dengan Ctrl + Z. Untuk menghentikannya kita harus memindahkan job tersebut ke foreground lalu mensuspendnya.

1. Job pada background dapat digunakan untuk menampilkan teks pada terminal, dimana dapat diabaikan jika mencoba mengerjakan job lain.

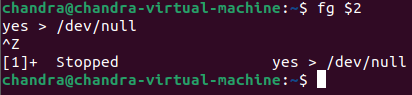
$ yes &

Untuk menghentikannya tidak dapat menggunakan Ctrl-C. Job harus dipindah ke foreground, baru dihentikan dengan cara tekan fg dan tekan Enter, kemudian dilanjutkan dengan Ctrl-Z untuk menghentikan sementara.

Analisis:

Pada saat kita menekan fg, muncul hasil dari perintah $yes & tadi. Jadi fungsi perintah fg disini adalah untuk memanggil proses yang berjalan di background. Dan kita bisa menghentikannya dengan menekan Ctrl + Z.

1. Apabila ingin menjalankan banyak job dalam satu waktu, letakkan job pada *foreground* atau *background* dengan memberikan job ID

**$ *fg %2*

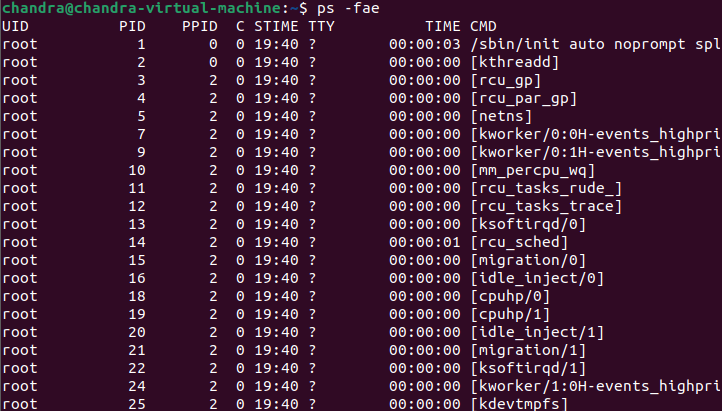
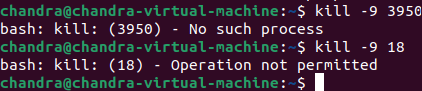
Analisa:

Perintah $ fg %2 dan $bg %2 disini berguna untuk menjalankan banyak job dalam waktu bersamaan. Job – job yang sedang tidak dipakai dimasukkan ke proses background, dan untuk mengenalinya kita menggunakan nomor job ID karena tidak ada job yang memiliki nomor ID sama.

1. Lihat job dengan perintah ps -fae dan tekan Enter. Kemudian hentikan proses dengan perintah kill.

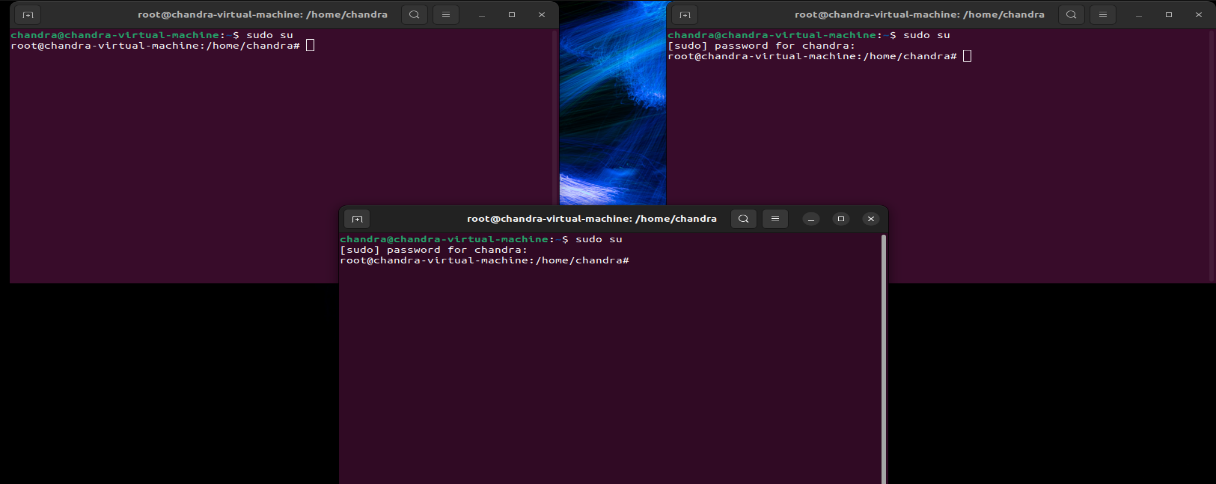
$ *ps -fae*

$ *kill -9 <NomorPID>*

Hasil :

**Percobaan 6**

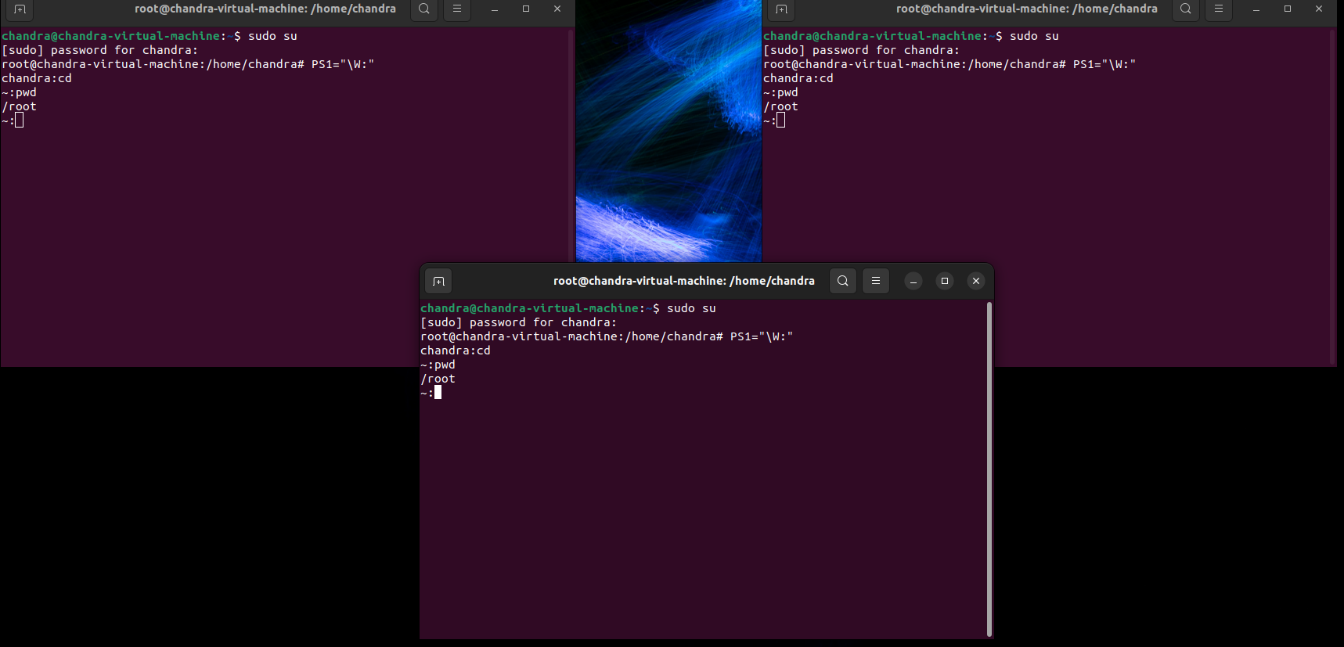
1. Pada setiap terminal, ketik **PS1 = ” \w:”** diikuti **Enter**. **\w** menampilkan path pada direktori home.

Hasil :

Analisis:

Pertama-tama, buka terminal dan bagi menjadi 3. Caranya dapat dilakukan dengan menambah tab baru di terminal, lalu klik kanan dan klik detach terminal untuk memisahkannya. Selanjutnya, memasukkan sudo su untuk masuk sebagai root. Perintah $ PS1=”\w:” diatas bertujuan untuk masuk ke directory home dari user root. Bila posisi kita masih berada di /home/<user>, maka kita bisa gunakan perintah $cd ~ untuk masuk ke root. Dan untuk mengeceknya kita gunakan perintah $pwd.

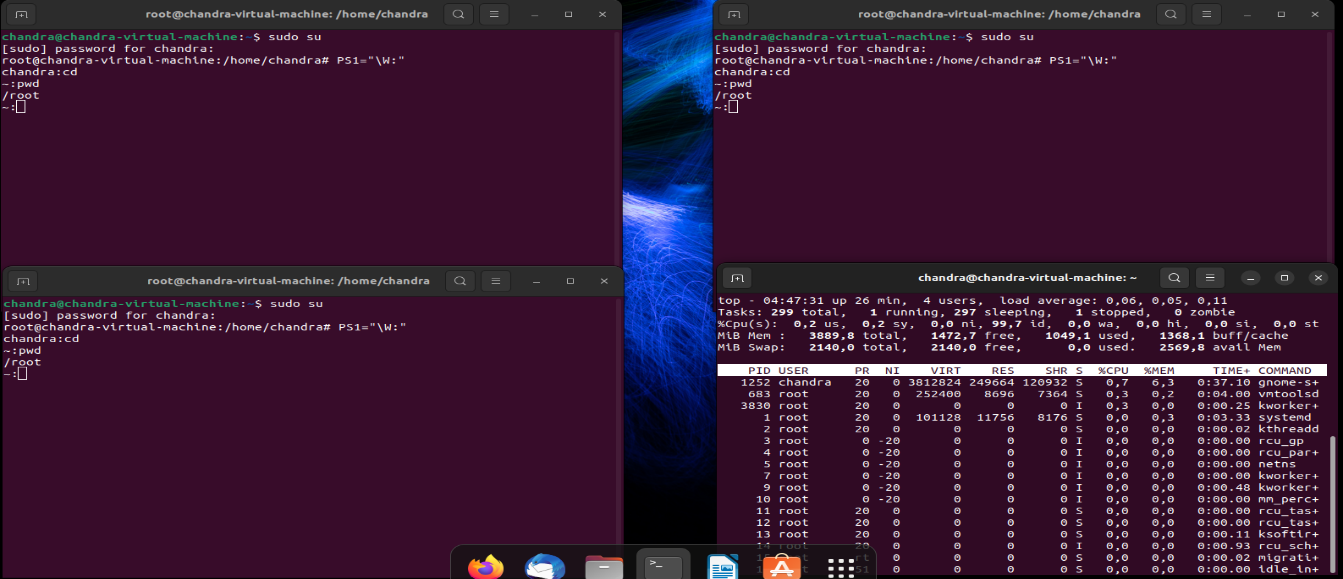
1. Karena login sebagai root, maka akan ditampilkan ~: pada setiap terminal. Untuk setiap terminal ketik pwd dan tekan Enter untuk melihat bahwa Anda sedang berada pada direktori /root.

Hasil:

Analisis:

$cd ~ untuk masuk ke root. Dan untuk mengeceknya kita gunakan perintah $pwd.

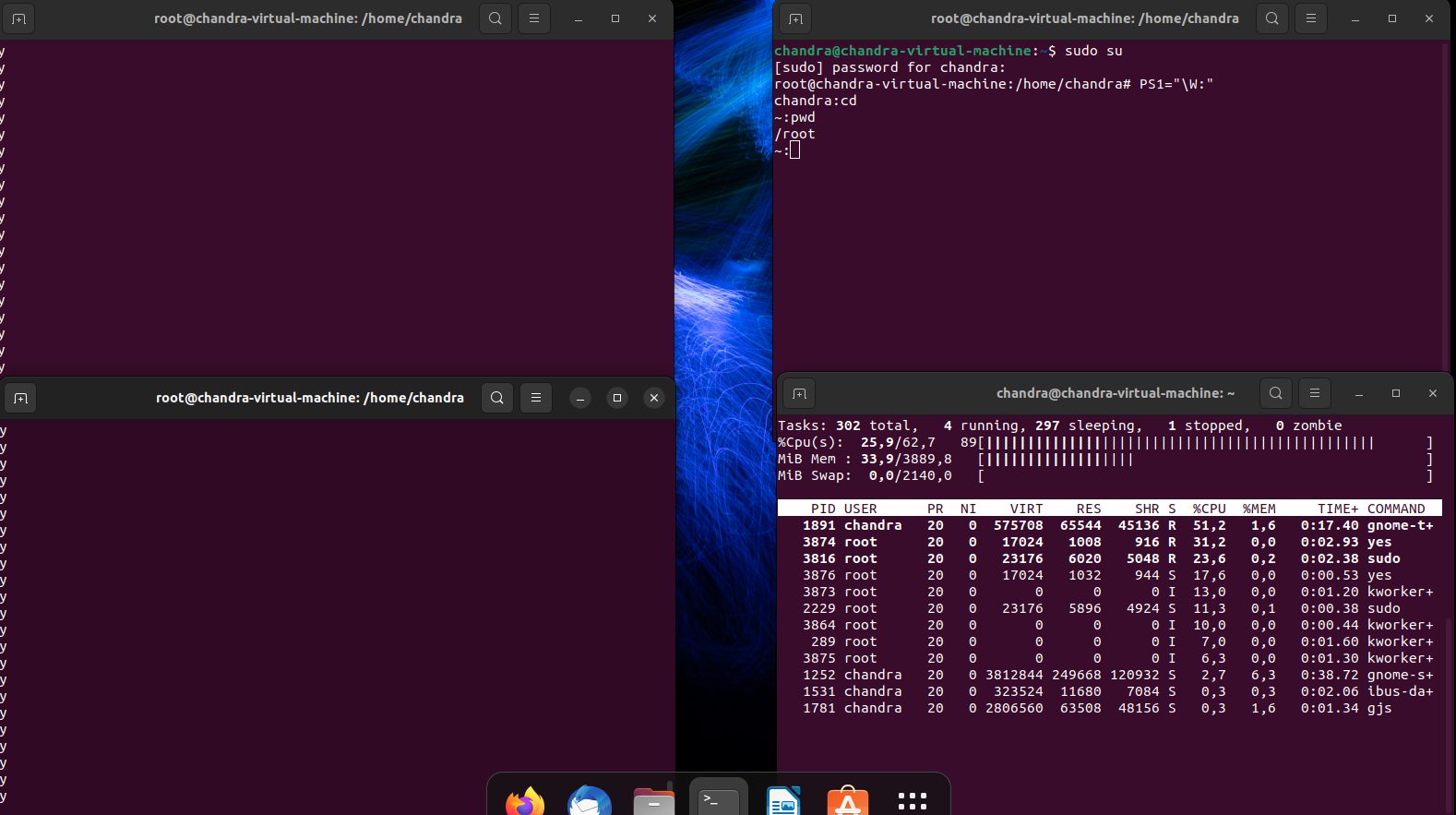
3. Pada terminal keempat, ketik top dan tekan Enter. Maka program top akan muncul. Ketik i. Top akan menampilkan proses yang aktif. Ketik lmt. Top tidak lagi menampilkan informasi pada bagian atas dari screen. Pada percobaan ini, terminal ke empat sebagai jendela Top



Hasil:

4. Pada terminal 1, bukalah program executable C++ dengan mengetik program

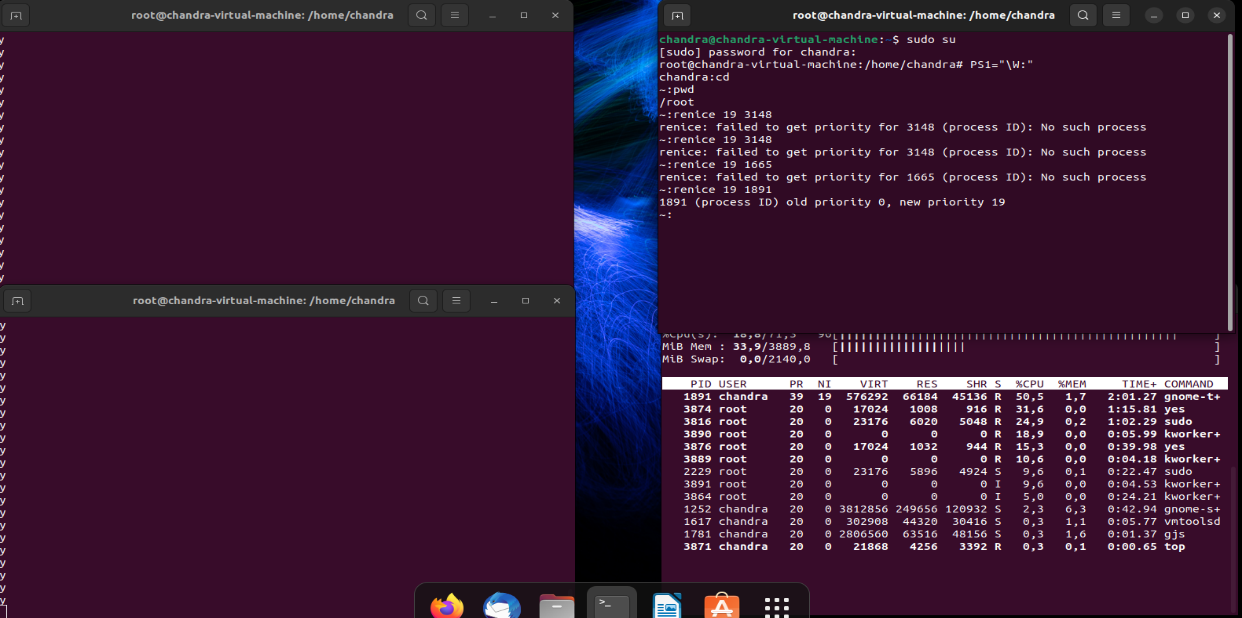
**yes** dan tekan **Enter**

Hasil:

Analisis:

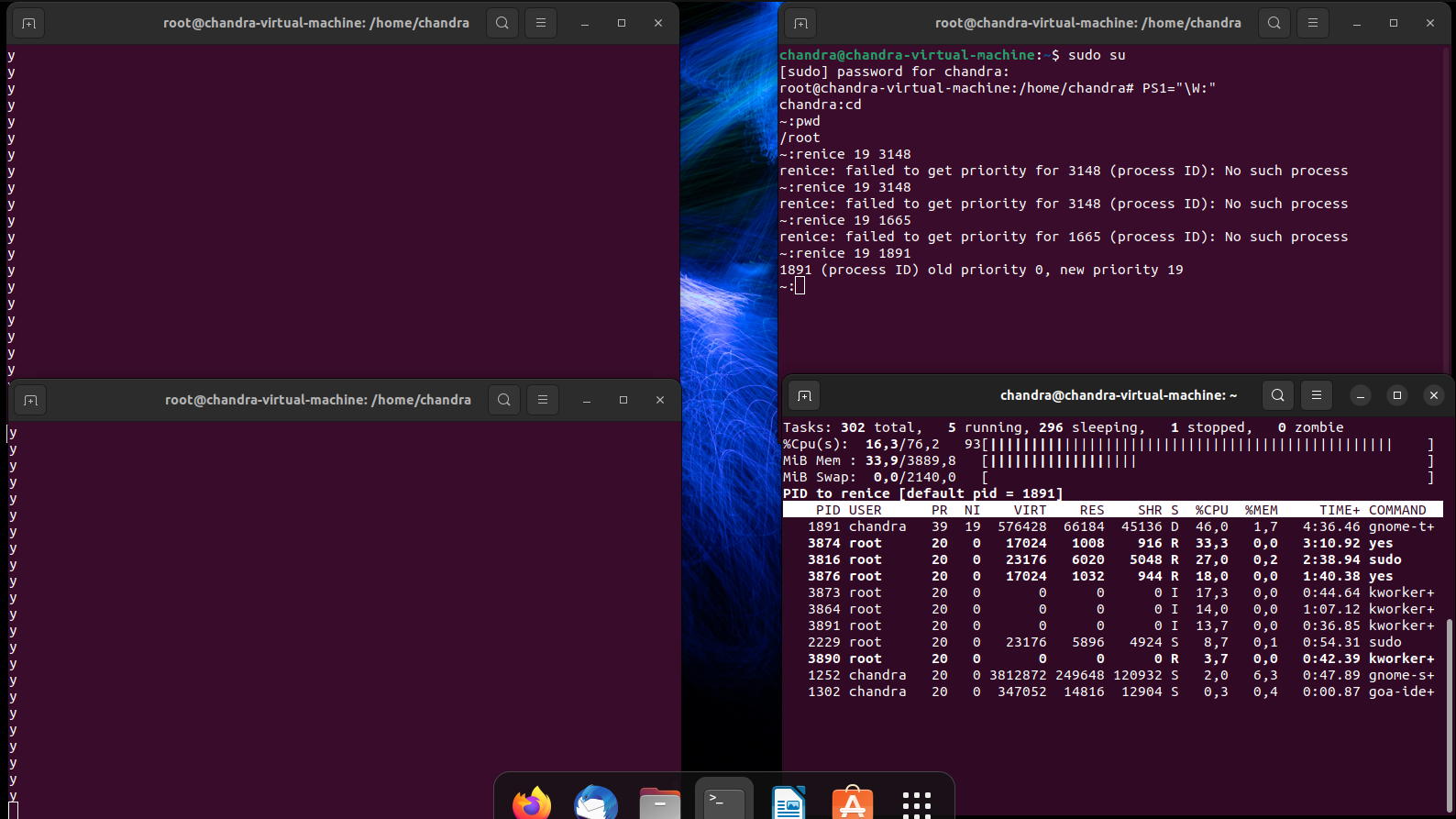
Karena pada percobaan ini menggunakan program sendiri yang perintah eksekusinya adalah ./a.out, maka yang muncul adalah angka ganjil antara angka 1 sampai 10, dan itu diulang-ulang terus tidak berhenti.

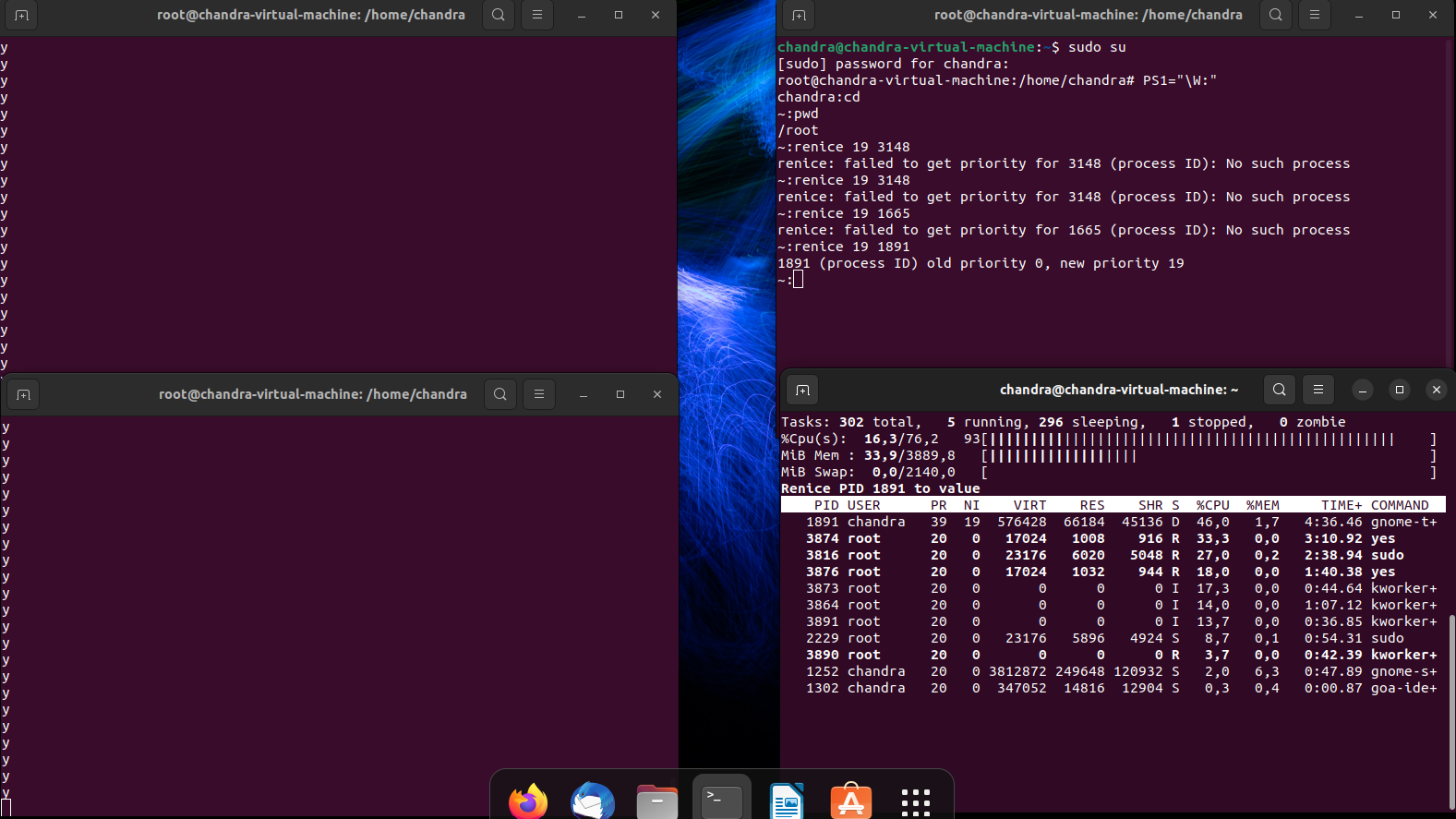
5. Jendela Top akan menampilkan dua program yes sebagai proses yang berjalan. Nilai %CPU sama pada keduanya. Hal ini berarti kedua proses mengkonsumsi waktu proses yang sama dan berjalan sama cepat. PID dari kedua proses akan berbeda, misalnya 3148 dan 3149. Kemudian gunakan terminal 3 (yang tidak menjalankan primes maupun Jendela Top) dan ketik renice 19 <PID terimnal 1> (contoh : renice 19 3148) dan diikuti Enter. Hal ini berarti mengganti penjadwalan prioritas dari proses ke 19.

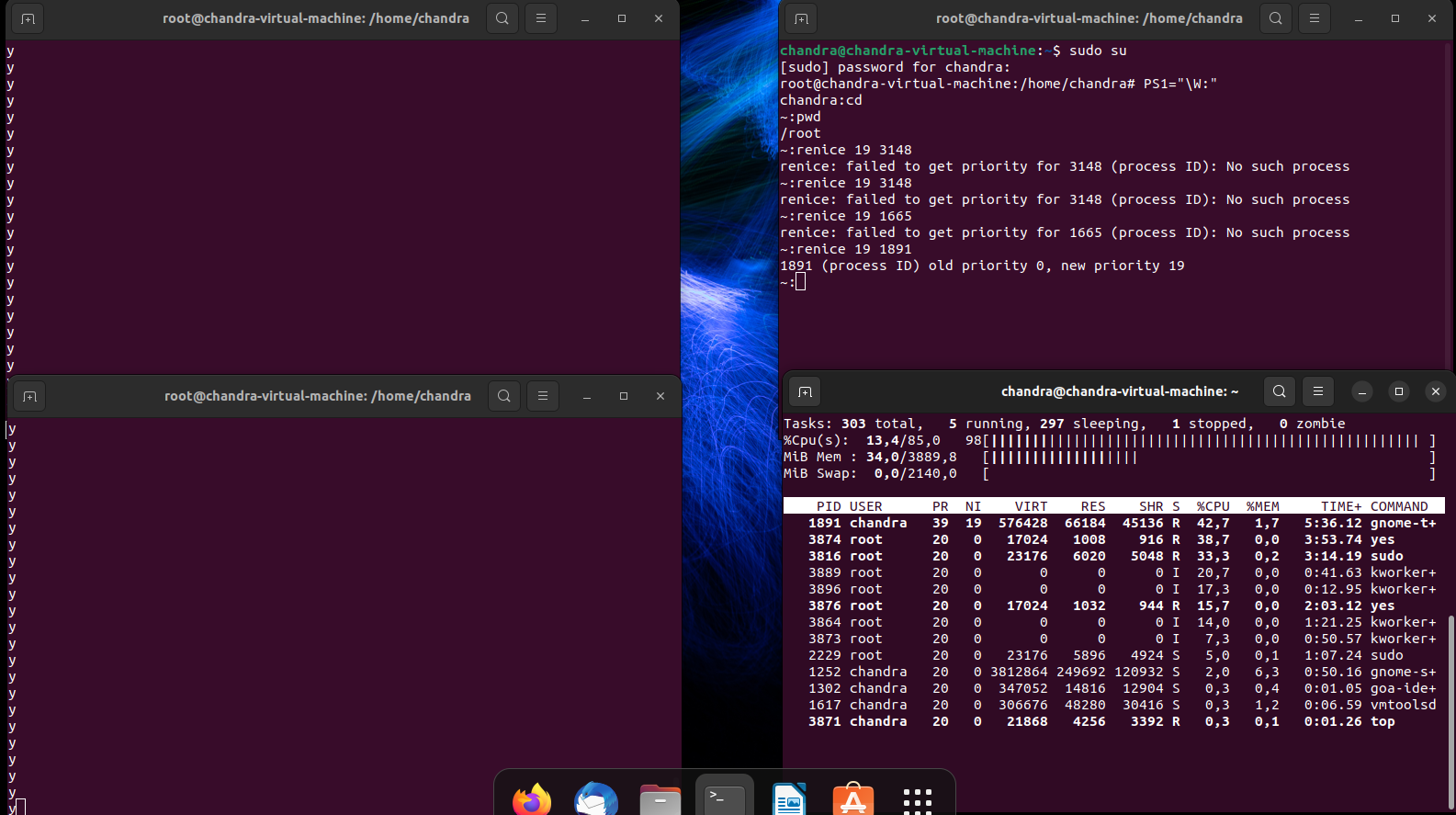
Hasil:

Analisis: Disini PID dua program yang telah kita jalankan tadi adalah 1891 dan 3871. Dua proses yang sama akan menggunakan sumber daya yang sama (mendekati), misalkan dua program yang memiliki PID 1891 dan 3871, keduanya menggunakan sumber daya yang hampir sama besarnya, namun PID dari kedua proses akan berbeda. Perintah $ renice 19 1891 berarti mengganti penjadwalan prioritas dari proses ke 19, dan NI berubah yang tadinya 0 menjadi 19.

6. Program top juga mempunyai fungsi yang sama dengan program renice. Pilih Jendela Top dan tekan r. Program top terdapat prompt PID to renice: tekan 3148 (ingat bahwa Anda harus mengganti 3148 dengan PID Anda sendiri) dan tekan Enter. Program top memberikan prompt Renice PID 3148 to value: tekan -19 dan tekan Enter.

 Hasil:



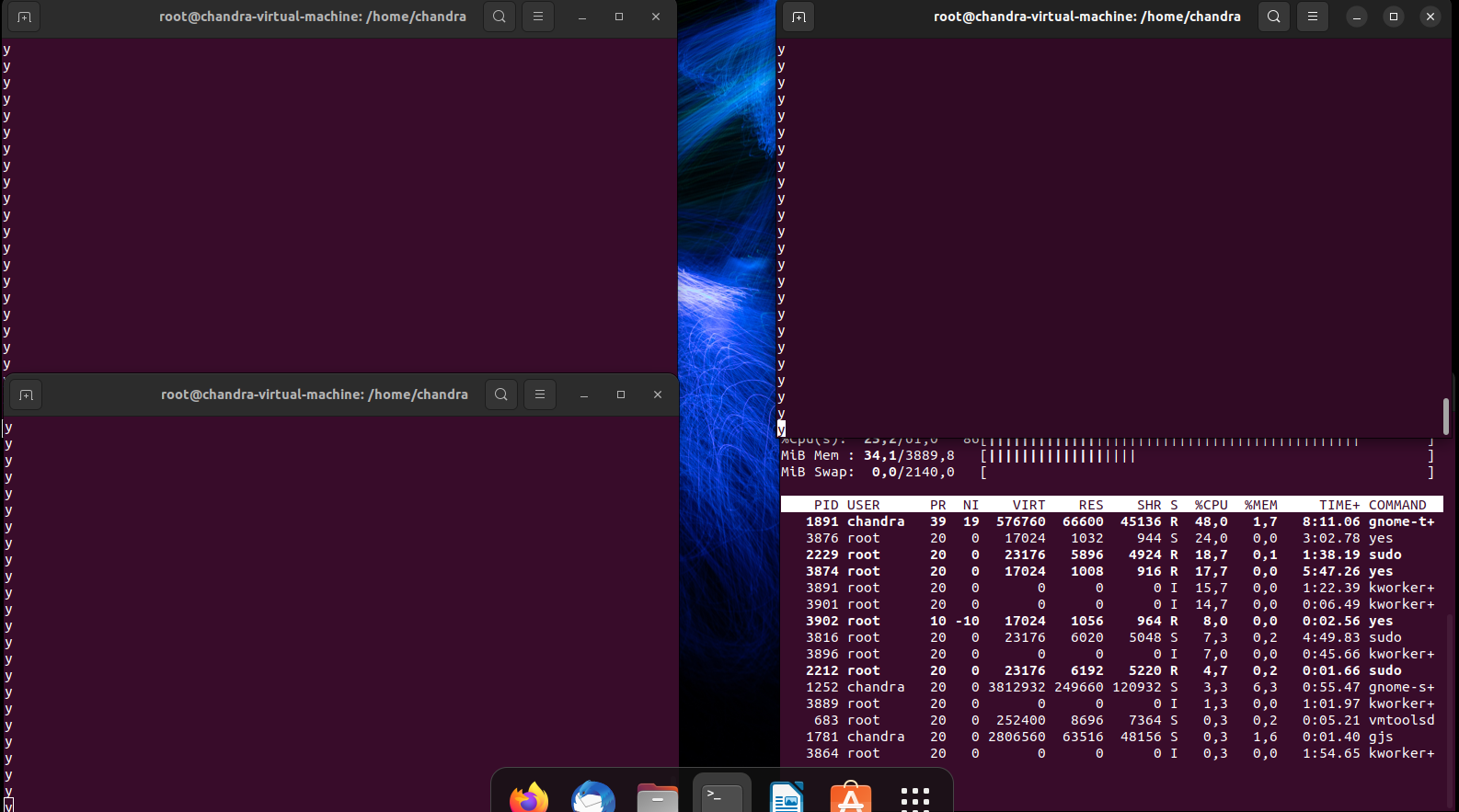


Analisis:

Setelah prioritasnya diubah menjadi 19, proses 3784 berjalan lebih lambat dari proses 3785. Ini disebabkan proses 3784 mendapatkan penjadwalan prioritas lebih besar dari proses 3785.

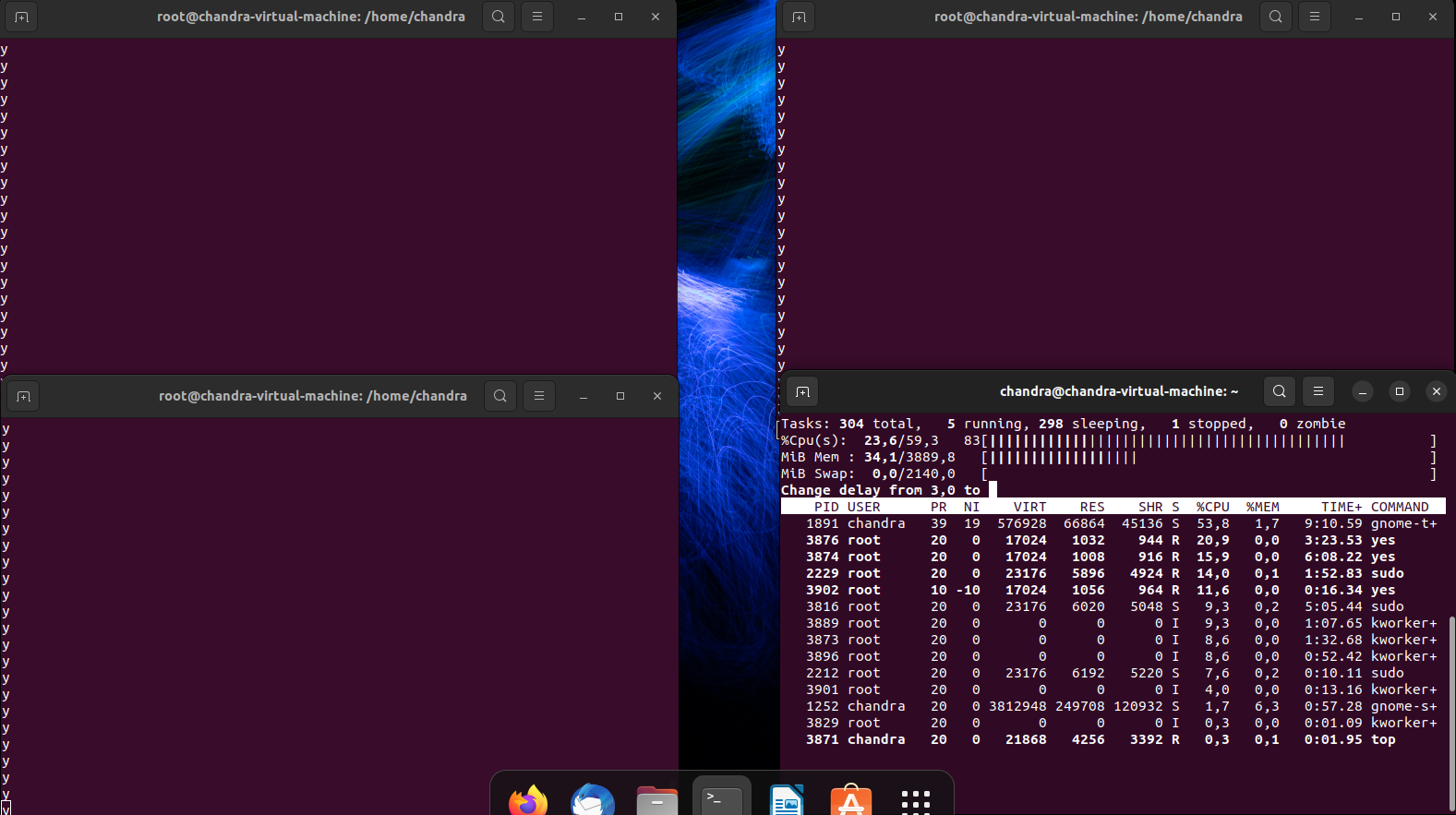
7. Pilih terminal 3 (yang sedang tidak menjalankan yes atau program top) dan ketik nice –n -10 yes dan tekan Enter. Tunggu beberapa saat agar program top berubah dan akan terlihat proses primes ketiga. Misalnya PID nya 4107. Opsi -10 berada pada kolom NI (penjadwalan prioritas).

Hasil :

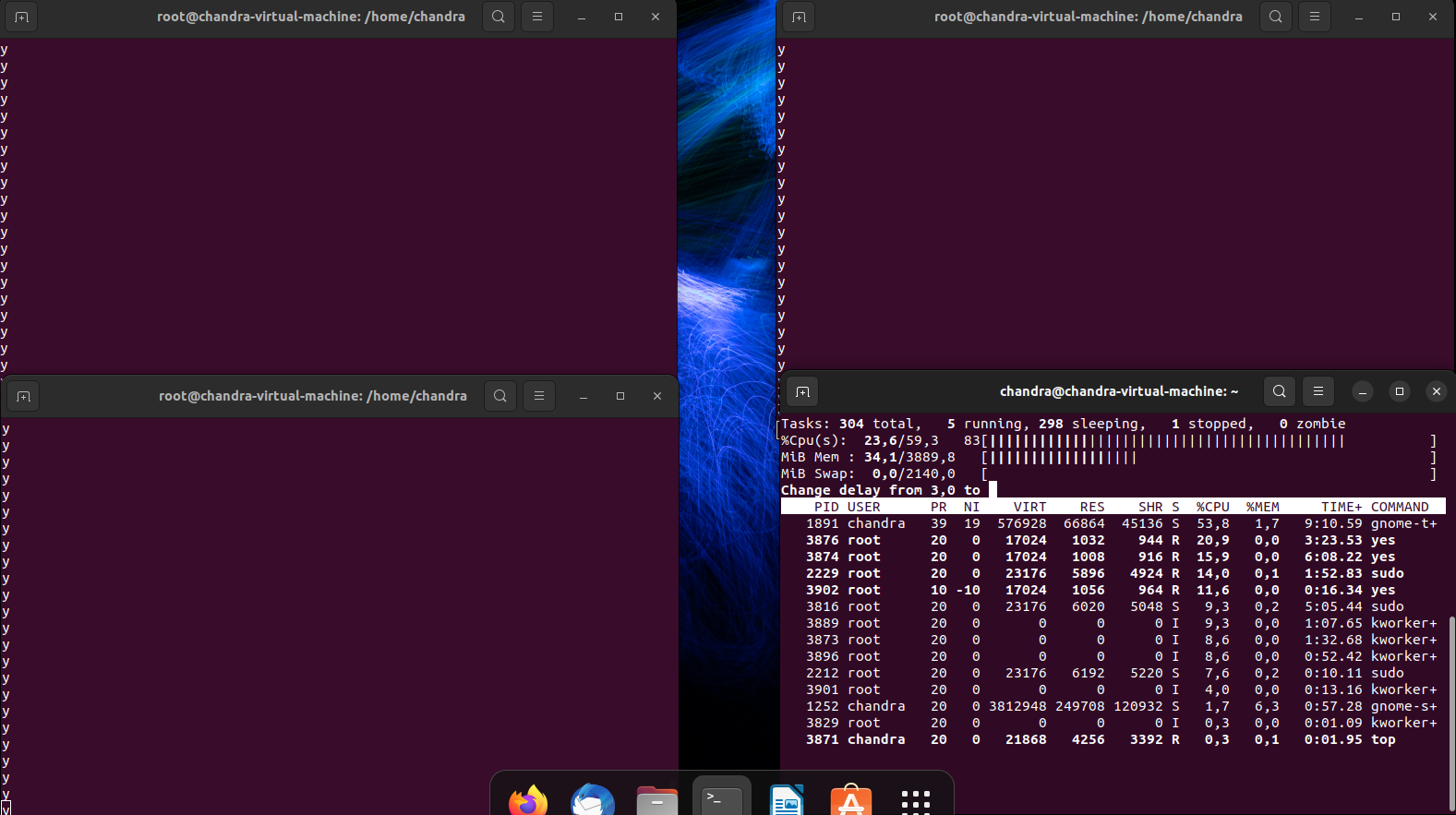
Analisis:

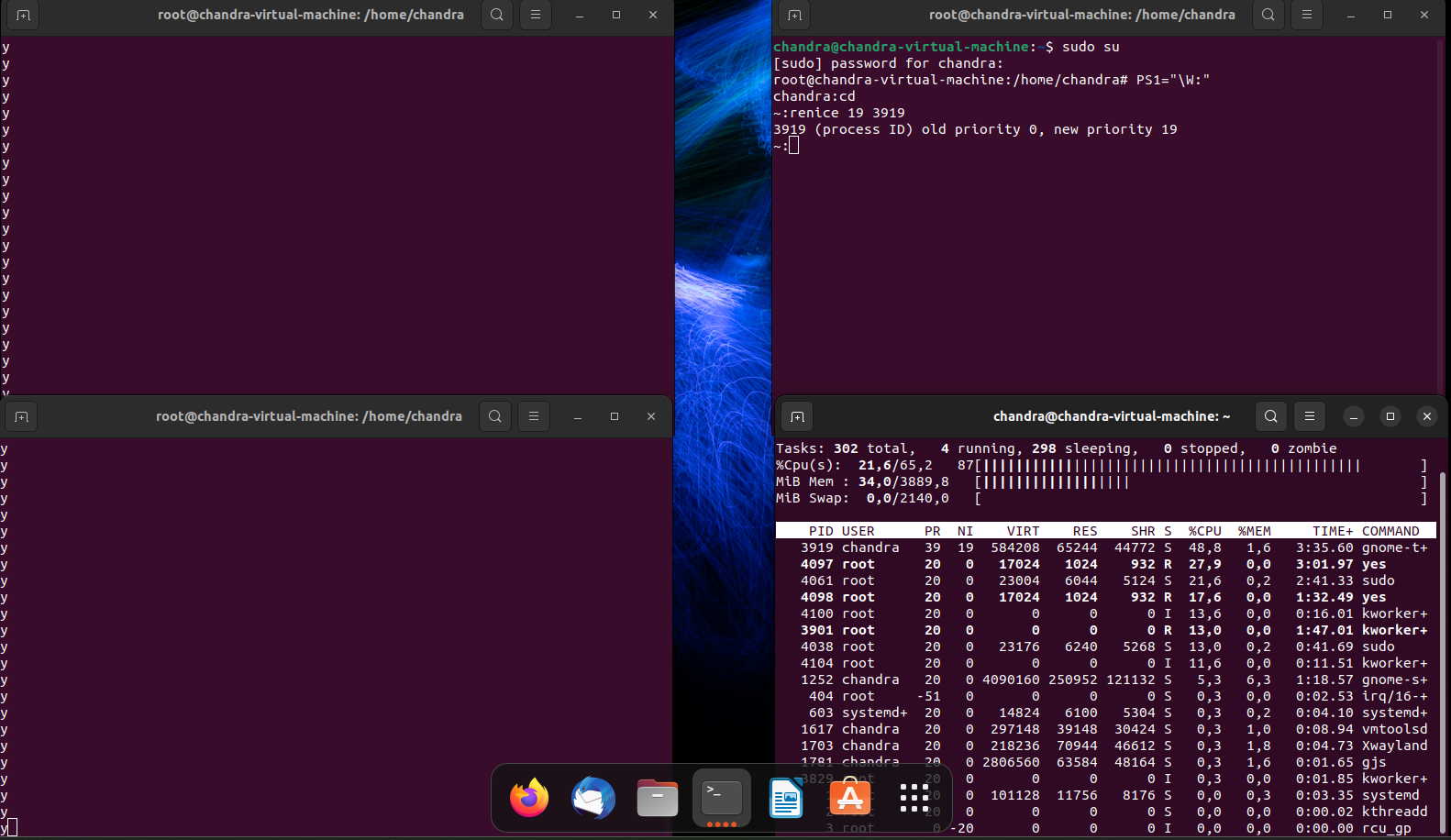
Perintah $ nice –n -10 ./a.out digunakan untuk membuat proses baru, dan opsi -10 merupakan penentuan prioritas dari proses tersebut. PID dari proses yang kita buat adalah 3902, dan disana terlihat pada kolom NI terdapat angka -10.

1. Jangan menggunakan mouse dan keyboard selama 10 detik. Program top menampilkan proses yang aktif selain program yes. Maka akan terlihat proses top terdaftar tetapi %CPU kecil (dibawah 1.0) dan konsisten. Juga terlihat proses berhubungan dengan dekstop grafis seperti X, panel dll.

Hasil :

1. Pindahkan mouse sehingga kursor berubah pada screen dan lihat apa yang terjadi dengan tampilan top. Proses tambahan akan muncul dan nilai %CPU berubah sebagai bagian grafis yang bekerja. Satu alasan adalah bahwa proses 4107 berjalan pada penjadwalan prioritas tinggi. Pilih jendela Top, ketik r. PID to renice: muncul prompt. Ketik 4107 (ubahlah 4107 dengan PID Anda) dan tekan Enter. Renice PID 4107 to value: muncul prompt. Ketik 0 dan tekan Enter. Sekarang pindahkan mouse ke sekeliling screen. Lihat perubahannya.

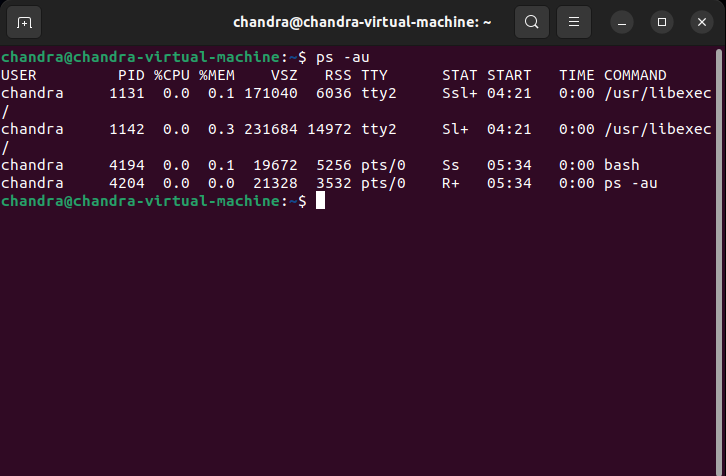
Hasil:

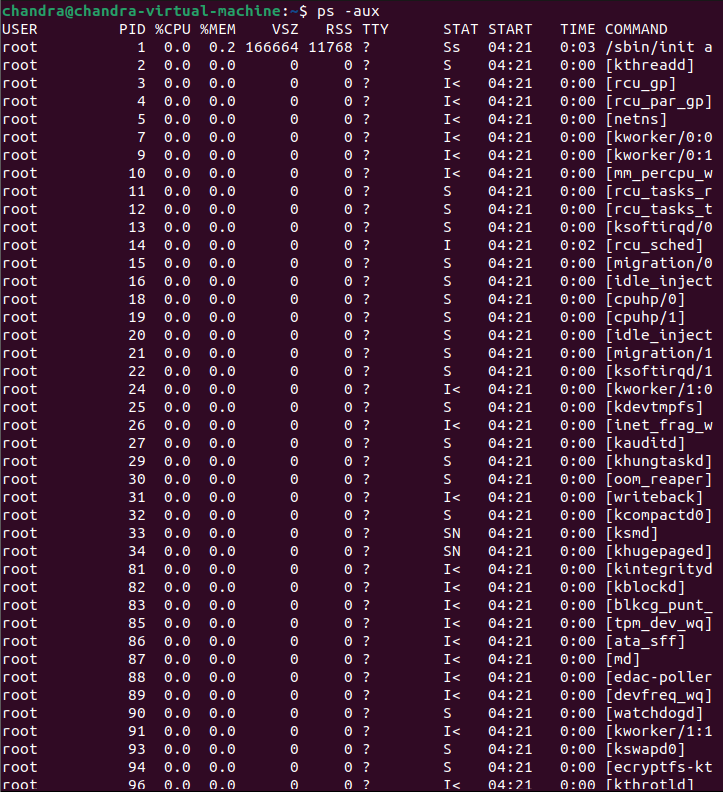
Analisa:

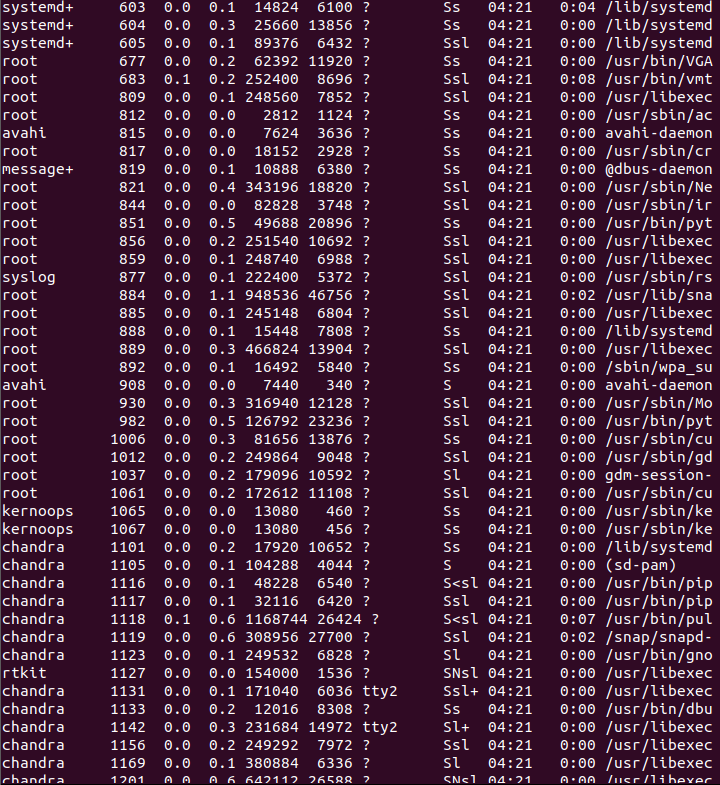
Saat kita memindahkan kursor mouse, beberapa proses yang muncul tadi penggunaan %CPU nya berubah semua yaitu cenderung bertambah. Dan juga muncul lagi proses lain dengan penggunaan %CPU lumayan banyak. Kita dapat mengubah konfigurasi suatu proses melalui perintah – perintah yang terdapat pada proses top. Dan ternyata setelah proses 3784 diubah prioritasnya menjadi 0, penggunaan sumber daya oleh terminal top menjadi lebih stabil (tidak banyak perubahan) walaupun mouse digerakkan ke sekeliling screen.

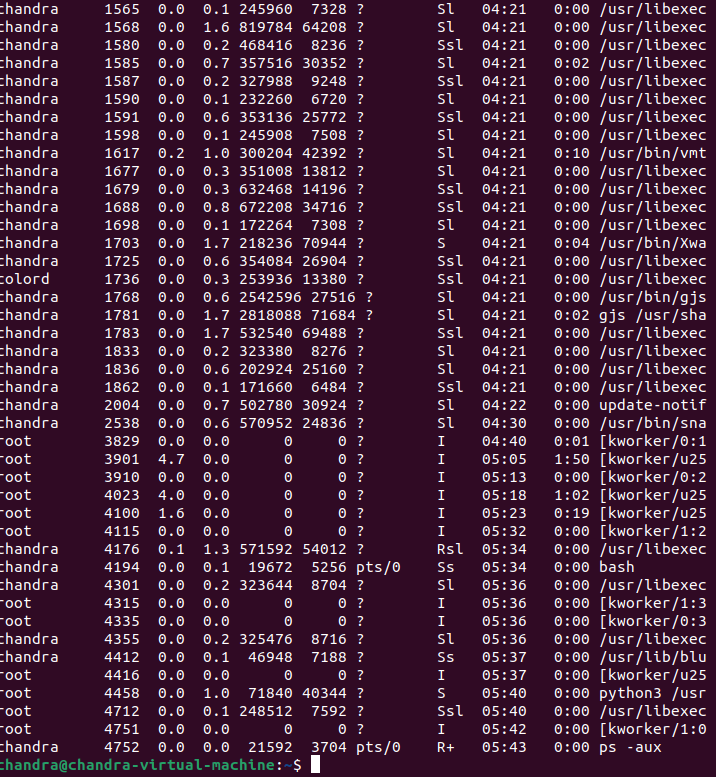
LATIHAN:

1. Masuk ke tty2 dengan Ctrl+Alt+F2. Ketik ps –au dan tekan Enter. Kemudian perhatikan keluaran sebagai berikut:

Hasil:







a. Sebutkan nama-nama proses yang bukan root

Jawab:

Untuk menampilkan proses dari seluruh user menggunakan perintah ps -aux untuk melihat nama proses yang bukan root. Berdasarkan gambar di atas, nama-nama proses yang bukan root adalah:

User = postfix, PID = 19160, cmd = pickup -l –

User = chandra, PID = 4196, cmd = bash

User = system+, PID = 603, cmd = /lib/system

User = syslog, PID = 1450, cmd = /usr/sbin/r

User = message+, PID = 819, cmd = @dbus-daemon

User = avahi,PID =910,cmd = avahi-daemon

b. Tulis PID dan COMMAND dari proses yang paling banyak menggunakan CPU time

Jawab:

User = chandra, PID = 4752, Command = ps -aux

c. Sebutkan buyut proses dan PID dari proses tersebut.

Jawab:

PID = 4355 dengan proses /usr/libexe.

d. Sebutkan beberapa proses daemon

User = message+, PID = 819, cmd = @dbus-daemon

User = avahi,PID =910,cmd = avahi-daemon

e. Pada prompt login lakukan hal- hal sebagai berikut:

$ csh

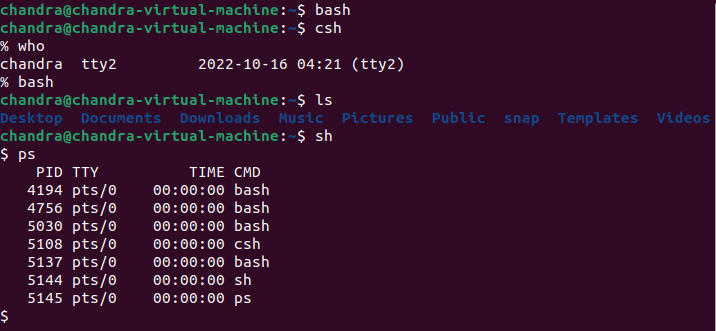
$ who

$ bash

$ ls

$ sh

$ ps

Hasil:

Analisa:

· Perintah $ csh: Shell interaktif yang memiliki sintaks yang lebih banyak dibandingkan bourne shell

· Perintah $ who: Untuk melihat user aktif yang login.

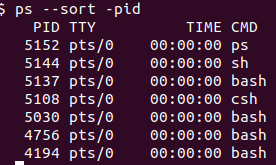
· Perintah $ bash: Shell interpreter bahasa sh untuk mengeksekusi perintah yang dibaca dari standart input atau dari sebuah file.

· Perintah $ ls: Untuk melihat nama file/direktori secara lengkap.

· Perintah $ sh: Bourne Shell atau shell (command interpreter) default dari unix.

· Perintah $ ps: Untuk menampilkan kondisi proses yang ada.

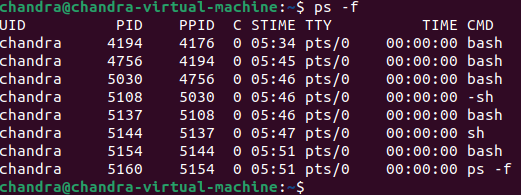
f. Sebutkan PID yang paling besar dan kemudian buat urut-urutan proses sampai ke PPID = 1.

Hasil:

Analisa: Untuk melihat urutan proses sampai ke PPID menggunakan perintah $ ps –sort -pid. Terlihat bahwa PID yang paling besar yaitu 20098.

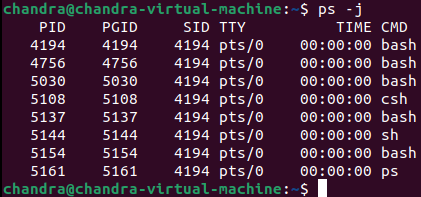
2. Cobalah format tampilan ps dengan opsi berikut dan perhatikan hasil tampilannya: ·

-f daftar penuh

Hasil:

Analisa: Perintah di atas digunakan untuk menampilkan dalam format penuh. seperti UID, PID, PPID, C, STIME, TTY, TIME, CMD.

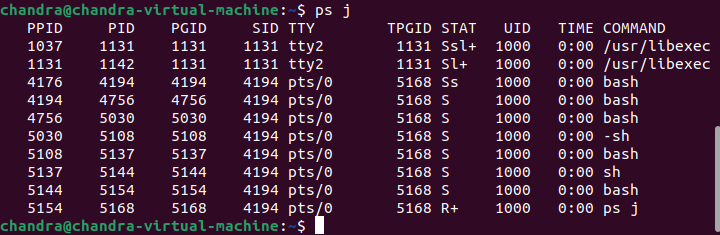
-j format job

Hasil:

Analisa: Perintah di atas untuk menampilkan dalam format job. Isinya dipenuhi oleh angka, karena cenderung menampilkan ID dimana proses sedang bekerja, berisi PID, PGID, SID, TTY, TIME, CMD.

j format job control

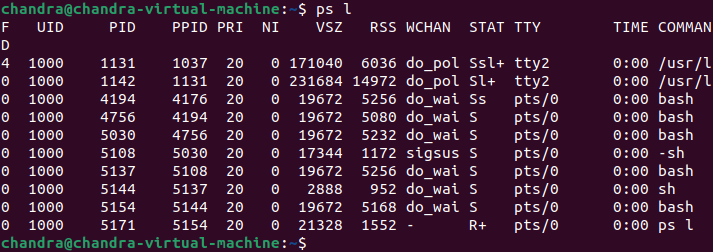
Hasil:



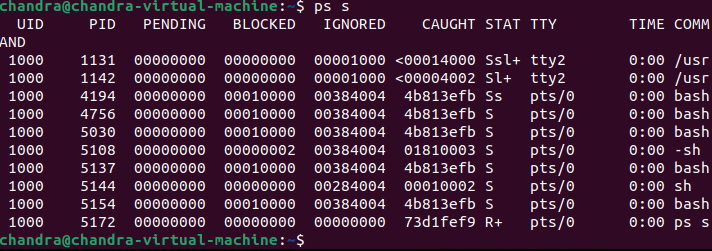
Analisa: Perintah di atas digunakan untuk menampilkan dengan format jobBSD. hasilnya berupa format job yang lebih detail, berisi PPID, PID, PGID, SID, TTY, TPGID, STAT, UID, TIME, COMMAND.

l daftar memanjang

Hasil:



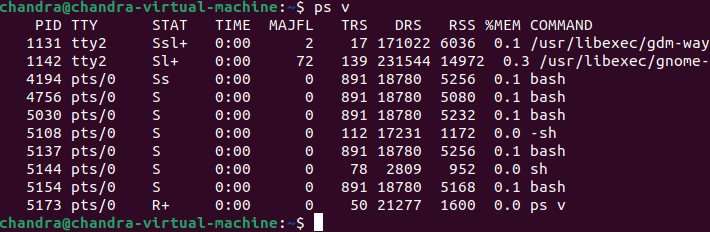
s format sinyal

Hasil:

Analisa: Perintah di atas menampilkan dengan format sinyal. (Biasanya angka yang muncul banyak), berisi UID, PID, PENDING, BLOCKED, IGNORED, CAUGHT, STAT, TTY, TIME, COMMAND.

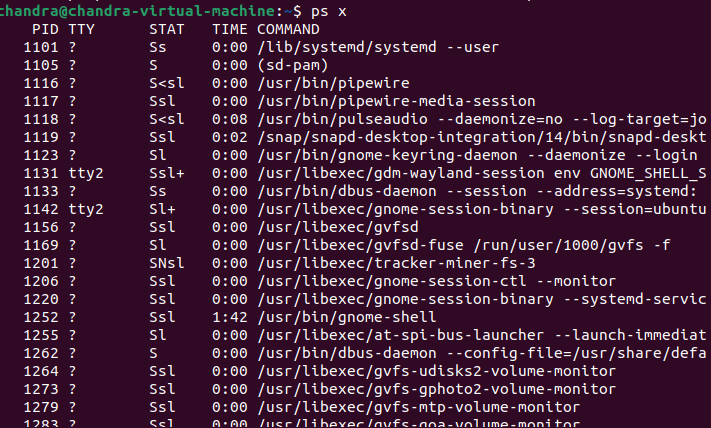
v format virtual memory

Hasil:



Analisa: Perintah di atas untuk menampilkan dengan format memori virtual. Berisi PID, TTY, STAT, TIME, MAJFL, TRS, DRS, RSS, %MEM, COMMAND.

X format register i386

Hasil:

3. Lakukan urutan pekerjaan berikut:

a. Gunakan perintah find ke seluruh direktory pada sistem, belokkan output sehingga daftar direktori dialihkan ke file directories.txt dan daftar pesan error dialihkan ke file errors.txt

Hasil:

Analisa: Perintah dilakukan dengan $ find > directories.txt > error.txt.

b. Gunakan perintah sleep 5. Apa yang terjadi dengan perintah ini?

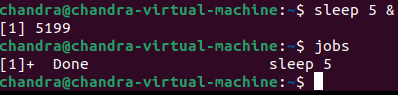
Hasil:

Analisa:

Proses Terminal terhenti selama 5 detik. Lalu baru bisa menulis command selanjutnya.

c. Jalankan perintah pada background menggunakan &

Hasil:

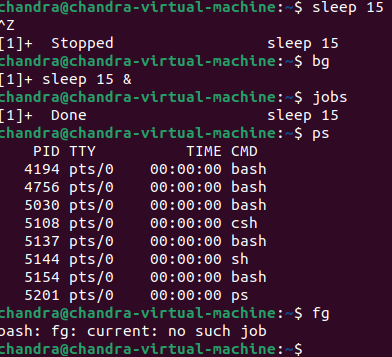


Analisa:

Karena proses hanya 5 detik, sehingga ketika kita lihat di jobs, maka statusnya akan selesai.

d. Jalankan sleep 15 pada foreground, hentikan sementara dengan Ctrl-Z dan kemudian letakkan pada background dengan bg. Ketikkan jobs. Ketikkan ps. Kembalikan job ke foreground dengan perintah fg.

Hasil:

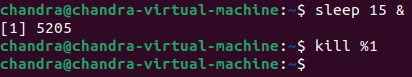


Analisa:

Pertama-tama, perintah sleep 15 dieksekusi, lalu distop sebelum terminal kembali dapat menulis command berikutnya. Setelah itu, lihat di background dengan perintah $ bg. Alasan disini pada saat jobs diketikan tidak keluar apa apa adalah karena proses sleep 15 telah selesai, jadi sudah tidak ada lagi jobs atau pekerjaan yang masih berjalan. Kita juga bisa memastikannya melalui ps dan tidak ada proses sleep yang berjalan, maka dari itu pada saat kita melakukan perintah fg terdapat output “no such job” yang berarti tidak ada pekerjaan atau job.

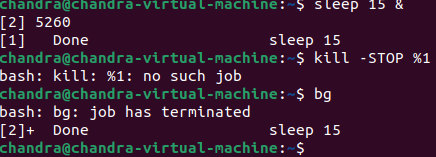
e. Jalankan sleep 15 pada background menggunakan & dan kemudian gunakan perintah kill untuk menghentikan proses diikuti job number.

Hasil:



Analisa: Proses dapat dihentikan menggunakan kill.

f. Jalankan sleep 15 pada background menggunakan & dan kemudian gunakan kill untuk menghentikan sementara proses. Gunakan bg untuk melanjutkan menjalankan proses.

Hasil:

Analisa: Proses dapat dihentikan menggunakan kill. Option -STOP digunakan untuk menstop proses sleep 15. Jika tidak menggunakan opsi tersebut, maka proses sleep akan terminated/berhenti sehingga proses sleep tidak dapat dilanjutkan.

g. Jalankan sleep 60 pada background 5 kali dan terminasi semua pada dengan menggunakan perintah killall.

Hasil:

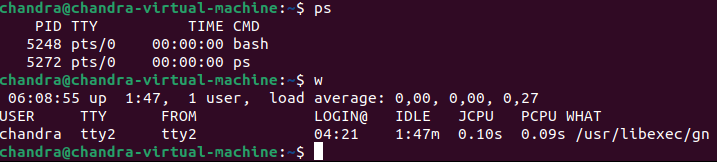
Analisa:

Perintah $ for i in {1..5} do sleep 60 & done adalah perintah untuk mengulang perintah sleep 60 sebanyak 5 kali. Perintah killall dapat digunakan untuk menghapus atau stop permanen proses dengan menambahkan nama proses setelah killall, sehingga semua proses yang ada nama tersebut akan dihentikan.

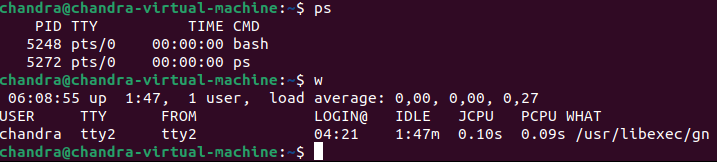
h. Gunakan perintah ps, w dan top untuk menunjukkan semua proses yang sedang dieksekusi.

Hasil:

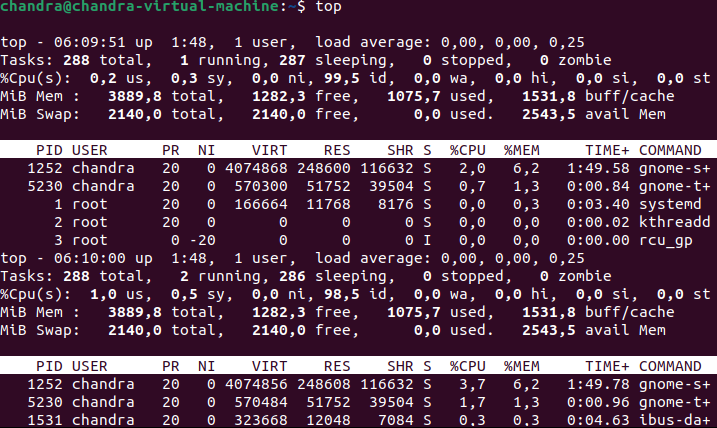
Ps



W



Top

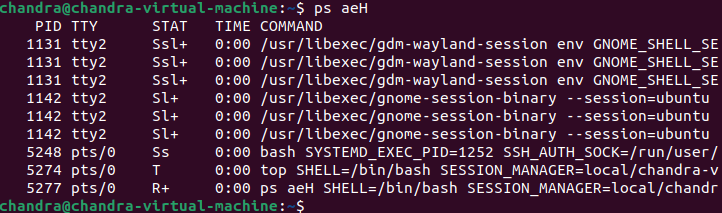


Analisa:

Perintah ps digunakan untuk melihat kondisi proses yang ada. Perintah w digunakan untuk menampilkan proses secara lengkap (running, done, terminated). Perintah top adalah tool baris perintah, tetapi perintah ini memberikan output interaktif dari semua proses yang sedang berjalan, menampilkan informasi dalam interface grafis. Output dari perintah top memberi kita banyak informasi, termasuk penggunaan CPU dan RAM secara keseluruhan pada sistem.

i. Gunakan perintah ps –aeH untuk menampilkan hierarki proses. Carilah init proses. Apakah Anda bisa identifikasi sistem daemon yang penting? Dapatkan Anda identifikasi shell dan subproses?

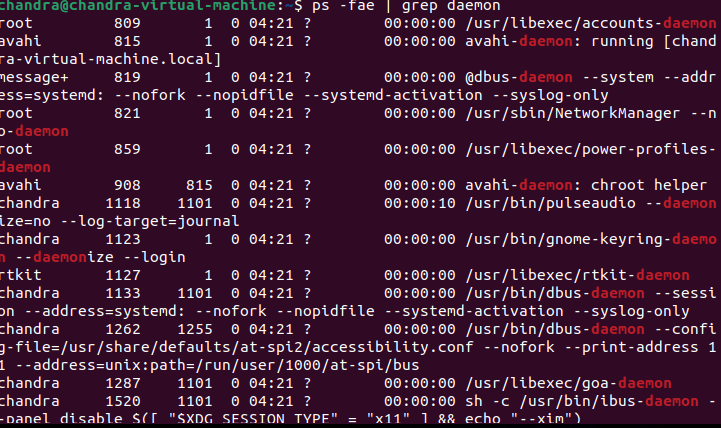
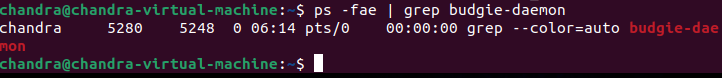
Hasil:



Analisa:

Init proses disini adalah pada system dengan PID = 1. Terdapat beberapa sistem daemon yang penting yaitu system, rsyslogd, cryptd, cupsd. Pada gambar akhir, bisa dilihat bahwa terdapat subproses dari gnome terminal.

j. Kombinasikan ps –fae dan grep, apa yang Anda lihat?

Hasil:

Ps -aeH hanya mengarahkan pada proses yang dimasukan setelah grep, contohnya ps -aeH | grep budgie daemon, berarti ps –aeH hanya mengarahkan pada semua proses yang bernama budgie daemon, dan pada tampilannya akan berwarna merah. Begitupula pada ps -aeH | grep daemon. Semua proses yang mengandung daemon, maka tampilannya akan berwarna merah.

k. Jalankan proses sleep 300 pada background. Log off komputer dan log in kembali. Lihat daftar semua proses yang berjalan. Apa yang terjadi pada proses sleep?

Hasil:



Analisa:

Pada log off menggunakan GUI, proses sleep masih berjalan selama waktunya belum habis. Sedangkan pada log off secara GUI, proses sleep tidak berjalan lagi walaupun waktunya belum habis.

KESIMPULAN:

Berdasarkan hasil percobaan yang telah dilakukan dapat disimpulkan bahwa proses adalah program yang sedang berjalan atau running. Setiap kali Anda menggunakan utilitas sistem atau program aplikasi dari shell, satu atau lebih proses "child" dibuat oleh shell berdasarkan perintah yang diberikan. Setiap kali perintah diberikan ke shell Linux, kernel menghasilkan ID proses. Adanya suatu proses ditunjukkan dengan adanya informasi berupa PID yang menampilkan pengenal proses, TTY digunakan untuk menampilkan nama terminal tempat proses tersebut aktif, TIME berfungsi untuk menampilkan waktu yang dibutuhkan untuk counter dan Perintah CMD (Command) digunakan untuk menampilkan instruksi/perintah yang digunakan. Ada perintah yang berbeda untuk menjalankan proses dan untuk menghentikan proses Anda dapat menggunakan perintah kill. Proses juga dapat dibuat dan ditampilkan dalam tampilan pohon atau hierarki.