HARDWARE DAN PENYIMPANAN DATA

3

OBJEKTIF :

1. Mahasiswa Mampu Mengetahui Komponen Hardware dan Bagaimana Proses Penyimpanan Data Pada Linux.
2. Mahasiswa Mampu Untuk Memahami Fungsi Hardware dan Perintah Dasar Dalam Penyimpanan data Pada Linux.

---

1. PENDAHULUAN HARDWARE

Komputer dimulai sebagai perangkat *hardware*, namun saat ini terus berkembang seiring dengan adanya mesin virtual/*virtual mechine(VMs)* yang seluruhnya dibangun dari *software* dan ditujukan sebagai mesin jarak jauh yang dapat diakses pengguna melalui protokol khusus, maupun secara remote.

Dengan mengingat mesin virtual, sangat penting untuk memahami apa yang membentuk fisik *hardware* yang membentuk sebuah komputer. Tanpa pemahaman tentang komponen apa yang membentuk komputer, bagaimana mereka terintegrasi dan berkomunikasi antara satu sama lain, dan bagaimana komputer seharusnya berfungsi, Anda tidak dapat menginstal, mengkonfigurasi, mengatur, mengamankan atau memecahkan masalah sistem dengan benar.

Ada ratusan jenis komputer tertentu dalam layanan, dari dekoder ke firewall khusus hingga laptop, server, dan banyak perangkat khusus, tetapi semuanya menampilkan setidaknya beberapa komponen dasar yang sama yang sebenarnya menjadikannya komputer. Bab ini memperkenalkan dan menjelaskan dasar-dasar yang membentuk komputer dan mengembangkan beberapa item opsional yang cukup umum.

2. MOTHERBOARD

Motherboard, atau sistem board, adalah papan *hardware* utama di komputer yang melaluinya *central processing unit (CPU)*, *random-access memory (RAM)* dan komponen lainnya semuanya terhubung. Bergantung pada jenis komputer (laptop, desktop, server), beberapa perangkat, seperti prosesor dan RAM, dipasang langsung ke motherboard, sementara perangkat lain, seperti add-on cards, terhubung melalui bus.

3. PROSESOR

Central processing unit (CPU atau prosesor) adalah salah satu komponen *hardware* paling penting dari sebuah komputer. Prosesor adalah otak komputer, tempat eksekusi kode berlangsung dan tempat sebagian besar kalkulasi dilakukan. Ini terhubung langsung (disolder) ke motherboard, karena motherboard biasanya dikonfigurasi untuk bekerja dengan jenis prosesor tertentu.

Jika sistem *hardware* memiliki lebih dari satu prosesor, sistem tersebut disebut multiprosesor. Jika lebih dari satu prosesor digabungkan menjadi satu chip prosesor keseluruhan, maka disebut multi-core.

Meskipun dukungan tersedia untuk lebih banyak jenis prosesor di Linux daripada sistem operasi lain, pada dasarnya hanya ada dua jenis prosesor yang digunakan di komputer desktop dan server: x86 dan x86_64. Pada sistem x86, sistem memproses data 32 bit dalam satu waktu; pada x86_64 sistem memproses data 64 bit dalam satu waktu. Sistem x86_64 juga mampu memproses data 32 bit sekaligus dalam mode kompatibel ke belakang. Salah satu keuntungan utama dari sistem 64-bit adalah kemampuannya untuk bekerja dengan lebih banyak memori, sementara keuntungan lainnya termasuk peningkatan efisiensi pemrosesan dan peningkatan keamanan.

Intel membuat keluarga prosesor x86 pada tahun 1978 dengan merilis prosesor 8086. Sejak saat itu, Intel telah menghasilkan banyak prosesor lain yang merupakan penyempurnaan dari 8086 asli; mereka dikenal secara umum sebagai

prosesor x86. Prosesor ini termasuk seri 80386 (i386), 80486 (i486), seri Pentium (i586) dan seri Pentium Pro (i686). Selain Intel, perusahaan lain seperti AMD dan Cyrix juga telah memproduksi prosesor yang kompatibel dengan x86. Sementara Linux mampu mendukung prosesor kembali ke generasi i386, banyak distribusi (terutama yang menampilkan dukungan korporat, seperti SUSE, Red Hat dan Canonical) membatasi dukungannya ke i686 atau yang lebih baru.

Keluarga prosesor x86_64, termasuk prosesor 64-bit dari Intel dan AMD, telah diproduksi sejak sekitar tahun 2000. Hasilnya, hampir semua prosesor modern yang dikirim saat ini adalah tipe x86_64. Sementara *hardware* telah tersedia selama lebih dari satu dekade sekarang *software* untuk mendukung keluarga prosesor ini berkembang jauh lebih lambat. Meskipun ini adalah tahun 2018 saat pembaruan ini, masih ada sejumlah paket yang masih tersedia untuk arsitektur x86.

Untuk melihat arsitektur CPU yang saat ini digunakan, gunakan *command* `arch` :

```
sysadmin@localhost:~$ arch
x86_64
```

Untuk informasi lebih lanjut tentang CPU, gunakan *command* `lscpu` :

```
sysadmin@localhost:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Thread(s) per core:    1
Core(s) per socket:    4
Socket(s):             1
NUMA node(s):          1
```

```
Vendor ID:              GenuineIntel
```

Baris pertama dari keluaran ini menunjukkan bahwa CPU digunakan dalam mode 64-bit, karena arsitektur yang dilaporkan adalah x86_64. Baris keluaran kedua menunjukkan bahwa CPU mampu beroperasi dalam mode 32 atau 64-bit, oleh karena itu, ini sebenarnya adalah CPU 64-bit.

4. RANDOM ACCESS MEMORY

Motherboard biasanya memiliki slot di mana *random-access memory (RAM)* dapat dihubungkan ke sistem. Sistem arsitektur 32-bit dapat menggunakan hingga 4 gigabyte (GB) RAM, sementara arsitektur 64-bit mampu menangani dan menggunakan lebih banyak RAM.

Dalam beberapa kasus, RAM yang dimiliki sistem mungkin tidak cukup untuk menangani semua persyaratan sistem operasi. Setelah dipanggil, program dimuat dalam RAM bersama dengan data apa pun yang perlu disimpan, sementara instruksi dikirim ke prosesor saat dijalankan.

Ketika sistem kehabisan RAM, memori virtual yang disebut *swap sapce* digunakan untuk menyimpan data sementara ke disk. Data yang disimpan dalam RAM yang belum digunakan baru-baru ini dipindahkan ke bagian hard drive yang ditetapkan sebagai swap, membebaskan lebih banyak RAM untuk digunakan oleh program yang sedang aktif. Jika perlu, data yang ditukar ini dapat dipindahkan kembali ke RAM di lain waktu. Sistem melakukan semua ini secara otomatis selama swap dikonfigurasi.

Untuk melihat jumlah RAM di sistem Anda, termasuk *swap space*, jalankan *command* `free` . *Command* `free` memiliki *option* `-m` untuk memaksa keluaran dibulatkan ke megabyte (MB) terdekat dan *option* `-g` untuk memaksa keluaran dibulatkan ke gigabyte (GB) terdekat:

```
sysadmin@localhost:~$ free -m
             total       used       free     shared    buffers     cached
Mem:          1894        356       1537          0         25        177
-/+ buffers/cache:        153       1741
Swap:         4063          0       4063
```

Outputnya menunjukkan bahwa sistem ini memiliki total 1.894 MB dan saat ini menggunakan 356 MB.

Jumlah swap tampaknya sekitar 4 GB, meskipun tampaknya tidak ada yang digunakan. Ini masuk akal karena begitu banyak RAM fisik yang *free*, jadi saat ini tidak perlu RAM virtual (*swap space*) untuk digunakan.

5. BUSES

Bus adalah koneksi berkecepatan tinggi yang memungkinkan komunikasi antar komputer atau komponen di dalam komputer. Motherboard memiliki bus yang memungkinkan beberapa perangkat untuk terhubung ke sistem, termasuk *Peripheral Component Interconnect (PCI)* dan *Universal Serial Bus (USB)*. Motherboard ini juga memiliki konektor untuk monitor, keyboard, dan mouse.

Jenis sistem yang berbeda akan menggunakan bus secara berbeda untuk menghubungkan komponen. Di komputer desktop atau server, ada motherboard dengan prosesor, RAM, dan komponen lain yang langsung terpasang, dan kemudian bus berkecepatan tinggi yang memungkinkan komponen tambahan dipasang melalui slot kartu, seperti video, jaringan, dan perangkat periferal lainnya.

Semakin banyak untuk laptop dan komputer faktor bentuk ringan / tipis / kecil, sebagian besar komponen komputer dapat langsung dihubungkan ke motherboard, termasuk prosesor biasa, RAM, serta komponen tambahan seperti kartu jaringan. Dalam konfigurasi ini, bus masih ada, tetapi secara efektif salah

satu faktor kenyamanan utama untuk dapat menukar atau meningkatkan perangkat tidak lagi dimungkinkan.

- **Perangkat periferal / *Peripheral Devices***

  Perangkat periferal adalah komponen yang terhubung ke komputer yang memungkinkan penyimpanan input, output, atau data. Keyboard, mouse, monitor, printer, dan hard disk semuanya dianggap periferal dan diakses oleh sistem untuk melakukan fungsi di luar pemrosesan pusat. Untuk melihat semua perangkat yang terhubung dengan bus PCI, pengguna dapat menjalankan *command* `lspci` . Berikut ini adalah contoh keluaran dari *command* ini. Bagian yang disorot menunjukkan bahwa sistem ini memiliki pengontrol VGA (konektor monitor), pengontrol penyimpanan SCSI (jenis hard drive) dan pengontrol Ethernet (konektor jaringan):

Catatan: Contoh di bawah ini menggunakan *command* `lspci`. *Command* ini tidak tersedia dalam lingkungan mesin virtual kursus ini.

```
sysadmin@localhost:~$ lspci

00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX
/DX Host bridge (rev 01)

00:01.0 PCI bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/
DX AGP bridge (rev 01)

00:07.0 ISA bridge: Intel Corporation 82371AB/EB/MB PIIX4 ISA (
rev 08)

00:07.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 ID
E (rev 01)

00:07.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev
08)

00:07.7 System peripheral: VMware Virtual Machine Communication
Interface (rev 10)

00:0f.0 VGA compatible controller: VMware SVGA II Adapter
```

```
03:00.0 Serial Attached SCSI controller: VMware PVSCSI SCSI Con
troller (rev 02
0b:00.0 Ethernet controller: VMware VMXNET3 Ethernet Controller
(rev 01)
```

- **Perangkat Universal Serial Bus / *Universal Serial Bus Devices***

  Meskipun bus PCI digunakan untuk banyak perangkat internal seperti kartu suara dan jaringan, perangkat eksternal (atau periferal) yang terhubung ke komputer melalui USB jumlahnya terus meningkat.

  Perangkat yang terhubung secara internal biasanya *cold-plug*, yang berarti sistem harus dimatikan untuk menghubungkan atau memutuskan perangkat. Perangkat USB adalah *hot-plug*, artinya mereka dapat dihubungkan atau diputuskan saat sistem sedang berjalan.

  Meskipun perangkat USB dirancang dengan hot-plug, penting untuk memastikan bahwa semua sistem file yang terpasang telah dilepas dengan benar, atau kehilangan data dan kerusakan sistem file dapat terjadi.

  Untuk menampilkan perangkat yang terhubung ke sistem melalui USB, jalankan *command* `lsusb` :

Catatan : Contohnya menggunakan *command* `lsusb`. *Command* ini tidak tersedia dalam lingkungan mesin virtual kursus ini.

```
sysadmin@localhost:~$ lsusb
Bus 001 Device 002: ID 0e0f:000b VMware, Inc.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 004: ID 0e0f:0008 VMware, Inc.
Bus 002 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 002 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
```

```
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

6. HARD DRIVE

Hard drive, juga disebut hard disk, perangkat disk, atau perangkat penyimpanan, dapat dipasang ke sistem dengan beberapa cara; pengontrol dapat diintegrasikan ke motherboard, pada kartu PCI, atau sebagai perangkat USB. Untuk keperluan sebagian besar sistem Linux, hard drive umumnya dapat didefinisikan sebagai perangkat penyimpanan elektromekanis atau elektronik yang menyimpan data untuk diakses oleh sistem.

Hard drive dibagi menjadi satu atau lebih partisi. Partisi adalah divisi logis dari hard drive, dirancang untuk mengambil sejumlah besar ruang penyimpanan yang tersedia dan memecahnya menjadi area yang lebih kecil. Meskipun pada Microsoft Windows biasanya memiliki satu partisi untuk setiap hard drive, pada distribusi Linux, banyak partisi per hard drive adalah hal yang umum.

Beberapa hard drive menggunakan teknologi partisi yang disebut *Master Boot Record (MBR)* sementara yang lain menggunakan jenis partisi yang disebut *GUID Partitioning Table (GPT)*. Jenis partisi MBR telah digunakan sejak masa awal Personal Computer (PC), dan jenis GPT telah tersedia sejak tahun 2000.

Istilah lama yang digunakan untuk menggambarkan hard disk internal adalah disk tetap/*fixed disk*, sebagai disk tetap (tidak dapat dilepas). Istilah ini memunculkan beberapa nama *command*: *command* `fdisk`, `cfdisk` dan `sfdisk`, yang merupakan *tool* untuk bekerja dengan disk yang dipartisi MBR.

Disk GPT menggunakan jenis partisi yang lebih baru, yang memungkinkan pengguna membagi disk menjadi lebih banyak partisi daripada yang didukung MBR. GPT juga memungkinkan memiliki partisi yang bisa lebih besar dari dua terabyte (MBR tidak). *Tool* untuk mengelola disk GPT diberi nama yang mirip dengan rekan `fdisk` nya: `gdisk`, `cgdisk`, dan `sgdisk`.

Ada juga sekumpulan *tools* yang mendukung disk jenis MBR dan GPT. Set *tools* ini mencakup *command* `parted` dan *tool* grafis `gparted`.

Catatan : Banyak distribusi Linux menggunakan *tool* `gparted` dalam rutinitas penginstalan, karena *tool* ini menyediakan antarmuka grafis ke opsi canggih yang tersedia untuk membuat partisi baru, mengubah ukuran yang sudah ada (seperti partisi Windows) dan banyak tugas lanjutan lainnya.

Hard drive dikaitkan dengan nama file (disebut file perangkat) yang disimpan di direktori /dev. Setiap nama file perangkat terdiri dari beberapa bagian.

- **Jenis File / *File Type***

  Nama file diawali berdasarkan jenis hard drive yang berbeda. Hard drive *IDE (Intelligent Drive Electronics)* dimulai dengan hd, sedangkan hard drive USB, *SATA (Serial Advanced Technology Attachment)* dan *SCSI (Small Computer System Interface)* dimulai dengan sd.

- **Pesanan Perangkat / *Device Order***

  Setiap hard drive diberi huruf yang mengikuti awalan. Misalnya, hard drive IDE pertama akan diberi nama /dev/hda dan yang kedua akan menjadi /dev/hdb, dan seterusnya.

- **Partisi / *Partition***

  Akhirnya, setiap partisi pada disk diberi indikator numerik yang unik. Misalnya, jika hard drive USB memiliki dua partisi, partisi tersebut dapat dikaitkan dengan file perangkat /dev/sda1 dan /dev/sda2.

  Contoh berikut menunjukkan sistem yang memiliki tiga perangkat sd: /dev/sda, /dev/sdb dan /dev/sdc . Selain itu, ada dua partisi di perangkat pertama, sebagaimana dibuktikan dengan file /dev/sda1 dan /dev/sda2 , dan satu partisi di perangkat kedua, sebagaimana dibuktikan dengan file /dev/sdb1 :

```
root@localhost:~$ ls /dev/sd*
/dev/sda /dev/sda1  /dev/sda2  /dev/sdb  /dev/sdb1  /dev/sdc
```

Command `fdisk` dapat digunakan untuk menampilkan informasi lebih lanjut tentang partisi :

Catatan: *command* berikut membutuhkan akses root.

```
root@localhost:~$ fdisk -l /dev/sda

Disk /dev/sda: 21.5 GB, 21474836480 bytes

255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sec
tors

Units = sectors of 1 * 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x000571a2



   Device Boot      Start         End      Blocks   Id System

/dev/sda1   *        2048    39845887    19921920   83 Linux

/dev/sda2        39847934    41940991     1046529    5 Extended

/dev/sda5        39847936    41940991     1046528   82 Linux swap
                                                     / Solaris
```

Catatan: Membuat dan memodifikasi partisi berada di luar cakupan kursus ini. Untuk informasi lebih lanjut, lihat NDG Linux I.

7. SOLID STATE DISK

Hard disk biasanya dianggap mencakup perangkat *spinning disk* tradisional, istilah ini juga dapat merujuk ke solid state drive atau SSD. SSD adalah disk yang lebih baru dan sangat berbeda.

Pertimbangkan perbedaan antara hard disk spinning platter tradisional dengan thumb drive USB. Hard disk spinning plater secara harfiah memiliki piringan piringan yang berputar di dalamnya yang dibaca oleh kepala penggerak, dan piringan yang berputar disusun untuk memanfaatkan sifat piranti yang berputar. Data ditulis (dan dibaca) dalam string panjang blok berurutan yang ditemui kepala drive saat platter berputar.

Solid state disk pada dasarnya adalah thumb drive USB yang berkapasitas lebih besar dalam aspek konstruksi dan fungsi. Karena tidak ada bagian yang bergerak, dan tidak ada disk yang berputar, hanya lokasi memori yang akan dibaca oleh pengontrol, solid state disk terlihat dan terukur lebih cepat dalam mengakses informasi yang disimpan dalam chip memorinya.

Solid state disk dikontrol oleh prosesor yang tertanam atau onboard yang membuat keputusan tentang di mana dan bagaimana data ditulis dan dibaca kembali dari chip memori saat diminta.

Keuntungan dari solid state disk adalah penggunaan daya yang lebih rendah, penghematan waktu dalam boot sistem, pemuatan program yang lebih cepat, tidak panas karna tidak ada getaran pada SSD. Kekurangannya adalah biaya yang lebih tinggi dibandingkan dengan hard disk yang berputar, kapasitas yang lebih rendah karena biaya yang lebih tinggi, dan jika disolder langsung pada motherboard / mainboard, tidak ada kemampuan untuk mengupgrade dengan menukar drive.

Catatan : Agar materi ini dapat dipahami, kami akan merujuk ke semua jenis disk, spinning atau solid state, sebagai hard disk, kecuali jika ada perbedaan penting yang mengharuskan kami untuk membedakan keduanya.

8. OPTICAL DRIVES

Drive optik, sering disebut sebagai CD-ROM, DVD, atau Blu-Ray, adalah media penyimpanan yang dapat dilepas. Sementara beberapa perangkat yang

digunakan dengan disk optik bersifat hanya-baca, perangkat lain dapat *burning* (menulis ke) disk, saat menggunakan jenis disk yang dapat ditulis. Ada berbagai standar untuk disk yang dapat ditulis dan ditulis ulang, seperti CD-R, CD + R, DVD + RW, dan DVD-RW.

Di mana disk yang dapat dilepas ini dipasang di sistem file merupakan pertimbangan penting bagi administrator Linux. Distribusi modern sering kali memasang disk di bawah folder / media, sedangkan distribusi yang lebih lama biasanya memasangnya di bawah folder / mnt. Misalnya, thumb drive USB mungkin dipasang di jalur / media / usbthumb.

Setelah pemasangan, sebagian besar *interface* GUI meminta pengguna untuk melakukan beberapa tindakan, seperti membuka konten disk di browser file atau memulai program media. Ketika pengguna selesai menggunakan disk, disk perlu dilepas menggunakan menu atau perintah `unmount`.

9. MANAJEMEN PERANGKAT

Linux secara alami memiliki sejumlah distribusi atau versi, sebagian besar difokuskan pada perangkat keras utama, sementara yang lain berjalan pada jenis atau model perangkat keras yang relatif tidak jelas. Dapat dikatakan bahwa ada distribusi Linux yang dirancang khusus untuk hampir semua platform perangkat keras modern, dan juga untuk banyak platform historis.

Masing-masing platform perangkat keras ini memiliki banyak variasi dalam komponen yang tersedia. Selain beberapa jenis hard drive, terdapat banyak grafis *card*, monitor, dan printer yang berbeda. Dengan popularitas perangkat USB, perangkat seperti penyimpanan USB, kamera, dan ponsel, jumlah perangkat yang tersedia yang ingin disambungkan ke sistem Linux bisa mencapai ribuan.

Banyaknya perangkat yang berbeda menimbulkan masalah karena perangkat keras biasanya membutuhkan driver, perangkat lunak yang memungkinkan mereka untuk berkomunikasi dengan sistem operasi yang diinstal.

Driver dapat dikompilasi sebagai bagian dari kernel Linux, dimuat ke dalam kernel sebagai modul, atau dimuat oleh perintah atau aplikasi pengguna. Perangkat eksternal, seperti pemindai dan printer, biasanya drivernya dimuat oleh aplikasi dan driver ini, pada gilirannya, berkomunikasi melalui perangkat melalui kernel melalui *interface* seperti USB.

Produsen perangkat keras sering menyediakan perangkat lunak, tetapi biasanya yang disediakan adalah untuk Microsoft Windows, bukan untuk Linux. Hal ini telah berubah seiring dengan meningkatnya popularitas Linux secara keseluruhan, tetapi pangsa pasar desktop Linux tetap sepertiga jauh dari sistem Microsoft dan Apple.

Sebagian karena dukungan vendor yang relatif jarang, ada banyak dukungan komunitas dari *developer* yang berusaha menyediakan driver untuk sistem Linux. Meskipun tidak semua perangkat keras memiliki driver yang diperlukan, namun ada jumlah yang cukup, sehingga menjadi tantangan bagi pengguna dan administrator Linux untuk menemukan driver yang tepat atau memilih perangkat keras yang memiliki tingkat dukungan tertentu di Linux.

Agar berhasil mengaktifkan perangkat di Linux, pilihan yang terbaik adalah memeriksa distribusi Linux untuk melihat apakah perangkat tersebut berlisensi untuk bekerja dengan distribusi itu. Distribusi komersial seperti Red Hat dan SUSE memiliki halaman web yang didedikasikan untuk mencantumkan perangkat keras yang disertifikasi atau disetujui untuk bekerja dengan perangkat lunak mereka.

Catatan : Tips agar berhasil menghubungkan perangkat Anda: hindari perangkat baru atau perangkat yang sangat khusus dan tanyakan kepada vendor perangkat untuk mengetahui apakah mereka mendukung Linux sebelum melakukan pembelian.

10. PERANGKAT VIDEO DISPLAY

Untuk menampilkan output ke monitor, sistem komputer harus memiliki perangkat tampilan video (video *card*) dan monitor. Perangkat tampilan video sering kali langsung terpasang atau dipasang ke motherboard, meskipun perangkat ini juga dapat dihubungkan melalui slot bus PCI pada motherboard.

Dengan banyaknya urutan vendor perangkat video, setiap perangkat tampilan video biasanya membutuhkan driver yang berpemilik yang disediakan oleh vendor. Semua driver, tetapi driver perangkat video, harus ditulis untuk sistem operasi tertentu, sesuatu yang biasa dilakukan untuk Microsoft Windows, tetapi tidak selalu untuk Linux. Untungnya, sebagian besar vendor tampilan video yang lebih besar sekarang menyediakan setidaknya beberapa tingkat dukungan Linux. Komunitas Linux telah melakukan tugas yang luar biasa dalam mengembangkan driver standar untuk kartu video.

Catatan : Masalah dukungan video yang dialami Linux sebagian besar difokuskan pada pasar desktop untuk Linux. Pangsa pasar server sebagian besar tidak terpengaruh oleh GUI, karena sebagian besar implementasi server Linux adalah mode teks.

Ada tiga jenis kabel video yang umum digunakan: versi lama namun masih digunakan kabel Video Graphics Array (VGA) 15-pin analog, yang cukup baru 29-pin antarmuka Digital Visual Interface (DVI) dan yang terakhir High-Definition Multimedia Interface yang sangat banyak digunakan (HDMI) yang mendukung resolusi hingga rentang 4K atau Ultra HD.

Agar monitor dapat bekerja dengan baik dengan perangkat tampilan video, mereka harus mampu mendukung resolusi yang sama dengan perangkat tampilan video. Biasanya, perangkat lunak yang menggerakkan perangkat tampilan video dapat secara otomatis mendeteksi resolusi maksimum yang dapat didukung oleh tampilan video dan monitor dan mengatur resolusi layar ke nilai itu.

11. POWER SUPPLIES

Power supplies adalah perangkat yang mengubah arus bolak-balik (120v, 240v) menjadi arus searah, yang digunakan komputer pada berbagai tegangan (3,3v, 5v, 12v). Power supplies umumnya tidak dapat diprogram; namun, fungsi yang tepat memiliki pengaruh besar pada sistem lainnya.

Meskipun biasanya tidak dirancang sebagai penekan lonjakan arus, perangkat ini sering kali melindungi komputer dari fluktuasi tegangan yang berasal dari sumber daya. Baik bagi administrator jaringan untuk memilih Power supplies berdasarkan kualitas daripada harga, karena Power supplies yang rusak dapat mengakibatkan kerusakan yang signifikan pada sistem komputer.

Sistem desktop, menara server, rak, dan sistem Standalone lebih rentan terhadap fluktuasi daya daripada sistem laptop. Jika daya berfluktuasi, ini dapat menyebabkan banyak kerusakan pada sistem berbasis non-baterai, sedangkan laptop terus menarik daya dari baterai internalnya hingga habis.

12. PENDAHULUAN PENYIMPANAN DATA

Pengingat : Ketika kebanyakan orang memilih Linux, mereka sebenarnya merujuk pada kombinasi perangkat lunak yang disebut GNU / Linux, yang mendefinisikan sistem operasi. GNU adalah perangkat lunak gratis yang mengelilingi kernel dan menyediakan sumber terbuka yang setara dengan banyak perintah UNIX umum. Bagian Linux dari kombinasi ini adalah kernel Linux, yang merupakan inti dari sistem operasi. Kernel dimuat pada saat boot dan tetap berjalan untuk mengelola setiap aspek dari sistem yang berfungsi.

Implementasi kernel Linux mencakup banyak subsistem yang merupakan bagian dari kernel itu sendiri dan subsistem lainnya yang dapat dimuat secara modular bila diperlukan. Fungsi utama dari kernel Linux adalah sebagai antarmuka

panggilan sistem, manajemen proses, manajemen memori, sistem file virtual, jaringan, dan driver perangkat.

Singkatnya, melalui shell, kernel menerima perintah dari pengguna dan mengelola proses yang menjalankan perintah tersebut dengan memberi mereka akses ke perangkat seperti memori, disk, antarmuka jaringan, keyboard, mouse, monitor, dan lainnya.

Sistem Linux pada umumnya memiliki ribuan file. **Filesystem Hierarchy Standard** memberikan panduan untuk distribusi tentang cara mengatur file-file. Penting untuk memahami peran kernel Linux dan bagaimana kernel ini memproses dan menyediakan informasi tentang *file sistem pseudo* di bawah /proc dan /sys .

13. PROSES

Kernel menyediakan akses ke informasi tentang proses aktif melalui file sistem *pseudo* yang terlihat di bawah direktori /proc . Perangkat keras tersedia melalui file khusus di bawah direktori /dev , sedangkan informasi tentang perangkat tersebut dapat ditemukan di file sistem *pseudo* lain di bawah direktori /sys .

File sistem *pseudo* tampak seperti file nyata pada disk tetapi file tersebut hanya ada di memori. Sebagian besar file sistem *pseudo* seperti /proc dirancang agar tampak seperti pohon hierarki dari akar sistem direktori, file, dan subdirektori, tetapi pada kenyataannya hanya ada di memori sistem, dan hanya tampak berada di perangkat penyimpanan dimana sistem file root aktif.

Direktori /proc tidak hanya berisi informasi tentang proses yang sedang berjalan, seperti namanya, direktori ini juga berisi informasi tentang perangkat keras sistem dan konfigurasi kernel saat ini.

Direktori /proc dibaca, dan informasinya digunakan oleh banyak perintah yang berbeda pada sistem, tetapi tidak terbatas pada

top, free, mount, umount dan lainnya. Perintah ini jarang diperlukan pengguna untuk menggali direktori /proc secara langsung — lebih mudah menggunakan perintah yang memanfaatkan informasinya.

Lihat contoh dibawah :

```
sysadmin@localhost:~$ ls /proc
1           cpuinfo      irq          modules       sys
128         crypto       kallsyms     mounts        sysrq-
                                                    trigger
17          devices      kcore        mtrr          sysvipc
21          diskstats    key-users    net           thread-self
23          dma          keys         pagetypeinfo  timer_list
39          driver       kmsg         partitions    timer_stats
60          execdomains  kpagecgroup  sched_debug   tty
72          fb           kpagecount   schedstat     uptime
acpi        filesystems  kpageflags   scsi          version
buddyinfo   fs           loadavg      self          version_
                                                    signature
bus         interrupts   locks        slabinfo      vmallocinfo
cgroups     iomem        mdstat       softirqs      vmstat
cmdline     ioports      meminfo      stat          zoneinfo
consoles    ipmi         misc         swaps
```

Output menunjukkan ada berbagai direktori yang bernama dan bernomor. Direktori bernomor untuk setiap proses yang berjalan di sistem, di mana nama direktori cocok dengan ID proses (PID) untuk proses yang sedang berjalan.

Misalnya, angka 72 menunjukkan PID 72, program yang sedang berjalan, yang diwakili oleh direktori dengan nama yang sama, berisi banyak file dan subdirektori yang menjelaskan proses yang sedang berjalan, konfigurasinya, penggunaan memori, dan banyak item lainnya.

Pada sistem Linux yang berjalan, selalu ada ID proses atau PID 1.

Ada juga sejumlah file biasa di direktori / proc yang menyediakan informasi tentang kernel yang sedang berjalan :

File	Isi
`/proc/cmdline`	Informasi yang diteruskan ke kernel saat pertama kali dijalankan, seperti parameter baris perintah dan instruksi khusus
`/proc/meminfo`	Informasi tentang penggunaan memori oleh kernel
`/proc/modules`	Daftar modul yang saat ini dimuat ke dalam kernel untuk menambahkan tambahan fungsionalitas

Sekali lagi, yang ingin melihat file-file ini secara langsung, karena perintah lain menawarkan output yang lebih ramah pengguna dan cara alternatif untuk melihat informasi ini.

Pertimbangkan hal ini :

Meskipun sebagian besar "file" di bawah direktori /proc tidak dapat diubah, bahkan oleh pengguna root, "file" di bawah direktori /proc/sys berpotensi untuk diubah oleh pengguna root. Memodifikasi file-file ini mengubah perilaku kernel Linux.

Modifikasi langsung dari file-file ini hanya menyebabkan perubahan sementara pada kernel. Untuk membuat perubahan permanen (tetap ada bahkan setelah reboot), entri dapat ditambahkan ke bagian yang sesuai dari file /etc/sysctl.conf .

Misalnya, direktori /proc/sys/net/ipv4 berisi file bernama icmp_echo_ignore_all. Jika file itu berisi karakter nol 0, seperti biasanya, maka sistem akan menanggapi permintaan icmp. Jika file itu berisi karakter satu 1, maka sistem tidak akan menanggapi permintaan icmp:

```
sysadmin@localhost:~$ su -
Password:
root@localhost:~# cat /proc/sys/net/ipv4/icmp_echo_ignore_all
0
root@localhost:~# ping -c1 localhost
PING localhost.localdomain (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost.localdomain (127.0.0.1): icmp_seq=1 ttl
=64 time=0.026 ms


--- localhost.localdomain ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.026/0.026/0.026/0.000 ms
root@localhost:~# echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_
all
root@localhost:~# ping -c1 localhost
PING localhost.localdomain (127.0.0.1) 56(84) bytes of data.
```

```
--- localhost.localdomain ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 10000
ms
```

13.1 HIERARKI PROSES

Ketika kernel selesai memproses selama prosedur boot, ia memulai proses *init* dan memberinya PID 1. Proses ini kemudian memulai proses sistem lainnya, dan setiap proses diberi PID secara berurutan.

Pertimbangkan hal ini :

Pada sistem berbasis **Sistem V**, proses init merupakan program /sbin/init . Pada sistem berbasis **systemd**, file `/bin/systemd` biasanya dapat dijalankan tetapi hampir selalu terhubung ke `/lib/system/systemd` yang dapat dieksekusi.

Terlepas dari proses init sistem apa yang sedang dijalankan, informasi tentang proses tersebut dapat ditemukan di direktori /proc/1 .

Karena salah satu proses init memulai proses lain, mereka, pada gilirannya, dapat memulai proses, yang dapat memulai proses lain, terus menerus. Ketika satu proses memulai proses lain, proses yang memulai disebut proses induk dan proses yang dimulai disebut proses anak. Saat melihat proses, PID induk diberi label PPID.

Ketika sistem telah berjalan untuk waktu yang lama, pada akhirnya sistem akan mencapai nilai PID maksimum, yang dapat dilihat dan dikonfigurasi melalui file /proc/sys/kernel/pid_max . Setelah PID terbesar digunakan, sistem "beralih" dan melanjutkan tanpa hambatan dengan menetapkan nilai PID yang tersedia di bagian bawah rentang.

Proses dapat "dipetakan" ke dalam pohon keluarga dari pasangan induk dan anak. Jika Anda ingin melihat pohon ini, gunakan perintah `pstree`:

```
sysadmin@localhost:~$ pstree
init-+-cron


     |-login---bash---pstree
     |-named---18*[{named}]
     |-rsyslogd---2*[{rsyslogd}]
     `-sshd
```

Jika Anda memeriksa hubungan proses induk dan anak menggunakan output dari perintah sebelumnya, itu bisa dijelaskan sebagai berikut:

- init adalah induk dari login
- login adalah anak init

- login adalah induk dari bash
- bash adalah anak login

- bash adalah induk dari pstree
- pstree adalah anak bash

13.2 MELIHAT PROSES

Cara lain untuk melihat proses yang sedang berjalan saat ini adalah dengan perintah `ps`. Secara default, perintah `ps` hanya menampilkan proses saat ini yang berjalan di shell saat ini. Sayangnya, perintah `ps` menyertakan dirinya sendiri dalam output ketika kita hanya ingin melihat proses yang lain:

```
sysadmin@localhost:~$ ps
  PID TTY          TIME CMD
 6054 ?        00:00:00 bash
 6070 ?        00:00:01 xeyes
 6090 ?        00:00:01 firefox
 6146 ?        00:00:00 ps
```

Jika Anda menjalankan `ps` dengan tambahan `--forest`, maka, mirip dengan perintah `pstree`, perintah ini menunjukkan baris untuk menunjukkan hubungan induk dan anak:

```
sysadmin@localhost:~$ ps --forest
  PID TTY          TIME CMD


 6054 ?        00:00:00 bash
 6090 ?        00:00:02   \_ firefox
 6180 ?        00:00:00   \_ dash
 6181 ?        00:00:00       \_ xeyes
 6188 ?        00:00:00       \_ ps
```

Untuk dapat melihat semua proses pada sistem, jalankan perintah `ps aux` atau perintah `ps -ef`:

```
sysadmin@localhost:~$ ps aux | head
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME
COMMAND
root         1  0.0  0.0  17872  2892 ?        Ss   08:06   0:00
/sbin?? /init
```

```
syslog      17  0.0  0.0 175744  2768 ?        Sl   08:06   0:00
/usr/sbin/rsyslogd -c5
root        21  0.0  0.0  19124  2092 ?        Ss   08:06   0:00
/usr/sbin/cron
root        23  0.0  0.0  50048  3460 ?        Ss   08:06   0:00
/usr/sbin/sshd
bind        39  0.0  0.0 385988 19888 ?       Ssl  08:06   0:00
/usr/sbin/named -u bind
root        48  0.0  0.0  54464  2680 ?        S    08:06   0:00
/bin/login -f
sysadmin    60  0.0  0.0  18088  3260 ?        S    08:06   0:00
-bash
sysadmin   122  0.0  0.0  15288  2164 ?        R+   16:26   0:00
ps aux
sysadmin   123  0.0  0.0  18088   496 ?        D+   16:26   0:00
-bash
sysadmin@localhost:~$ ps -ef | head
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 08:06 ?     00:00:00 /sbin?? /init
syslog      17     1  0 08:06 ?     00:00:00 /usr/sbin/rsyslogd
                                             -c5
root        21     1  0 08:06 ?     00:00:00 /usr/sbin/cron
root        23     1  0 08:06 ?     00:00:00 /usr/sbin/sshd
bind        39     1  0 08:06 ?     00:00:00 /usr/sbin/named
                                             -u bind
root        48     1  0 08:06 ?     00:00:00 /bin/login -f
sysadmin    60    48  0 08:06 ?     00:00:00 -bash
sysadmin   124    60  0 16:46 ?     00:00:00 ps -ef
sysadmin   125    60  0 16:46 ?     00:00:00 head
```

Output dari semua proses yang berjalan pada sistem bisa sangat banyak. Dalam contoh sebelumnya, output dari perintah `ps` difilter oleh perintah `head`, jadi hanya sepuluh proses pertama yang ditampilkan. Jika Anda tidak memfilter output dari perintah `ps`, maka Anda mungkin harus menelusuri ratusan proses untuk menemukan apa yang Anda cari.

Cara yang biasa digunakan untuk mengurangi jumlah baris output yang mungkin harus disortir oleh pengguna adalah dengan menggunakan perintah `grep` untuk memfilter baris output yang sesuai dengan *keyword*, seperti nama proses. Misalnya, jika hanya ingin melihat informasi tentang proses `firefox`, jalankan perintah seperti:

```
sysadmin@localhost:~$ ps -e | grep firefox
 6090 pts/0    00:00:07 firefox
```

Mengirim output ke pager seperti perintah `less` juga dapat membuat output dari perintah `ps` lebih mudah diatur.

Seorang administrator mungkin ingin mengetahui proses pengguna lain. Ada beberapa macam opsi yang didukung perintah `ps`, yang menghasilkan cara yang berbeda untuk melihat proses pengguna individu. Ketika menggunakan opsi UNIX tradisional untuk melihat proses pengguna tertentu, gunakan opsi `-u`:

```
sysadmin@localhost:~$ ps -u root
  PID TTY          TIME CMD
    1 ?        00:00:00 init
   13 ?        00:00:00 cron
   15 ?        00:00:00 sshd
   43 ?        00:00:00 login
```

13. MELIHAT PROSES SECARA REAL TIME

Jika perintah `ps` memberikan gambaran umum dari proses yang berjalan saat perintah dijalankan, perintah `top` memiliki antarmuka dinamis berbasis layar yang secara teratur memperbarui output dari proses yang sedang berjalan. Penggunaan perintah `top` sebagai berikut:

```
sysadmin@localhost:~$ top
```

Secara default, output dari perintah `top` diurutkan berdasarkan persentase % waktu CPU yang saat ini digunakan oleh setiap proses, nilai yang lebih tinggi akan dicantumkan terlebih dahulu, yang berarti lebih banyak proses intensif CPU yang dicantumkan terlebih dahulu:

```
top - 00:26:56 up 28 days, 20:53,  1 user,  load average: 0.11,
0.15, 0.17

Tasks:   8 total,   1 running,   7 sleeping,   0 stopped,   0 z
ombie

%Cpu(s):  0.2 us,  0.2 sy,  0.0 ni, 99.6 id,  0.0 wa,  0.0 hi,
0.0 si,  0.0 st

KiB Mem : 13201464+total, 76979904 free, 47522152 used,  751258
0 buff/cache

KiB Swap: 13419622+total, 13415368+free,    42544 used. 8386745
6 avail Mem



  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     T
IME+ COMMAND

    1 root      20   0   18376   3036   2764 S   0.0  0.0   0:0
0.12 init
```

```
    9 syslog    20   0  191328   5648   3100 S   0.0  0.0   0:0
0.04 rsyslogd

   13 root      20   0   28356   2552   2320 S   0.0  0.0   0:0
0.00 cron

   15 root      20   0   72296   3228   2484 S   0.0  0.0   0:0
0.00 sshd

   24 bind      20   0  878888  39324   7148 S   0.0  0.0   0:0
2.72 named

   43 root      20   0   78636   3612   3060 S   0.0  0.0   0:0
0.00 login

   56 sysadmin  20   0   18508   3440   3040 S   0.0  0.0   0:0
0.00 bash

   72 sysadmin  20   0   36600   3132   2696 R   0.0  0.0   0:0
0.03 top
```

Ada sejumlah besar perintah interaktif yang dapat dijalankan dari program `top` yang sedang berjalan. Gunakan tombol **H** untuk melihat daftar lengkap.

Salah satu keuntungan dari perintah `top` adalah dapat dibiarkan berjalan agar tetap berada di atas proses untuk tujuan pemantauan. Jika suatu proses mulai mendominasi, atau berlarian dengan sistem, maka secara default akan muncul di bagian atas daftar yang disajikan oleh perintah `top`. Administrator yang menjalankan perintah `top` kemudian dapat melakukan salah satu dari dua tindakan:

Key	Action
K	Menghentikan proses runaway.
R	Menyesuaikan prioritas proses.

Menekan tombol **K** saat perintah `top` sedang berjalan akan meminta pengguna untuk memberikan PID dan kemudian nomor sinyal. Mengirim sinyal default meminta proses dihentikan, tetapi pengiriman sinyal nomor 9, yaitu sinyal KILL, memaksa proses untuk berhenti.

Menekan tombol **R** saat perintah `top` dijalankan akan meminta pengguna untuk proses *renice*, dan kemudian untuk *niceness values*. *Niceness values* dapat berkisar dari -20 hingga 19, dan memengaruhi prioritas. Hanya pengguna root yang dapat menggunakan *niceness values* yang lebih rendah dari yang sekarang atau *niceness values* negatif, yang menyebabkan proses berjalan dengan prioritas yang ditingkatkan. Setiap pengguna dapat memberikan *niceness values* yang lebih tinggi dari *niceness values* saat ini, yang menyebabkan proses berjalan dengan prioritas yang lebih rendah.

Keuntungan lain dari perintah `top` adalah dapat memberikan representasi keseluruhan tentang seberapa sibuk sistem saat ini dan *trend* dari waktu ke waktu.

```
top - 00:26:56 up 28 days, 20:53,  1 user,  load average: 0.11,
0.15, 0.17
Tasks:   8 total,   1 running,   7 sleeping,   0 stopped,   0 z
ombie
%Cpu(s):  0.2 us,  0.2 sy,  0.0 ni, 99.6 id,  0.0 wa,  0.0 hi,
0.0 si,  0.0 st
KiB Mem : 13201464+total, 76979904 free, 47522152 used,  751258
0 buff/cache
KiB Swap: 13419622+total, 13415368+free,    42544 used. 8386745
6 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     T
IME+ COMMAND
    1 root      20   0   18376   3036   2764 S   0.0  0.0   0:0
0.12 init
    9 syslog    20   0  191328   5648   3100 S   0.0  0.0   0:0
0.04 rsyslogd
```

```
   13 root      20   0   28356   2552   2320 S   0.0  0.0   0:0
0.00 cron
   15 root      20   0   72296   3228   2484 S   0.0  0.0   0:0
0.00 sshd
   24 bind      20   0  878888  39324   7148 S   0.0  0.0   0:0
2.72 named
   43 root      20   0   78636   3612   3060 S   0.0  0.0   0:0
0.00 login
   56 sysadmin  20   0   18508   3440   3040 S   0.0  0.0   0:0
0.00 bash
   72 sysadmin  20   0   36600   3132   2696 R   0.0  0.0   0:0
0.03 top
```

Rata-rata beban yang ditunjukkan pada baris pertama output dari perintah `top` menunjukkan seberapa sibuk sistem selama satu, lima, dan lima belas menit terakhir. Informasi ini juga dapat dilihat dengan menjalankan perintah `uptime` atau secara langsung dengan menampilkan konten file /proc/loadavg :

Catatan : Untuk keluar dari program `top` dan kembali ke prompt, ketik **q**.

```
sysadmin@localhost:~$ cat /proc/loadavg

0.12 0.46 0.25 1/254 3052
```

- **Rata-rata muatan:**

```
0.12 0.46 0.25 1/254 3052
```

Tiga angka pertama dalam file ini menunjukkan rata-rata muatan selama interval satu, lima, dan lima belas menit terakhir.

- **Angka Proses:**

```
0.12 0.46 0.25 1/254 3052
```

Nilai keempat adalah pecahan yang menunjukkan jumlah proses yang saat ini menjalankan kode pada CPU 1 dan jumlah proses 254

- **PID Terakhir:**

```
0.12 0.46 0.25 1/254 3052
```

Nilai kelima adalah nilai PID terakhir yang mengeksekusi kode pada CPU.

Jumlah yang dilaporkan sebagai rata-rata muatan sebanding dengan jumlah inti CPU yang mampu menjalankan proses. Pada CPU core tunggal, nilai satu (1) berarti sistem terisi penuh. Pada CPU core empat nilai 1 berarti sistem hanya terisi 1/4 atau 25% saja.

Alasan lain administrator ingin menjalankan perintah `top` adalah kemampuan untuk melihat penggunaan memori secara real-time. Baik perintah `top` atau `free`, keduanya menampilkan statistik tentang penggunaan memori.

Perintah `top` juga dapat menunjukkan persentase memori yang digunakan oleh setiap proses, sehingga proses yang menghabiskan banyak memori dapat dengan cepat diidentifikasi.

14. MEMORI

Memori pada sistem Linux modern diatur dan dikelola oleh kernel. Memori perangkat keras pada sistem digunakan bersama oleh semua proses pada sistem, melalui metode yang disebut *virtual addressing*. Memori fisik dapat direferensikan oleh sejumlah proses, yang mana mungkin mengira proses mampu menangani lebih banyak memori daripada yang sebenarnya bisa dilakukan. *Virtual addressing* memungkinkan banyak proses untuk mengakses memori yang sama tanpa konflik atau *crash.*

Ini dilakukan dengan mengalokasikan area tertentu dari hard disk fisik (atau virtual) untuk digunakan sebagai pengganti RAM fisik. Memori dibagi menjadi blok unit berukuran sama yang dapat dialamatkan seperti sumber lain di

sistem. Sistem tidak hanya dapat mengakses memori dari alamat sistem lokal, tetapi juga dapat mengakses memori yang terletak di tempat lain, seperti di komputer yang berbeda, perangkat virtual, atau bahkan pada volume yang secara fisik terletak di benua lain!.

Penting untuk diperhatikan perbedaan antara ruang pengguna dan ruang kernel. Ruang kernel adalah tempat kode untuk kernel disimpan dan dijalankan. Ini biasanya berada dalam kisaran alamat memori yang "dilindungi" dan tetap terisolasi dari proses lain dengan hak istimewa yang lebih rendah. Ruang pengguna, di sisi lain, tersedia untuk pengguna dan program. Mereka berkomunikasi dengan Kernel melalui "system call" API yang bertindak sebagai perantara antara program reguler dan Kernel. Sistem yang memisahkan program yang berpotensi tidak stabil atau berbahaya dari pekerjaan Kernel yang kritis inilah yang memberi sistem Linux stabilitas dan ketahanan yang diandalkan oleh pengembang aplikasi.

14.1 MELIHAT MEMORI

Menjalankan perintah `free` tanpa opsi apa pun yang memberikan ringkasan dari memori yang digunakan pada saat itu.

```
sysadmin@localhost:~$ free
             total       used       free     shared    buffers
cached
Mem:      32953528   26171772    6781756          0       4136
22660364
-/+ buffers/cache:    3507272   29446256
Swap:            0          0          0
```

Jika Anda ingin memantau penggunaan memori dari waktu ke waktu dengan perintah `free`, Anda dapat menjalankannya dengan opsi `-s` (seberapa sering memperbarui) dan menentukan jumlah detik tersebut. Misalnya,

menjalankan perintah `free` berikut akan memperbarui output setiap sepuluh detik:

```
sysadmin@localhost:~$ free -s 10
              total        used        free      shared  buff/cache   available
Mem:      132014640    47304084    77189512        3008      752104    84085528
Swap:     134196220       42544   134153676

              total        used        free      shared  buff/cache   available
Mem:      132014640    47302928    77190668        3008      752104    84086684
Swap:     134196220       42544   134153676
```

Untuk mempermudah menafsirkan apa yang dihasilkan dari perintah `free`, opsi `-m` atau `-g` dapat berfungsi untuk menampilkan output masing-masing dalam bentuk megabyte atau gigabyte. Tanpa opsi ini, output ditampilkan dalam byte:

Saat membaca output dari perintah `free`:

- **Bagian Header:**

```
             total       used       free     shared    buffers     cached
Mem:      32953528   26171772    6781756          0       4136   22660364
-/+ buffers/cache:    3507272   29446256
Swap:            0          0          0
```

- **Statistik Memori Fisik:**

```
             total       used       free     shared    buffers     cached
Mem:      32953528   26171772    6781756          0       4136   22660364
-/+ buffers/cache:    3507272   29446256
Swap:            0          0          0
```

- **Penyesuaian Memori:**

Baris ketiga mewakili jumlah memori fisik setelah menyesuaikan nilai-nilai tersebut dengan tidak memperhitungkan memori apa pun yang digunakan oleh kernel untuk buffer dan cache. Secara teknis, memori "bekas" ini dapat "diklaim kembali" jika diperlukan:

```
             total       used       free     shared    buffers     cached
Mem:      32953528   26171772    6781756          0       4136   22660364
-/+ buffers/cache:    3507272   29446256
Swap:            0          0          0
```

- **Menukar Memori:**

Baris keempat output mengacu pada memori swap, yang mana dikenal sebagai memori virtual. Memori Virtual adalah ruang pada hard disk yang digunakan seperti memori fisik ketika jumlah memori fisik menjadi rendah. Secara efektif, ini membuat sistem tampak memiliki lebih banyak memori daripada yang dimilikinya, tetapi menggunakan ruang swap juga dapat memperlambat sistem:

```
             total       used       free     shared    buffers     cached
Mem:      32953528   26171772    6781756          0       4136   22660364
-/+ buffers/cache:    3507272   29446256
Swap:            0          0          0
```

Jika jumlah memori dan swap yang tersedia sangat rendah, maka sistem akan mulai menghentikan proses secara otomatis, sehingga penting untuk

memantau penggunaan memori sistem. Seorang administrator yang melihat sistem kehabisan memori bebas dapat menggunakan perintah `top` atau `kill` untuk menghentikan proses pilihan mereka sendiri, daripada membiarkan sistem yang memilih prosesnya.

15. FILE LOG

Karena kernel dan berbagai proses berjalan di sistem, mereka menghasilkan output yang menjelaskan bagaimana mereka berjalan. Beberapa dari output ini ditampilkan sebagai output standar dan *error* di jendela terminal tempat proses dijalankan, meskipun sebagian dari data ini tidak dikirim ke layar. Output sisanya ditulis ke berbagai file. Informasi ini disebut *file log* atau *log messages.*

File log berguna untuk berbagai alasan; file log membantu memecahkan masalah dan menentukan apakah akses tidak sah telah dicoba atau tidak.

Beberapa proses dapat mencatat datanya sendiri ke file-file ini, proses lain mengandalkan proses terpisah (daemon) untuk menangani file data log ini.

Catatan: *Syslog* adalah istilah yang digunakan secara umum untuk mendeskripsikan logging di sistem Linux seperti yang telah digunakan selama beberapa waktu. Dalam beberapa kasus, ketika seorang author mengatakan *syslog* yang sebenarnya mereka maksud adalah *sistem pencatatan (apa pun) yang saat ini digunakan pada distribusi ini*.

Logging Daemons berbeda dari dua cara utama pada distribusi terbaru. Metode lama untuk melakukan *logging system* adalah dua daemon (bernama syslogd dan klogd) yang bekerja bersama, tetapi di distribusi yang lebih baru, satu layanan bernama rsyslogd menggabungkan dua fungsi tadi dan lebih banyak lagi menjadi satu daemon.

Pada distribusi yang lebih baru, yang berbasis `systemd`, daemon logging dinamai journald, dan log dirancang untuk menerima output teks, tetapi juga

biner. Metode standar untuk melihat log berbasis journald adalah dengan menggunakan perintah `journalctl`.

Terlepas dari apa proses daemon yang digunakan, file log itu sendiri hampir selalu ditempatkan ke dalam struktur direktori /var/log . Meskipun beberapa nama file mungkin berbeda, berikut beberapa file umum yang dapat ditemukan di direktori ini:

File	Contents
`boot.log`	Pesan yang dihasilkan sebagai layanan yang dimulai selama startup sistem.
`cron`	Pesan yang dibuat oleh daemon crond untuk pekerjaan yang akan dijalankan secara berulang.
`dmesg`	Pesan yang dihasilkan oleh kernel selama boot sistem.
`maillog`	Pesan yang dihasilkan oleh daemon mail untuk pesan email yang dikirim atau diterima.
`messages`	Pesan yang berisi informasi mengenai aktivitas yang terjadi pada perangkat
`secure`	Pesan dari proses yang membutuhkan otorisasi atau otentikasi (seperti proses login).

File	Contents
`journal`	Pesan dari konfigurasi default systemd-journald.service; dapat dikonfigurasi di file /etc/journald.conf di antara tempat lain.
`Xorg.0.log`	Pesan dari server X Windows (GUI).

Anda dapat melihat konten berbagai file log menggunakan dua metode yang berbeda. Pertama, seperti kebanyakan file lainnya, Anda dapat menggunakan perintah `cat`, atau perintah `less` untuk mencari, meng-scroll, dan opsi lainnya.

Metode kedua adalah menggunakan perintah `journalctl` pada sistem berbasis systemd, terutama karena file /var/log/journal sekarang berisi informasi biner, jika menggunakan perintah `cat` atau `less` dapat menghasilkan perilaku layar yang membingungkan dari kode kontrol dan item biner di file log.

File Log Rotate, artinya mengganti file log lama ke file log yang lebih baru. Nama file yang muncul pada tabel di atas mungkin memiliki akhiran numerik atau tanggal yang ditambahkan padanya: misalnya, secure.0 atau secure-20181103.

File Log Rotate biasanya terjadi dengan jadwal yang teratur: misalnya, seminggu sekali. Ketika file log diputar, sistem berhenti menulis ke file log dan menambahkan sufiks padanya. Kemudian membuat file baru dengan nama asli, dan proses logging terus menggunakan file baru ini.

Pada mayoritas daemon modern, cenderung mengacu ke akhir tanggal. Jadi, pada hari minggu yang berakhir pada tanggal 3 November 2018, daemon

logging mungkin berhenti menulis ke */var/log/messages* (atau */var/log/journal*) , lalu mengganti nama file menjadi */var/log/messages-20181103*, dan kemudian mulai menulis ke file */var/log/messages-20181103* yang baru.

Meskipun mayoritas file log berisi teks, tetapi dapat dilihat dengan aman dengan banyak tools, file lain seperti file */var/log/btmp* dan */var/log/wtmp* yang mengandung biner. Dengan menggunakan perintah `file`, pengguna dapat memeriksa tipe konten file sebelum mereka melihatnya untuk memastikan file log aman untuk dilihat. Perintah file berikut mengklasifikasikan */var/log/wtmp* sebagai data, yang biasanya berarti file tersebut biner:

```
sysadmin@localhost:~$ file /var/log/wtmp
/var/log/wtmp: data
```

Untuk file yang berisi data biner, tersedia perintah yang akan membaca file, menafsirkan kontennya, dan kemudian mengeluarkan output teks. Sebagai contoh, perintah `lastb` dan `last` dapat digunakan untuk melihat file */var/log/btmp* dan */var/log/wtmp* .

Perintah `lastb` membutuhkan hak akses root untuk dijalankan di lingkungan mesin virtual kita.

16. PESAN KERNEL

File */var/log/dmesg* berisi pesan kernel yang dihasilkan selama startup sistem. */var/log/messages* berisi pesan kernel yang dihasilkan saat sistem berjalan, tetapi pesan tersebut bercampur dengan pesan lain dari daemon atau proses.

Meskipun kernel tidak memiliki file log sendiri, seseorang dapat mengkonfigurasinya dengan memodifikasi file */etc/syslog.conf* atau */etc/rsyslog.conf*. Selain itu, perintah `dmesg` dapat digunakan untuk melihat

kernel ring buffer, yang menampung sejumlah besar pesan yang dihasilkan oleh kernel.

Pada sistem yang aktif, atau sistem yang mengalami banyak *error* pada kernel, kapasitas buffer ini mungkin terlampaui, dan beberapa pesan mungkin hilang. Ukuran buffer ini diatur pada saat kernel dikompilasi, jadi tidak mudah untuk diubah.

Menjalankan perintah `dmesg` dapat menghasilkan teks hingga 512 kilobyte, jadi sebaiknya lakukan filter perintah dengan *pipe* ke perintah lain seperti `less` atau `grep`. Misalnya, jika pengguna memecahkan masalah dengan perangkat USB, maka mencari teks USB dengan perintah `grep` akan sangat membantu. Opsi `-i` digunakan untuk mengabaikan kasus:

```
sysadmin@localhost:~$ dmesg | grep -i usb
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
ohci_hcd 0000:00:06.0: new USB bus registered, assigned bus num
ber 1
usb usb1: New USB device found, idVendor=1d6b, idProduct=0001
usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumbe
r=1
```

17. FILESYSTEM HIERARCHY STANDARD

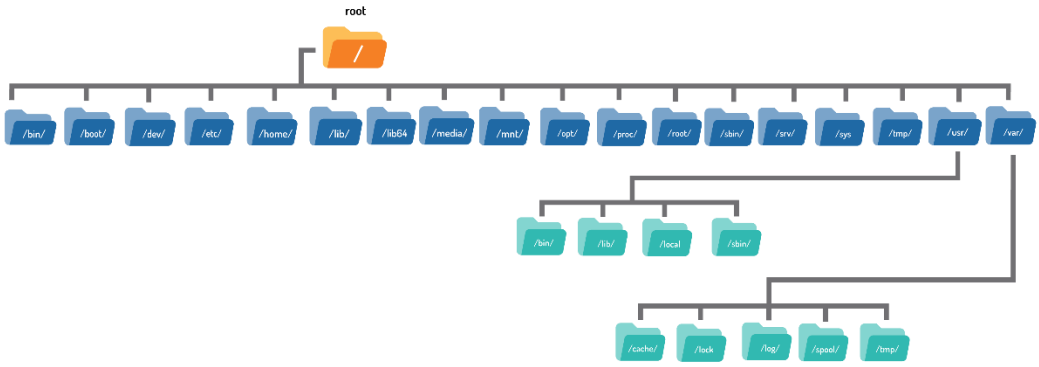
Standar adalah seperangkat aturan atau pedoman yang patut untuk diikuti. Namun, pedoman ini pasti bisa rusak, baik oleh seluruh distribusi atau oleh administrator di mesin pribadi.

Standar FHS mengkategorikan setiap direktori sistem dalam beberapa cara:

- Sebuah direktori dapat dikategorikan *shareable* dan *unshareable*, mengacu pada apakah direktori tersebut dapat dibagikan di jaringan dan digunakan oleh banyak mesin.
- Direktori dimasukkan ke dalam kategori file statis (konten file tidak akan berubah) atau file variabel (konten file dapat berubah).

Untuk membuat klasifikasi ini, seringkali perlu untuk merujuk ke subdirektori di bawah direktori tingkat atas. Misalnya, direktori `/var` itu sendiri tidak dapat dikategorikan sebagai dapat dibagikan atau tidak dapat dibagikan, tetapi salah satu subdirektorinya, direktori `/var/mail`, dapat dibagikan. Sebaliknya, direktori `/var/lock` tidak boleh dibagikan.

	Not Shareable	**Shareable**
Variable	`/var/lock`	`/var/mail`
Static	`/etc`	`/opt`



Standar FHS mendefinisikan empat hierarki direktori yang digunakan dalam mengatur file sistem file. Berikut adalah hierarki tingkat atas atau akar:

Directory	Contents
/	Dasar dari struktur, atau root sistem file, direktori ini menyatukan semua direktori terlepas dari apakah itu partisi lokal, perangkat yang dapat dilepas, atau jaringan bersama
/bin	Biner penting seperti perintah `ls`, `cp`, dan `rm`, dan menjadi bagian dari sistem berkas root
/boot	File yang diperlukan untuk mem-boot sistem, seperti kernel Linux dan file konfigurasi terkait

Directory	Contents
/dev	File yang mewakili perangkat keras dan file khusus lainnya, seperti file /dev/null dan /dev/zero
/etc	File konfigurasi host penting seperti file /etc/hosts atau /etc/
/home	Direktori beranda pengguna
/lib	Library penting untuk mendukung file yang dapat dieksekusi di direktori /bin dan /sbin
/lib64	Library penting yang dibangun untuk arsitektur tertentu. Misalnya, direktori /lib64 untuk prosesor 64-bit AMD / Intel x86 yang kompatibel
/media	Mount point untuk media yang dapat dilepas dan dipasang secara otomatis
/mnt	Mount point untuk memasang filesystem secara manual
/opt	Lokasi penginstalan perangkat lunak pihak ketiga opsional

Directory	Contents
/proc	Sistem file virtual untuk kernel untuk melaporkan informasi proses, serta informasi lainnya
/root	Direktori home dari pengguna root
/sbin	Sistem Biner penting yang terutama digunakan oleh pengguna root
/sys	Sistem file virtual untuk informasi tentang perangkat keras yang terhubung ke sistem
/srv	Lokasi tempat layanan situs khusus yang dihosting
/tmp	Direktori tempat semua pengguna diizinkan untuk membuat file sementara dan yang akan dibersihkan saat boot
/usr	Hierarki kedua File tidak penting untuk penggunaan multi-user
/usr/local	Hirarki ketiga File untuk perangkat lunak yang tidak berasal dari distribusi
/var	Hierarki keempat File yang berubah seiring waktu

Directory	Contents
/var/cache	File yang digunakan untuk menyimpan data aplikasi
/var/log	Sebagian besar file log
/var/lock	Kunci file untuk sumber bersama
/var/spool	File gulungan untuk dicetak dan dikirim
/var/tmp	File sementara yang akan dipertahankan saat reboot

Hierarki kedua dan ketiga, terletak di bawah direktori /usr dan /usr/local, ulangi pola dari berbagai direktori kunci yang ditemukan di bawah hierarki pertama atau filesystem root. Hierarki keempat, direktori /var, juga mengulangi beberapa direktori top-level seperti lib, opt dan tmp.

Walaupun filesystem root dan isinya dianggap penting atau diperlukan untuk mem-boot sistem, direktori /var, /usr dan /usr/local dianggap tidak penting untuk proses boot. Akibatnya, filesystem root dan direktorinya mungkin satu-satunya yang tersedia dalam situasi tertentu seperti booting ke mode single-user, lingkungan yang dirancang untuk memecahkan masalah sistem.

Direktori /usr dimaksudkan untuk menampung perangkat lunak untuk digunakan oleh banyak pengguna. Direktori /usr terkadang dibagikan melalui jaringan dan dipasang sebagai read-only.

Hierarki /usr/local adalah untuk instalasi perangkat lunak yang tidak berasal dari distribusi. Seringkali direktori ini digunakan untuk perangkat lunak yang dikompilasi dari source code.

17.1 ORGANISASI FILE MENGGUNAKAN FHS

Meskipun standar FHS berguna untuk pemahaman rinci tentang tata letak direktori yang digunakan oleh sebagian besar distribusi Linux, dibawah ini terdapat penjelasan yang lebih umum tentang tata letak direktori seperti yang ada pada distribusi Linux tipikal.

Direktori Beranda Pengguna

Direktori `/home` memiliki direktori di bawahnya untuk setiap akun pengguna. Misalnya, user bob akan memiliki direktori home `/home/bob`. Biasanya, hanya user bob yang akan memiliki akses ke direktori ini. Tanpa diberikan izin khusus pada direktori lain, pengguna hanya dapat membuat file di direktori home mereka, direktori `/tmp` , dan direktori `/var/tmp` .

Direktori Biner

Direktori biner berisi program yang dijalankan oleh pengguna dan administrator untuk memulai proses atau aplikasi yang berjalan pada sistem.

Biner Pengguna Khusus

Direktori biner yang dimaksudkan untuk digunakan oleh pengguna yang tidak memiliki privilege untuk:

- `/bin`
- `/usr/bin`
- `/usr/local/bin`

Terkadang perangkat lunak pihak ketiga juga menyimpan file yang dapat dieksekusi di direktori seperti:

- `/usr/local/application/bin`
- `/opt/application/bin`

Selain itu, tidak jarang setiap pengguna memiliki direktori bin sendiri yang terletak di direktori home mereka; misalnya, `/home/bob/bin`.

Biner yang Dibatasi Root

Di sisi lain, direktori sbin utama ditujukan untuk digunakan oleh administrator sistem (pengguna root). Direktorinya adalah :

- `/sbin`
- `/usr/sbin`
- `/usr/local/sbin`

Beberapa aplikasi administratif pihak ketiga juga dapat menggunakan direktori seperti:

- `/usr/local/application/sbin`
- `/opt/application/sbin`

Tergantung pada distribusinya, variabel PATH mungkin tidak berisi semua kemungkinan direktori bin dan sbin . Untuk menjalankan perintah di salah satu direktori ini, direktori tersebut perlu disertakan dalam daftar variabel PATH, atau pengguna perlu menentukan path ke perintah tersebut.

Direktori Aplikasi Perangkat Lunak

Tidak seperti sistem operasi Windows, di mana aplikasi mungkin memiliki semua file yang diinstal dalam satu subdirektori di bawah direktori C:\Program Files , aplikasi di Linux memiliki file di beberapa direktori yang tersebar di seluruh sistem file Linux. Untuk distribusi turunan Debian, Anda dapat menjalankan perintah `dpkg -L packagename` untuk mendapatkan daftar lokasi file. Dalam distribusi turunan Red Hat, Anda dapat menjalankan perintah `rpm -ql packagename` untuk daftar lokasi berkas milik aplikasi itu.

File biner program yang dapat dieksekusi dapat masuk ke direktori /usr/bin jika disertakan dengan sistem operasi, atau mereka dapat masuk ke

direktori /usr/local/bin atau /opt/application/bin jika berasal dari direktori pihak ketiga.

Data untuk aplikasi dapat disimpan di salah satu subdirektori berikut:

- /usr/share
- /usr/lib
- /opt/application
- /var/lib

File yang terkait dengan dokumentasi dapat disimpan di salah satu subdirektori berikut:

- /usr/share/doc
- /usr/share/man
- usr/share/info

File konfigurasi global untuk aplikasi memungkinkan untuk disimpan ke subdirektori di bawah direktori /etc, sedangkan file konfigurasi yang dipersonalisasi (khusus untuk pengguna) untuk aplikasi berada di dalam subdirektori tersembunyi dari direktori beranda pengguna.

Direktori Library

Libraries adalah file yang berisi kode yang dibagikan ke beberapa program. Mayoritas nama file library diakhiri dengan ekstensi file .so, yang berarti objek bersama.

Beberapa versi library mungkin ada karena kode berbeda dalam setiap file meskipun melakukan fungsi yang sama seperti versi library lainnya. Salah satu alasan mengapa kode mungkin berbeda, meskipun mungkin melakukan hal yang sama seperti file library lainnya, adalah karena ia dikompilasi untuk dijalankan pada jenis prosesor yang berbeda. Misalnya, biasanya sistem yang menggunakan

kode yang dirancang untuk prosesor jenis Intel / AMD 64-bit memiliki library 32-bit dan library 64-bit.

Library yang mendukung program biner penting yang ditemukan di direktori `/bin` dan `/sbin` biasanya terletak di `/lib` atau`/lib64`.

Untuk mendukung file executable /usr/bin dan /usr/sbin , direktori library `/usr/lib` dan `/usr/lib64` biasanya digunakan.

Untuk aplikasi pendukung yang tidak didistribusikan dengan sistem operasi, direktori `/usr/lib` dan `/usr/lib64` sering digunakan.

Direktori Variabel Data

Direktori `/var` dan subdirektorinya berisi data yang sering berubah. Jika sistem Anda digunakan untuk kegiatan email, maka `/var/mail` atau `/var/spool/mail` biasanya digunakan untuk menyimpan data email pengguna. Jika Anda mencetak dari sistem Anda, maka direktori `/var/spool/cups` digunakan untuk menyimpan sementara data yang ingin di *print*.

Tergantung pada peristiwa apa yang sedang dicatat dan seberapa banyak aktivitas yang terjadi, sistem menentukan seberapa besar file log-nya. Pada sistem yang sibuk, mungkin ada data dengan jumlah yang besar di file log. File-file ini disimpan di direktori `/var/log`.

Meskipun file log dapat berguna untuk memecahkan masalah, mereka juga dapat menyebabkan masalah. Hal yang perlu diperhatikan pada semua direktori ini adalah direktori tersebut dapat mengisi ruang disk dengan cepat pada sistem yang aktif. Jika direktori `/var` bukan merupakan partisi terpisah, maka sistem berkas root dapat menjadi penuh dan menyebabkan kemacetan pada sistem.

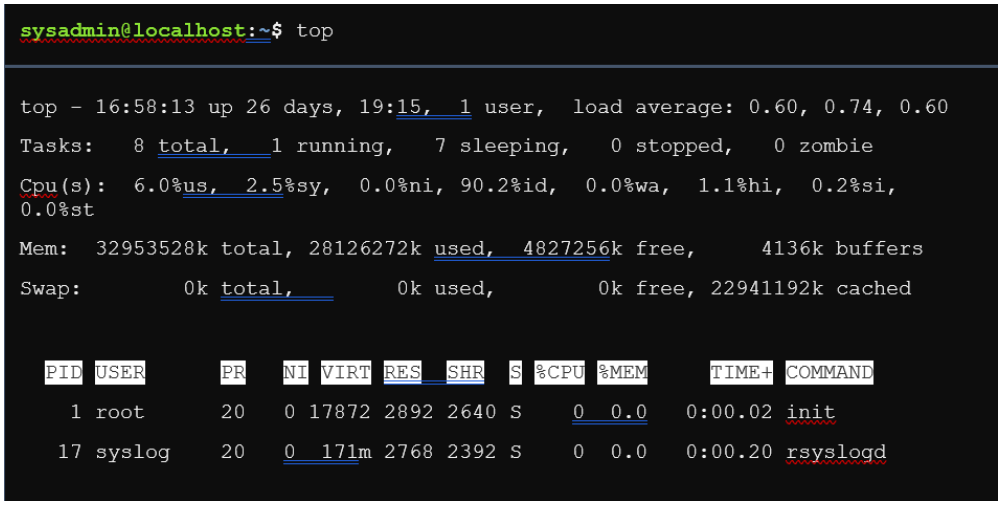
RANGKUMAN

1. Dibalik beroperasinya suatu komputer, terdapat komponen-komponen yang saling terhubung di dalamnya. Beberapa komponen tersebut adalah Motherboard, Prosesor, RAM, Buses, Hard Drive, SSD, Power Supply, dan sebagainya. Untuk mengelola hardware oleh linux, kita harus menggunakan driver agar hardware compatible dalam berkomunikasi dengan sistem operasi.
2. Untuk memproses perintah yang diberikan user dan menangani interupsi lalu diteruskan ke perangkat, gunakan Kernel Linux. Kernel akan memberikan akses tentang informasi mengenai proses aktif melalui file sistem *pseudo* yang terlihat dibawah direktori `/proc`, informasi tentang perangkat keras tersedia melalui file khusus di bawah direktori `/dev` , dan informasi tentang perangkat tersebut dapat ditemukan di file sistem pseudo lain di bawah direktori `/sys`.

   Berikut merupakan contoh penggunaan perintah untuk melihat proses yang sedang berjalan :

```
sysadmin@localhost:~$ ls /proc
1           cpuinfo     irq         modules     sys
128         crypto      kallsyms    mounts      sysrq-trigger
17          devices     kcore       mtrr        sysvipc
21          diskstats   key-users   net         thread-self
```

3. Untuk melihat informasi lain seperti penggunaan memori, penggunaan proses secara real time, dan menampilkan pesan kernel.
4. Untuk melihat informasi seputar memori secara singkat, kita dapat menggunakan command `ps`, jika ingin melihat informasi seputar memori secara detail, kita dapat menggunakan command `top`.



5. Data mengenai proses yang sedang berjalan saat ini, beberapa outputnya ditampilkan pada terminal, dan output lainnya akan ditulis di file log.
6. Filesystem Hierarchy Standard merupakan standard khusus yang menjadi acuan dalam pengelompokkan data. Yang mana FHS itu dikategorikan menjadi *sharable* dan *unsharable* serta static dan variabel. Dalam kategori tersebut, terdapat 4 hierarki dalam FHS.

Referensi:

https://www.netacad.com/courses/os-it/ndg-linux-essentials