Modul Workshop Network and Security Lab B201

Prerequisite

Agar lebih mudah untuk mengikuti, diharapkan peserta sudah memiliki sistem maupun VM yang terinstall linux. Linux yang digunakan boleh dari Debian, Ubuntu, Arch, maupun turunannya.

Kemudian, buka terminal atau command prompt dan jalankan perintah berikut:

```bash
cd && wget --no-check-certificate https://github.com/LordRonz/workshop-requirements/
```

> ***Note:*** *Biasanya, cara paste ke terminal adalah menggunakan* `CTRL + SHIFT + V`*, dan copy adalah* `CTRL + SHIFT + C`*. Bisa juga dengan melakukan klik kanan lalu pilih opsi copy atau paste.*

A. Pengenalan Linux

Linux/UNIX merupakan salah satu contoh dari open source yang sering digunakan oleh instansi atau masyarakat umum. Pada tahun 1969, Ken Thompson dan Dennis Ritchie (developer bahasa C), para peneliti di AT&T Bell Laboratory Amerika, membuat sistem operasi UNIX, yaitu cikal bakal dari Linux. UNIX mendapatkan perhatian besar karena merupakan sistem operasi pertama yang dibuat bukan oleh hardware maker. Selain itu, karena seluruh source code yang pernah dibuat menggunakan bahasa C, sehingga mempermudah pemindahannya ke berbagai platform. (Priambodo, 2014).

Dalam waktu singkat UNIX berkembang secara pesat dan terpecah dalam dua aliran. yaitu UNIX yang dikembangkan oleh Universitas Berkeley dan yang dikembangkan oleh AT&T. Dari sini lahirlah proyek POSIX (Portable Operating System Interface for UNIX) yang dimotori oleh IEEE (The Institute of Electrical and Electronics Engineers) yang bertujuan untuk menetapkan spesifikasi standar UNIX. (Priambodo, 2014). Tentu saja untuk memberikan perspektif lebih silahkan ke **wikipedia**.

B. Linux System Administration

Linux System administration merupakan perintah-perintah dalam mengoperasikan sistem operasi Linux. Mulai dari instalasi, pemeliharaan, hingga tool-tool untuk mendeteksi gejala-gejala yang mungkin akan menimbulkan masalah.

Berikut ini adalah beberapa dari command-command linux yang penting untuk diketahui.

> man

reference >

Command `man` merupakan command untuk menampilkan manual page dari system. Setiap argumen page biasanya adalah nama dari program, fungsi (biasanya bahasa C) atau utilitas. Kemudian manual page untuk program tersebut akan ditampilkan. Sebuah section apabila diberikan maka akan membuat man hanya menampilkan bagian section tersebut.

Daftar dibawah merupakan section number dari manual dan diikuti dengan tipe halaman yang ada di dalamnya.

Section Number	Type
1	Executable programs or shell commands
2	System calls (functions provided by the kernel)
3	Library calls (functions within program libraries)
4	Special files (usually found in /dev)
5	File formats and conventions, e.g. /etc/passwd
6	Games
7	Miscellaneous (including macro packages and conventions), e.g. man(7), groff(7)
8	System administration commands (usually only for root)
9	Kernel routines [Non standard]

Berikut ini adalah contoh penggunaannya.

```bash
man man
```

Command dibawah akan menampilkan manual dari fungsi `fork` dalam bahasa C

```bash
man 3 fork
```

> pwd

reference >

Command `pwd` atau **p**rint **w**orking **d**irectory akan menampilkan nama dari direktori terkini dari terminal yang digunakan untuk memanggil command ini

Contoh penggunaan:

```bash
pwd
```

Maka outputnya kira-kira seperti `/home/amogus/HOMEWORK`

> cd

reference >

Command `cd` atau *change directory* berfungsi untuk menavigasi atau berpindah pindah lokasi direktori pada sistem file di Linux. Contoh penggunaannya adalah sebagai berikut

```bash
cd Documents
```

Maka kita akan bergerak maju satu level ke direktori bernama Documents. Kita juga bisa bergerak mundur dengan menggunakan `..`

```bash
cd ../
```

Maka kita kembali ke direktori sebelumnya.

Kita juga bisa menggunakan argumen `-` untuk berpindah ke direktori sebelumnya.

```bash
cd -
```

> ls

reference >

Command `ls` atau list ini berfungsi untuk menampilkan informasi dari file-file yang berada pada direktori tertentu (secara default adalah direktori terkini). Untuk penggunaan lengkapnya bisa dilihat di reference atau dengan command `man ls`.

Command dibawah akan menampilkan file-file yang tidak *hidden* pada `pwd`.

```bash
ls
```

Kira-kira outputnya seperti ini

```
❯ ls
commitlint.config.js  next-sitemap.js    src
cypress               node_modules       tailwind.config.js
cypress.json          package.json       tsconfig.json
LICENSE               postcss.config.js  yarn-error.log
next.config.mjs       public             yarn.lock
next-env.d.ts         README.md
```

Apabila kita menggunakan argumen seperti `-l` maka outputnya akan lebih deskriptif, dan bila menggunakan `-A` maka outputnya akan menyertakan file yang berawalan dengan titik (.) alias hidden. Argumen `-h` akan mengubah tampilan ukuran file menjadi human readable. Biasanya saya menggunakan kombinasi 3 argumen tersebut:

```bash
ls -lAh
```

Outputnya kira-kira seperti ini

```
❯ ls -lAh
total 612K
-rw-r--r--   1 lordronz lordronz  450 Jan 14 20:37 commitlint.config.js
drwxr-xr-x   7 lordronz lordronz 4.0K Jan 16 00:00 cypress
-rw-r--r--   1 lordronz lordronz    3 Jan 16 00:00 cypress.json
-rw-r--r--   1 lordronz lordronz  147 Jan 14 20:37 .editorconfig
-rw-r--r--   1 lordronz lordronz   41 Jan 14 20:37 .env.example
-rw-r--r--   1 lordronz lordronz   23 Jan 14 20:37 .eslintignore
-rw-r--r--   1 lordronz lordronz  708 Jan 14 20:37 .eslintrc.json
drwxr-xr-x   8 lordronz lordronz 4.0K Jan 16 11:34 .git
-rw-r--r--   1 lordronz lordronz 4.7K Jan 14 20:37 .gitattributes
drwxr-xr-x   3 lordronz lordronz 4.0K Jan 14 20:37 .github
-rw-r--r--   1 lordronz lordronz  499 Jan 16 00:00 .gitignore
drwxr-xr-x   3 lordronz lordronz 4.0K Jan 14 20:49 .husky
-rw-r--r--   1 lordronz lordronz 1.1K Jan 15 18:44 LICENSE
-rw-r--r--   1 lordronz lordronz  131 Jan 14 20:37 .lintstagedrc.json
drwxr-xr-x   5 lordronz lordronz 4.0K Jan 16 11:33 .next
-rw-r--r--   1 lordronz lordronz  825 Jan 16 11:32 next.config.mjs
-rw-r--r--   1 lordronz lordronz  201 Jan 14 20:37 next-env.d.ts
-rw-r--r--   1 lordronz lordronz  353 Jan 15 18:44 next-sitemap.js
drwxr-xr-x 532 lordronz lordronz  20K Jan 16 11:32 node_modules
-rw-r--r--   1 lordronz lordronz 1.8K Jan 16 11:32 package.json
-rw-r--r--   1 lordronz lordronz   83 Jan 14 20:37 postcss.config.js
-rw-r--r--   1 lordronz lordronz  392 Jan 14 20:37 .prettierignore
-rw-r--r--   1 lordronz lordronz   85 Jan 14 20:37 .prettierrc.json
drwxr-xr-x   4 lordronz lordronz 4.0K Jan 16 10:31 public
```

Linux file permission

reference >

Pada section sebelumnya apabila kita melakukan command `` `ls -l` `` maka kita dapat melihat file permission atau file mode yang mana berguna untuk meregulasi level interaksi pada file atau direktori yang dapat dilakukan oleh system. Ini mirip seperti access control pada windows, namun lebih advanced. Inilah mengapa linux sangat secure (dengan catatan penggunanya menggunakan dengan benar, tentu saja).

Pada output dari `` `ls` `` sebelumnya, terdapat kolom drwxrwxrwx, penjelasannya adalah sebagai berikut.

`` `d` ``	`` `rwx` ``	`` `rwx` ``	`` `rwx` ``
Tipe file, secara teknis bukan termasuk dari file permission, kalau regular file nilainya adalah `` `-` ``, apabila direktori maka `` `d` ``	Permission yang dimiliki oleh user pemilik file	Permission yang dimiliki oleh grup dari user	Permission yang dimiliki oleh user-user lain

Tiap-tiap dari permission `rwx` yang diatas dirincikan sebagai berikut:

	Karakter	Efek pada file	Efek pada direktori
Read permission	`-`	File tidak dapat dibuka	Isi direktori tidak dapat dilihat
Read permission	`r`	File dapat dibuka	Isi direktori dapat dilihat
Write permission	`-`	File tidak dapat dimodifikasi	Isi direktori tidak dapat dimodifikasi
Write permission	`w`	File dapat dimodifikasi	Isi direktori dapat dimodifikasi, seperti membuat file atau folder baru
Execute permission	`-`	File tidak dapat dieksekusi	Direktori tidak dapat diakses dengan command `cd`
Execute permission	`x`	File dapat dieksekusi	Direktori dapat diakses atau dibuka dengan command `cd`

> chmod

reference >

Command `chmod` berfungsi untuk mengubah permission dari sebuah file. Kita dapat menggunakan symbolic maupun numeric mode untuk mengganti permission.

Format dari syntax symbolic modenya adalah sebagai berikut.

```bash
chmod [OPTIONS] [ugoa…][-+=]perms…[,…] FILE...
```

Flag pertama yaitu `[ugoa]` menyatakan user dengan permission mana yang mesti diubah.

- `u` Pemilik file
- `g` Semua user yang ada di grup
- `o` Semua user lain yang ada di luar grup
- `a` Semua user, sama dengan `ugo`

Jika flag user ini tidak diberikan maka default adalah `a`.

Flag berikutnya adalah `[-+=]` menyatakan operasi yang akan menentukan permission apa yang akan dihapus, ditambah, atau diset.

- `-` Menghapus permission
- `+` Menambah permission
- `=` Mengubah permission menjadi permission yang diberikan. Jika tidak ada argumen permission yang diberikan, semua permission dari user yang siberikan pada flag sebelumnya akan dihapus.

Berikut beberapa contoh penggunaannya

- Berikan pemilik file permission untuk mengeksekusi

```bash
chmod u+x a.out
```

- Hapus semua permission untuk membaca file untuk semua user selain pemilik file

```bash
chmod go-r a.out
```

- Berikan semua permission ke pemilik, read permission ke grup dan tidak ada permission untuk user lain.

```bash
chmod u=rwx,g=r,o= a.out
```

Untuk numeric mode, kita menggunakan angka yang merepresentasikan masing-masing permission.

- `r` (read) = 4
- `w` (write) = 2
- `r` (execute) = 1
- `-` (no permission) = 0

Angka dari permission yang dimiliki oleh tiap kelas user direpresentasikan dengan jumlah atau *sum* dari angka permission.

Misalnya command berikut

```bash
chmod 744 a.out
```

Artinya, pemilik file memiliki permission read dan write karena 4 + 2 = 6. Lalu, grup dan user lain memiliki permission read saja.

```bash
chmod 400 ./*
```

Command diatas akan membuat semua file di dalam `pwd` menjadi memiliki permission read untuk owner saja, dan no permission untuk kelas user lain.

> ***Note:*** *Jangan sembarangan mengganti permission dari file sistem, silahkan kunjungi* ***link ini*** *untuk penjelasannya.*

> chown

reference >

Command ini digunakan untuk mengganti owner dari sebuah file atau direktori.

```bash
chown root fizzbuzz.hs
```

Command di atas akan mengganti pemilik file fizzbuzz menjadi root.

> ***Note:*** *kita perlu root privilege untuk mengeksekusi command diatas*

> touch

reference >

Command `touch` sejatinya berfungsi untuk mengupdate waktu akses dan modifikasi pada file, namun, apabila file yang dituju tidak ada, maka file tersebut akan dibuat. Inilah mengapa command ini digunakan untuk membuat file baru. Berikut contoh penggunaannya.

```bash
touch yep.cock
```

Apabila file `yep.cock` ada, maka waktu akses dan modifiednya akan diupdate, sebaliknya maka akan dibuat file kosong dengan nama tersebut.

> cat

reference >

Command con**cat**enate ini berfungsi untuk mengoutputkan isi dari file ke standard output.

Apabila argumen nama file tidak diberikan maka `cat` akan membaca dari standard input kemudian langsung mengoutputkannya

```bash
cat
```

Command berikut akan mengoutputkan isi dari `main.c`

```bash
cat main.c
```

> mkdir

reference >

Command **ma**k**e** **dir**ectory ini berguna untuk... membuat direktori

```bash
mkdir ./Documents/projects
```

Command diatas akan membuat direktori projects yang ada didalam folder Documents.

> nano

reference >

Nano merupakan text editor default dari kebanyakan distro Linux. Saya menggunakan **Arch Linux** dan nano merupakan text editor defaultnya.

> less

reference >

Command `less` digunakan untuk membaca file. Ada command yang mirip namun lebih primitif, `more`. *Less is more than more.*

> tail

reference >

Command `tail` digunakan untuk mengoutputkan *n* baris terakhir dari sebuah file. Default *n* adalah

10.

```bash
tail fizzbuzz.hs
```

Kita juga bisa menggunakan *piping*, menggabungkannya dengan command `cat`.

```bash
cat fizzbuzz.hs | tail -5
```

> head

reference >

Command `head` pada dasarnya sama saja dengan tail, namun dia mengoutputkan *n* baris pertama dari sebuah file.

```bash
head fizzbuzz.hs
```

> cp

reference >

Command `cp` digunakan untuk menyalin file ke destinasi tertentu. Syntaxnya adalah `cp [SOURCE] [DESTINATION]`.

```bash
cp fizzbuzz.pl programs
```

Command diatas akan menyalin file `fizzbuzz.pl` ke direktori programs. Kita juga bisa menyalin dan merename sekaligus.

```bash
cp fizzbuzz.pl programs/fizzbuzzer.pl
```

Ketika kita ingin menyalin semua file dalam direktori, kita bisa menggunakan *wildcard* `*`.

```bash
cp * programs
```

> mv

reference >

Command *mov*e digunakan untuk memindahkan file. Ketika kita memindah file, kita sebenarnya juga merename file tersebut. Maka dari itu `mv` juga digunakan untuk merename sebuah file.

```bash
mv script.py scrap.py
```

Jika kita ingin merename beberapa karakter pada nama file kita bisa menggunakannya seperti ini.

```bash
mv script{1,69}.py
```

Command tersebut merename `script1.py` menjadi `script69.py`

> clear

reference >

Command ini berguna untuk, membersihkan terminal.

```bash
clear
```

> file

reference >

Command `file` digunakan untuk menampilkan tipe file, contohnya seperti berikut.

```bash
file a.out
```

> rm

reference >

Command `rm` digunakan untuk menghapus file maupun direktori. Command berikut akan menghapus file `main.py`.

```bash
rm main.py
```

Lalu untuk menghapus direktori beserta isinya kita menggunakan argumen `-r`. Berikut kita menghapus folder `poggers` beserta seluruh isinya.

```bash
rm -r poggers
```

Namun kita perlu berhati-hati, karena command ini akan benar-benar menghapus file, tidak memindahkannya ke Recycle Bin seperti OS sebelah. Malahan kita bisa saja tidak sengaja menghapus file sistem.

> ***Note:*** *Perlu kehati-hatian dalam menggunakan command* `rm`*, karena user tidak akan ditanyakan konfirmasi, kecuali jika kita menggunakan flag* `-i`*. Flag seperti* `-rf` *akan menghapus file dan direktori secara rekursif tanpa ampun, sehingga akan destruktif jika targetnya adalah root* `/`*. Silahkan kunjungi* ***link ini*** *untuk informasi lebih lanjut.*

> sudo

reference >

Command **s**uper**u**ser **do** digunakan untuk menjalankan command atau mengakses file yang dimiliki oleh *root user*. Jika kita memang sudah login dengan root user maka tidak perlu menggunakan command ini. Tapi sebaiknya kita jangan menggunakan root user untuk penggunaan sehari-hari karena alasan keamanan.

```bash
sudo cat /etc/shadow
```

Apabila kita mengeksekusi command diatas dengan menggunakan user non-root tanpa `sudo`, maka kita akan mendapat error *Permission Denied*.

> history

reference >

Command `history` digunakan untuk mengoutputkan history dari penggunaan shell command yang kita lakukan. Biasanya text history ini disimpan di `~/.bash_history` jika menggunakan bash atau `~/.zsh_history` jika menggunakan zsh.

```bash
history
```

Biasanya `history` ini akan sangat panjang, jadi penggunaannya bisa dikombinasikan dengan `tail` menggunakan pipe (|).

```bash
history | tail
```

Stream redirection

reference >

Terdapat 3 kategori dalam stream redirection, masing-masing untuk standard output, standard input, dan standard error.

1. Standard Output

`>` merupakan command yang digunakan untuk redirect dari standard output menuju file.

```bash
history | tail > yo.txt
```

Output dari `history` akan difilter oleh `tail`, lalu akan diberikan ke dalam file `yo.txt`.

2. Standard Input

`<` merupakan command yang digunakan untuk redirect dari file ke standard input dari sebuah command.

```bash
less < yo.txt
```

Output dari `history` akan difilter oleh `tail`, lalu akan diberikan ke dalam file `yo.txt`.

3. Standard Error

`<` merupakan command yang digunakan untuk redirect dari file ke standard input dari sebuah command.

```bash
ls % 2> wad00.txt
```

`ls %` akan mengakibatkan error, tapi pesan error tidak dioutputkan di terminal, melainkan dituliskan di file `wad00.txt`.

Apabila kita menggunakan `>` maka kita akan melakukan overwrite pada file yang dituju. Jika tidak ingin overwrite, maka kita bisa menggunakan append dengan cara memberikan double bracket seperti `>>` untuk standard output, `<<` untuk standard input, dan `2>>` untuk standard error.

> grep

reference >

Command `grep` mencari dari file input sebuah baris yang cocok dengan *pattern* yang diberikan. Ketika baris yang cocok ditemukan, baris tersebut dicopy ke standard output(default) yang artinya baris tersebut diprint di terminal.

```bash
grep int main.c
```

Kita juga dapat menggunakan pipe

```bash
cat main.c | grep int
```

Kita juga dapat menggunakan **Regular Expression**.

```bash
grep ^int main.c
```

> find

reference >

Command `find` digunakan untuk mencari file yang ada di sistem dengan menggunakan berbagai parameter dan filter. Untuk mencari file dengan nama, kita menggunakan syntax berikut

```bash
find -name "main"
```

Perlu diketahui perintah diatas bersifat case sensitive, sehingga mencari `main` itu berbeda dengan `Main`. Untuk mencari file tanpa memedulikan case, kita menggunakan opsi `-iname`.

```bash
find -iname "main"
```

Apabila kita mau mencari file yang **tidak** mengikuti pattern tertentu, kita bisa melakukan invert pada pencarian dengan opsi `-not`.

```bash
find -not -iname "main"
```

Atau kita bisa juga menggunakan tanda seru (!), dengan catatan diberikan backslash `\` karena tanda seru merupakan karakter khusus.

```bash
find \! -iname "main"
```

Selain menggunakan nama, kita juga bisa menggunakan tipe file, dengan parameter `-type`.

Berikut adalah beberapa descriptor yang dapat kita gunakan untuk menentukan tipe file:

- `f`: regular file
- `d`: directory
- `l`: symbolic link
- `c`: character devices
- `b`: block devices

Kita dapat mencari semua file yang berakhiran `.py` dengan command berikut.

```bash
find -type f -name "*.py"
```

Selain itu masih banyak lagi tipe filter yang dapat kita lakukan, silahkan mengunjungi referensi untuk selengkapnya.

> which

reference >

Command `which` akan mencetak path lengkap dari sebuah executable atau program yang akan dijalankan apabila argumen dari command ini diinputkan di shell prompt. Command ini akan mencari executable yang diberikan sebagai argumen di direktori yang terdapat pada environment variable *PATH*. *PATH* merupakan env yang memberi tahu shell di direktori mana kita harus mencari sebuah executable.

```bash
which python
```