

Praktikum 2

Operasi Input Output

POKOK BAHASAN:

- ✓ Pipeline
- ✓ Redirection

TUJUAN BELAJAR:

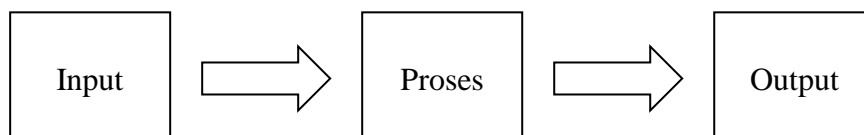
Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

- ✓ Memahami konsep proses I/O dan redirection
- ✓ Memahami standar input, output dan error
- ✓ Menggunakan notasi output, append dan here document
- ✓ Memahami konsep PIPE dan filter

DASAR TEORI:

1 PROSES I/O

Sebuah proses memerlukan Input dan Output.



Instruksi (*command*) yang diberikan pada Linux melalui Shell disebut sebagai *eksekusi program* yang selanjutnya disebut *proses*.

Setiap kali instruksi diberikan, maka Linux kernel akan menciptakan sebuah proses dengan memberikan nomor PID (*Process Identity*).

Proses dalam Linux selalu membutuhkan Input dan menghasilkan suatu Output.

Dalam konteks Linux input/output adalah :

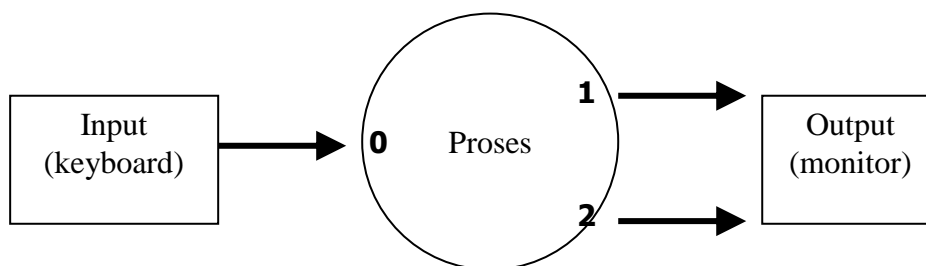
- Keyboard (input)
- Layar (output)
- Files
- Struktur data kernel
- Peralatan I/O lainnya (misalnya Network)

2 FILE DESCRIPTOR

Linux berkomunikasi dengan file melalui *file descriptor* yang direpresentasikan melalui angka yang dimulai dari 0, 1, 2 dan seterusnya.

Tiga buah file descriptor standar yang lalu diciptakan oleh proses adalah :

- 0 = keyboard (standar input)
- 1 = layar (standar output)
- 2 = layar (standar error)



Linux tidak membedakan antara peralatan hardware dan file, Linux memanipulasi peralatan hardware sama dengan file.

3 PEMBELOKAN (REDIRECTION)

Pembelokan dilakukan untuk standard input, output dan error, yaitu untuk mengalihkan file descriptor dari 0, 1 dan 2. Simbol untuk pembelokan adalah :

- 0< atau < pengganti standard input
1> atau > pengganti standard output
2>

4 PIPA (PIPELINE)

Mekanisme pipa digunakan sebagai alat komunikasi antar proses.

Input \Rightarrow Proses1 \Rightarrow Output = Input \Rightarrow Proses2 \Rightarrow Output

Proses 1 menghasilkan output yang selanjutnya digunakan sebagai input oleh Proses 2. Hubungan output input ini dinamakan pipa, yang menghubungkan Proses 1 dengan Proses2 dan dinyatakan dengan symbol “|”.

Proses1 | Proses2

5 FILTER

Filter adalah utilitas Linux yang dapat memproses standard input (dari keyboard) dan menampilkan hasilnya pada standard output (layar). Contoh filter adalah `cat`, `sort`, `grep`, `pr`, `head`, `tail`, `paste` dan lainnya.

Pada sebuah rangkaian pipa :

$P_1 | P_2 | P_3 \dots\dots | P_{n-1} | P_n$

Maka P_2 sampai dengan P_{n-1} mutlak harus utilitas Linux yang berfungsi sebagai filter. P_1 (awal) dan P_n (terakhir) boleh tidak filter. Utilitas yang bukan filter misalnya `who`, `ls`, `ps`, `lp`, `lpr`, `mail` dan lainnya.

Beberapa perintah Linux yang digunakan untuk proses penyaringan antara lain :

- Perintah `grep`
Digunakan untuk menyaring masukannya dan menampilkan baris-baris yang hanya mengandung pola yang ditentukan. Pola ini disebut *regular expression*.
- Perintah `wc`
Digunakan untuk menghitung jumlah baris, kata dan karakter dari baris-baris masukan yang diberikan kepadanya. Untuk mengetahui berapa baris gunakan option `-l`, untuk mengetahui berapa kata, gunakan option `-w` dan untuk mengetahui berapa karakter, gunakan option `-c`. Jika salah satu option tidak digunakan, maka tampilannya adalah jumlah baris, jumlah kata dan jumlah karakter.

- Perintah `sort`
Digunakan untuk mengurutkan masukannya berdasarkan urutan nomor ASCII dari karakter.
- Perintah `cut`
Digunakan untuk mengambil kolom tertentu dari baris-baris masukannya, yang ditentukan pada option `-c`.
- Perintah `uniq`
Digunakan untuk menghilangkan baris-baris berurutan yang mengalami duplikasi, biasanya digabungkan dalam pipeline dengan `sort`.

TUGAS PENDAHULUAN:

Jawablah pertanyaan-pertanyaan di bawah ini :

1. Apa yang dimaksud *redirection* ?
2. Apa yang dimaksud *pipeline* ?
3. Apa yang dimaksud perintah di bawah ini :
`echo, cat, more, sort, grep, wc, cut, uniq`

PERCOBAAN:

1. Login sebagai user.
2. Bukalah Console Terminal dan lakukan percobaan-percobaan di bawah ini.
Perhatikan hasil setiap percobaan.
3. Selesaikan soal-soal latihan.

Percobaan 1 : File descriptor

1. Output ke layar (standar output), input dari system (kernel)

```
$ ps
```

2. Output ke layar (standar output), input dari keyboard (standard input)

```
$ cat
hallo, apa khabar
hallo, apa khabar
exit dengan ^d
exit dengan ^d
[Ctrl-d]
```

3. Input nama direktori, output tidak ada (membuat direktori baru), bila terjadi error maka tampilan error pada layar (standard error)

```
$ mkdir mydir
$ mkdir mydir      (Terdapat pesan error)
```

Percobaan 2 : Pembelokan (*redirection*)

1. Pembelokan standar output

```
$ cat 1> myfile.txt
Ini adalah teks yang saya simpan
Ke file myfile.txt
```

2. Pembelokan standar input, yaitu input dibelokkan dari keyboard menjadi dari file

```
$ cat 0< myfile.txt
$ cat myfile.txt
```

3. Pembelokan standar error untuk disimpan di file

```
$ mkdir mydir      (Terdapat pesan error)
$ mkdir mydir 2> myerror.txt
$ cat myerror.txt
```

4. Notasi `2>&1` : pembelokan standar error (`2>`) adalah identik dengan file descriptor 1.

```
$ ls filebaru          (Terdapat pesan error)
$ ls filebaru 2> out.txt
$ cat out.txt
$ ls filebaru 2> out.txt 2>&1
$ cat out.txt
```

5. Notasi `1>&2` (atau `>&2`) : pembelokan standar output adalah sama dengan file descriptor 2 yaitu standar error

```
$ echo "mencoba menulis file" 1> baru
$ cat filebaru 2> baru 1>&2
$ cat baru
```

6. Notasi `>>` (append)

```
$ echo "kata pertama" > surat
$ echo "kata kedua" >> surat
$ echo "kata ketiga" >> surat
$ cat surat
$ echo "kata keempat" > surat
$ cat surat
```

7. Notasi here document (`<<++ ++`) digunakan sebagai pembatas input dari keyboard. Perhatikan bahwa tanda pembatas dapat digantikan dengan tanda apa saja, namun harus sama dan tanda penutup harus diberikan pada awal baris

```
$ cat <<++
Hallo, apa kabar ?
Baik-baik saja ?
Ok!
++
$ cat <<%%
Hallo, apa kabar ?
Baik-baik saja ?
Ok!
%%
```

8. Notasi `-` (input keyboard) adalah representan input dari keyboard. Artinya menampilkan file 1, kemudian menampilkan input dari keyboard dan menampilkan file 2. Perhatikan bahwa notasi `"-"` berarti menyelipkan input dari keyboard

```
$ cat myfile.txt - surat
```

9. Untuk membelokkan standart output ke file, digunakan operator >

```
$ echo hello
$ echo hello > output
$ cat output
```

10. Untuk menambahkan output ke file digunakan operator >>

```
$ echo bye >> output
$ cat output
```

11. Untuk membelokkan standart input digunakan operator <

```
$ cat < output
```

12. Pembelokan standart input dan standart output dapat dikombinasikan tetapi tidak boleh menggunakan nama file yang sama sebagai standart input dan output.

```
$ cat < output > out
$ cat out
$ cat < output >> out
$ cat out
$ cat < output > output
$ cat output
$ cat < out >> out          (Proses tidak berhenti)
[Ctrl-c]
$ cat out
```

Percobaan 3 : Pipa (*pipeline*)

1. Operator pipa (|) digunakan untuk membuat eksekusi proses dengan melewati data langsung ke data lainnya.

```
$ who
$ who | sort
$ who | sort -r
$ who > tmp
$ sort tmp
$ rm tmp
$ ls -l /etc | more
$ ls -l /etc | sort | more
```

Percobaan 4 : Filter

1. Pipa juga digunakan untuk mengkombinasikan utilitas sistem untuk membentuk fungsi yang lebih kompleks

```
$ w -h | grep <user>
$ grep <user> /etc/passwd
$ ls /etc | wc
$ ls /etc | wc -l
$ cat > kelas1.txt
Badu
Zulkifli
Yulizir
Yudi
Ade
[Ctrl-d]
$ cat > kelas2.txt
Budi
Gama
Asep
Muchlis
[Ctrl-d]
$ cat kelas1.txt kelas2.txt | sort
$ cat kelas1.txt kelas2.txt > kelas.txt
$ cat kelas.txt | sort | uniq
```

LATIHAN:

1. Lihat daftar secara lengkap pada direktori aktif, belokkan tampilan standard output ke file baru.
2. Lihat daftar secara lengkap pada direktori /etc/passwd, belokkan tampilan standard output ke file baru tanpa menghapus file baru sebelumnya.
3. Urutkan file baru dengan cara membelokkan standard input.
4. Urutkan file baru dengan cara membelokkan standard input dan standard output ke file baru.urut.
5. Buatlah direktori latihan2 sebanyak 2 kali dan belokkan standard error ke file rmdirerror.txt.
6. Urutkan kalimat berikut :
Jakarta
Bandung
Surabaya
Padang

Palembang

Lampung

Dengan menggunakan notasi here document (<@@@ ...@@@)

7. Hitung jumlah baris, kata dan karakter dari file baru.urut dengan menggunakan filter dan tambahkan data tersebut ke file baru.
8. Gunakan perintah di bawah ini dan perhatikan hasilnya.

```
$ cat > hello.txt
dog cat
cat duck
dog chicken
chicken duck
chicken cat
dog duck
[Ctrl-d]
$ cat hello.txt | sort | uniq
$ cat hello.txt | grep "dog" | grep -v "cat"
```

LAPORAN RESMI:

1. Analisa hasil percobaan 1 sampai dengan 4, untuk setiap perintah jelaskan tampilannya.
2. Kerjakan latihan diatas dan analisa hasilnya
3. Berikan kesimpulan dari praktikum ini.