

## FILE, DIREKTORI, DAN PENGARSIPAN

### OBJEKTIF :

1. Mahasiswa Mampu Membuat, Memindahkan, dan Menghapus, File dan Direktori Menggunakan Command Line
  2. Mahasiswa Mampu Mengarsipkan File Menggunakan Command Line
- 

### PENDAHULUAN

Saat bekerja di Sistem Operasi Linux, penting untuk kita mengetahui cara memanipulasi file dan direktori. Beberapa distribusi Linux memiliki aplikasi berbasis GUI yang memungkinkan kita untuk mengelola file, tetapi bermanfaat untuk mengetahui cara melakukan operasi ini melalui baris perintah.

Penting untuk dicatat bahwa semua yang ada di Linux , huruf besar/kecil memiliki fungsi yang berbeda, fitur inilah yang dibawa dari Unix. Ini berarti bahwa shell mengenali **a** sebagai karakter huruf kecil , berbeda dari **A** sebagai karakter huruf besar. Saat memanipulasi file, perlu diperhatikan kapitalisasi: file `hello.txt` berbeda dari `HELLO.txt` dan `Hello.txt`.

*Pengarsipan file* digunakan ketika satu atau lebih file perlu dikirim atau disimpan seefisien mungkin. Ada dua aspek fundamental yang akan dibahas dalam bab ini:

- *Pengarsipan* : Menggabungkan beberapa file menjadi satu, yang menghilangkan overhead dalam file individual dan membuat file lebih mudah dikirim.
- *Kompresi* : Membuat file lebih kecil dengan menghapus informasi yang berlebihan.

Ketika arsip *didekompresi*, dan satu atau lebih file diekstraksi, ini disebut *un-archiving*.

Meskipun ruang *disk* relatif murah, pengarsipan dan kompresi tetap memiliki nilai:

- Saat menyediakan file dalam jumlah besar, seperti kode sumber ke aplikasi atau kumpulan dokumen, lebih mudah bagi orang untuk mengunduh arsip terkompresi daripada mengunduh file satu per satu.
- File log memiliki kebiasaan mengisi *disk*, jadi akan sangat membantu jika Anda membaginya berdasarkan tanggal dan mengompres versi yang lebih lama.
- Saat mencadangkan direktori, lebih mudah untuk menyimpan semuanya dalam satu arsip daripada versi (perbarui) setiap file.
- Beberapa perangkat streaming seperti kaset bekerja lebih baik jika Anda mengirim aliran data daripada file individual.
- Sering kali lebih cepat untuk memampatkan file sebelum mengirimnya ke *tape drive* atau melalui jaringan yang lebih lambat dan mendekompresinya di ujung lain daripada mengirimnya tanpa kompresi. Sangat penting bagi *administrator* Linux untuk terbiasa dengan alat untuk pengarsipan dan kompresi file.

## 6.1 GLOBBING CHARACTERS

Karakter *Glob* sering disebut sebagai *wild card*. Ini adalah karakter simbol yang memiliki arti khusus pada *shell*.

*Glob* sangat berguna karena memungkinkan Anda menentukan pola yang cocok dengan nama file dalam direktori. Jadi, alih-alih memanipulasi satu file pada satu waktu, Anda dapat dengan mudah menjalankan perintah yang memengaruhi banyak file. Misalnya, dengan menggunakan karakter *glob*, dimungkinkan untuk memanipulasi semua file dengan ekstensi tertentu atau dengan panjang nama file tertentu.

Tidak seperti perintah yang dijalankan *shell*, atau option dan argumen yang diteruskan *shell* ke perintah, karakter *glob* di interpretasikan oleh shell itu sendiri sebelum mencoba menjalankan perintah apa pun. Hasilnya, karakter *glob* dapat digunakan dengan perintah apa pun.

Contoh yang diberikan dalam bab ini menggunakan perintah `echo` untuk demonstrasi.

### 6.1.1 KARAKTER ASTERISK

Karakter *asterisk* atau karakter bintang “\*” digunakan untuk mewakili nol atau lebih dari karakter apapun dalam nama file. Misalnya, untuk menampilkan semua file di direktori `/etc` yang diawali dengan huruf `t`:

```
sysadmin@localhost:~$ echo /etc/t*
/etc/terminfo /etc/timezone /etc/tmpfiles.d
```

Pola “`t*`” cocok dengan file apa pun di direktori `/etc` yang dimulai dengan karakter `t` yang diikuti oleh nol atau lebih dari karakter apa pun. Dengan kata lain, file apa pun yang dimulai dengan huruf `t`.

Kita dapat menggunakan karakter *asterisk* di sembarang tempat dalam pola nama file. Misalnya, berikut ini cocok dengan semua nama file di direktori `/etc` yang diakhiri dengan `.d`:

```
sysadmin@localhost:~$ echo /etc/*.d
/etc/apparmor.d /etc/binfmt.d /etc/cron.d /etc/depmod.d /etc/init.d /etc/insserv
.conf.d /etc/ld.so.conf.d /etc/logrotate.d /etc/modprobe.d /etc/modules-load.d /
etc/pam.d /etc/profile.d /etc/rc0.d /etc/rc1.d /etc/rc2.d /etc/rc3.d /etc/rc4.d
/etc/rc5.d /etc/rc6.d /etc/rcS.d /etc/rsyslog.d /etc/sudoers.d /etc/sysctl.d /et
c/tmpfiles.d /etc/update-motd.d
```

Dalam contoh berikutnya, semua file dalam direktori `/etc` yang dimulai dengan huruf `r` dan diakhiri dengan `.conf` ditampilkan:

```
sysadmin @ localhost : ~ $ echo /etc/r*.conf
/etc/resolv.conf /etc/rsyslog.conf
```

### 6.1.2 KARAKTER TANDA TANYA

Karakter tanda tanya “?” mewakili satu karakter. Setiap karakter tanda tanya sama persis dengan satu karakter, tidak lebih dan tidak kurang. Misalkan

Anda ingin menampilkan semua file dalam direktori /etc yang dimulai dengan huruf t dan memiliki tepat 7 karakter setelah karakter t:

```
sysadmin@localhost:~$ echo /etc/t???????
/etc/terminfo /etc/timezone
```

Karakter *Glob* dapat digunakan bersama untuk menemukan pola yang lebih kompleks. Pola perintah /etc/\*???????????????????? hanya mencocokkan file di direktori /etc dengan dua puluh atau lebih karakter dalam nama file:

```
sysadmin@localhost:~$ echo /etc/*?????????????????????
/etc/bindresvport.blacklist /etc/ca-certificates.conf
```

Tanda bintang dan tanda tanya juga dapat digunakan bersama untuk mencari file dengan ekstensi tiga huruf dengan menggunakan pola /etc/\*.???:

```
sysadmin@localhost:~$ echo /etc/*.??
/etc/issue.net /etc/locale.gen
```

### 6.1.3 KARAKTER BRACKET

Karakter bracket “[ ]” digunakan untuk mencocokkan satu karakter dengan mewakili berbagai karakter yang mungkin cocok dengan karakter. Misalnya, pola /etc/ [gu] \* akan cocok dengan file apa pun yang dimulai dengan karakter g atau u dan berisi nol atau lebih karakter tambahan:

```
sysadmin@localhost:~$ echo /etc/[gu]*
/etc/gai.conf /etc/groff /etc/group /etc/group- /etc/gshadow /etc/gshadow- /etc/
gss /etc/ucf.conf /etc/udev /etc/ufw /etc/update-motd.d /etc/updatedb.conf
```

Tanda kurung juga dapat digunakan untuk mewakili berbagai karakter. Misalnya, pola /etc/ [a-d] \* akan cocok dengan semua file yang dimulai dengan huruf apa pun antara a dan d:

```
sysadmin@localhost:~$ echo /etc/[a-d]*
/etc/adduser.conf /etc/alternatives /etc/apparmor /etc/apparmor.d /etc/apt /etc/
bash.bashrc /etc/bind /etc/bindresvport.blacklist /etc/binfmt.d /etc/ca-certific
ates /etc/ca-certificates.conf /etc/calendar /etc/console-setup /etc/cron.d /etc
/cron.daily /etc/cron.hourly /etc/cron.monthly /etc/cron.weekly /etc/crontab /et
c/dbus-1 /etc/debconf.conf /etc/debian_version /etc/default /etc/deluser.conf /e
tc/depmod.d /etc/dhcp /etc/dpkg
```

Pola `/etc/*[0-9]*` akan menampilkan file yang berisi setidaknya satu nomor:

```
sysadmin@localhost:~$ echo /etc/*[0-9]*
/etc/X11 /etc/dbus-1 /etc/hostname /etc/mke2fs.conf /etc/python3 /etc/python3.6
/etc/rc0.d /etc/rc1.d /etc/rc2.d /etc/rc3.d /etc/rc4.d /etc/rc5.d /etc/rc6.d
```

Rentang ini didasarkan pada tabel teks **ASCII**. Tabel ini mendefinisikan daftar karakter, mengurnya dalam urutan standar tertentu. Jika pesanan yang tidak valid diberikan, tidak ada kecocokan yang akan dibuat:

```
sysadmin@localhost:~$ echo /etc/*[9-0]*
/etc/*[9-0]*
```

#### 6.1.4 KARAKTER TANDA SERU

Karakter tanda seru ! digunakan bersama dengan tanda kurung siku untuk meniadakan rentang. Misalnya, pola `/etc/[!DP]*` akan cocok dengan file apapun yang *tidak* dimulai dengan **D** atau **P**.

```
sysadmin@localhost:~$ echo /etc/[!a-t]*
/etc/X11 /etc/ucf.conf /etc/udev /etc/ufw /etc/update-motd.d /etc/updatedb.conf
/etc/vim /etc/vtrgb /etc/wgetrc /etc/xdg
```

#### 6.1.5 LISTING MENGGUNAKAN GLOBS

*Command ls* biasanya digunakan untuk membuat daftar file dalam sebuah direktori; Akibatnya, menggunakan perintah `echo` mungkin tampak seperti pilihan yang aneh. Namun, ada sesuatu tentang *command ls* yang menyebabkan masalah saat membuat daftar file menggunakan pola *glob*.

Inginlah bahwa ini adalah *shell*, bukan *command echo* atau *ls*, yang memperluas pola *glob* menjadi nama file yang sesuai. Dengan kata lain, jika *command echo /etc/a\** dijalankan, apa yang dilakukan *shell* sebelum menjalankan *command echo* diganti *a\** dengan semua file dan direktori di dalam direktori */etc* yang sesuai dengan pola.

Jadi, jika *command ls /etc/a\** dijalankan, apa yang benar-benar dijalankan oleh *shell* adalah seperti ini:

```
ls /etc/adduser.conf /etc/alternatives /etc/apparmor /etc/apparmor.d /etc/apt
```

Saat *command* `ls` melihat beberapa argumen, *command* tersebut melakukan operasi daftar pada setiap item secara terpisah. Dengan kata lain, `ls /etc/a*` sama seperti menjalankan perintah berikut secara berurutan:

```
ls /etc/adduser.conf  
ls /etc/alternatives  
ls /etc/apparmor  
ls /etc/apparmor.d  
ls /etc/apt
```

Sekarang pertimbangkan apa yang terjadi jika perintah `ls` melewati file, seperti `/etc/adduser.conf`:

```
sysadmin@localhost:~$ ls /etc/adduser.conf  
/etc/adduser.conf
```

Menjalankan *command* `ls` pada satu file menghasilkan nama file yang akan dicetak; biasanya ini berguna jika option `-l` digunakan untuk melihat detail tentang file tertentu:

```
sysadmin@localhost:~$ ls -l /etc/adduser.conf  
-rw-r--r-- 1 root root 3028 May 26 2018 /etc/adduser.conf
```

Namun, bagaimana jika *command* `ls` tersebut diberi nama direktori sebagai argumen? Dalam hal ini, output dari perintah berbeda jika argumennya adalah file biasa:

```
sysadmin@localhost:~$ ls /etc/apparmor  
init parser.conf subdomain.conf
```

Jika *command* `ls` diberi nama direktori, perintah tersebut akan menampilkan *isi* direktori (nama file di direktori), bukan hanya nama direktori. Nama file pada contoh sebelumnya adalah nama file dalam direktori `/etc/apparmor`.

Mengapa ini menjadi masalah saat menggunakan *glob*? Perhatikan output berikut ini:

```
sysadmin@localhost:~$ ls /etc/ap*
/etc/apparmor:
init parser.conf subdomain.conf

/etc/apparmor.d:
abstractions disable local tunables usr.sbin.named
cache force-complain sbin.dhclient usr.bin.man usr.sbin.rsyslogd

/etc/apt:
apt.conf.d preferences.d sources.list sources.list.d trusted.gpg.d
```

Ketika *command* `ls` melihat nama file sebagai argumen, itu hanya menampilkan nama file. Namun, untuk direktori mana pun, ini menampilkan isi direktori, bukan hanya nama direktori.

Ini menjadi lebih membingungkan dalam situasi seperti berikut:

```
sysadmin@localhost:~$ ls /etc/x*
autostart systemd user-dirs.conf user-dirs.defaults
```

Pada contoh sebelumnya, sepertinya *perintah* `ls` salah. Namun, yang sebenarnya terjadi adalah satu-satunya hal yang cocok dengan glob `/etc/x*` adalah `/etc/xdg` direktorinya.

Jadi, perintah `ls` hanya menampilkan file di direktori itu!

Ada solusi sederhana untuk masalah ini: selalu gunakan option `-d` dengan *glob*, yang memberi tahu *command* `ls` untuk menampilkan nama direktori, bukan isinya:

```
sysadmin@localhost:~$ ls -d /etc/x*
/etc/xdg
```

## 6.2 MENYALIN FILE

Untuk menyalin file pada linux menggunakan *command* `cp`. Itu membutuhkan sumber dan tujuan. Struktur perintahnya adalah sebagai berikut:

```
cp [source] [destination]
```

*Source* adalah file yang akan disalin. *Destination* adalah di mana salinan tersebut berada. Saat berhasil, perintah `cp` tidak memiliki keluaran (**tidak ada berita adalah kabar baik**). Perintah berikut menyalin `/etc/hostsfile` ke direktori *home* Anda:

```
sysadmin@localhost:~$ cp /etc/hosts ~  
sysadmin@localhost:~$ ls  
Desktop Downloads Pictures Templates hosts  
Documents Music Public Videos
```

**Catatan:** karakter “~” mewakili direktori home Anda.

### 6.2.1 MODE VERBOSE

Option `-v` menyebabkan *command cp* menghasilkan output jika berhasil. Option `-v` merupakan singkatan *verbose*:

```
sysadmin@localhost:~$ cp -v /etc/hosts ~  
`/etc/hosts' -> `/home/sysadmin/hosts'
```

Jika tujuannya adalah direktori, file baru yang dihasilkan tetap menggunakan nama yang sama dengan file aslinya. Untuk memberi file baru nama yang berbeda, berikan nama baru sebagai bagian dari tujuan:

```
sysadmin@localhost:~$ cp /etc/hosts ~/hosts.copy  
sysadmin@localhost:~$ ls  
Desktop Downloads Pictures Templates hosts  
Documents Music Public Videos hosts.copy
```

### 6.2.2 MENGHINDARI OVERWRITE DATA

*Command cp* dapat merusak data yang ada jika file tujuan sudah ada. Dalam kasus di mana file tujuan ada, *command cp* menimpa konten file yang ada dengan konten file sumber.

Untuk menggambarkan masalah potensial ini, pertama-tama file baru dibuat di direktori home dengan menyalin file yang sudah ada:

```
sysadmin@localhost:~$ cp /etc/hostname example.txt
```

Lihat informasi tentang file dengan *command ls*:

```
sysadmin@localhost:~$ ls -l example.txt  
-rw-r--r-- 1 sysadmin sysadmin 10 Dec 15 22:55 example.txt
```

Lihat konten file menggunakan *command cat*:

```
sysadmin@localhost:~$ cat example.txt  
localhost
```

Dalam contoh berikutnya, *command cp* tersebut menghancurkan konten asli file `example.txt`:

```
sysadmin@localhost:~$ cp /etc/timezone example.txt
```

Perhatikan bahwa setelah *command* `cp` selesai, ukuran file telah berubah dan isinya berbeda:

```
sysadmin@localhost:~$ ls -l example.txt
-rw-r--r-- 1 sysadmin sysadmin 8 Dec 15 22:58 example.txt
sysadmin@localhost:~$ cat example.txt
Etc/UTC
```

Dua option dapat digunakan untuk melindungi dari penimpaan yang tidak disengaja. Dengan option `-i` (*interactive*), perintah `cp` meminta konfirmasi pengguna sebelum menimpa file. Contoh berikut menunjukkan option ini, pertama-tama memulihkan konten file asli:

```
sysadmin@localhost:~$ cp -i /etc/hosts example.txt
cp: overwrite '/home/sysadmin/example.txt'? n
```

Jika nilai `y` (yes) diberikan, maka proses penyalinan akan berlangsung. Namun, nilai `n` (no) diberikan saat diminta untuk menimpa file, jadi tidak ada perubahan yang dilakukan pada file.

Option `-i` mengharuskan Anda untuk menjawab `y` atau `n`, untuk setiap salinan yang bisa menimpa isi file yang ada. Hal ini bisa menjadi sulit ketika banyak penimpaan terjadi, seperti contoh yang ditunjukkan di bawah ini:

```
sysadmin@localhost:~$ cp -i /etc/skel/* ~
cp: -r not specified; omitting directory '/etc/skel/.'
cp: -r not specified; omitting directory '/etc/skel/..'
cp: overwrite '/home/sysadmin/.bash_logout'? n
cp: overwrite '/home/sysadmin/.bashrc'? n
cp: overwrite '/home/sysadmin/.profile'? n
cp: overwrite '/home/sysadmin/.selected_editor'? n
```

Seperti yang Anda lihat dari contoh di atas, *command* `cp` mencoba menimpa empat file yang ada, memaksa pengguna untuk menjawab empat petunjuk. Jika situasi ini terjadi pada 100 file, ini bisa menjadi sangat mengganggu.

Untuk menjawab `n` setiap prompt secara otomatis, gunakan option `-n`. Ini singkatan dari *no clobber*, atau *no overwrite*.

```
sysadmin@localhost:~$ cp -n /etc/skel/* ~
cp: -r not specified; omitting directory '/etc/skel/.'
cp: -r not specified; omitting directory '/etc/skel/..'
```

### 6.2.3 MENYALIN DIREKTORI

Secara default, perintah `cp` tidak akan menyalin direktori; segala upaya untuk melakukannya menghasilkan pesan kesalahan:

```
sysadmin@localhost:~$ cp -n /etc/skel/.* ~  
cp: -r not specified; omitting directory '/etc/skel/.'  
cp: -r not specified; omitting directory '/etc/skel/..'
```

Namun, perintah `cp` memiliki option `-r` untuk menyalin file dan direktori.

Hati-hati dengan opsi ini. Seluruh struktur direktori akan disalin yang dapat mengakibatkan penyalinan banyak file dan direktori!

## 6.3 MEMINDAHKAN FILE

Untuk memindahkan file, gunakan *command* `mv`. Sintaks untuk perintah `mv` ini mirip dengan perintah `cp`:

```
mv [source] [destination]
```

Dalam contoh berikut, file `hosts` yang dibuat sebelumnya dipindahkan dari direktori saat ini ke direktori `Videos`:

```
sysadmin@localhost:~$ ls  
Desktop Downloads Pictures Templates example.txt hosts.copy  
Documents Music Public Videos hosts  
sysadmin@localhost:~$ mv hosts Videos  
sysadmin@localhost:~$ ls  
Desktop Downloads Pictures Templates example.txt  
Documents Music Public Videos hosts.copy  
sysadmin@localhost:~$ ls Videos  
hosts
```

Ketika sebuah file dipindahkan, file tersebut dihapus dari lokasi aslinya dan ditempatkan di lokasi baru. Memindahkan file bisa jadi agak rumit di Linux karena pengguna memerlukan izin khusus untuk menghapus file dari direktori. Tanpa izin yang tepat, pesan kesalahan *Permission denied* dikembalikan:

```
sysadmin@localhost:~$ mv /etc/hosts .  
mv: cannot move '/etc/hosts' to `./hosts': Permission denied
```

### 6.3.1 MENGUBAH NAMA FILE

*Command* `mv` tidak hanya digunakan untuk memindahkan file, tetapi juga untuk mengganti nama file. Jika tujuan dari *command* `mv` tersebut adalah

direktori, file tersebut dipindahkan ke direktori yang ditentukan. Nama file hanya berubah jika nama file tujuan juga ditentukan.

```
sysadmin@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  example.txt
Documents  Music     Public    Videos     hosts.copy
sysadmin@localhost:~$ mv example.txt Videos/newexample.txt
sysadmin@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  hosts.copy
Documents  Music     Public    Videos
sysadmin@localhost:~$ ls Videos
hosts  newexample.txt
```

Jika direktori tujuan tidak ditentukan, nama file akan diganti menggunakan nama file tujuan dan tetap berada di direktori sumber. Misalnya, perintah berikut mengganti nama file newexample.txt menjadi myfile.txt:

```
sysadmin@localhost:~$ cd Videos
sysadmin@localhost:~/Videos$ ls
hosts  newexample.txt
sysadmin@localhost:~/Videos$ mv newexample.txt myfile.txt
sysadmin@localhost:~/Videos$ ls
hosts  myfile.txt
```

*Command mv pada contoh sebelumnya yang berarti memindahkan file newexample.txt dari direktori saat ini ke direktori saat ini dan beri nama file baru myfile.txt*

### 6.3.2 ADDITIONAL MOVE OPTIONS

Seperti command cp, command mv tersebut menyediakan option berikut:

Pilihan	Berarti
-i	Interaktif : Tanyakan apakah file akan ditimpas.
-n	No Clobber : Jangan menimpa konten file tujuan.
-v	Verbose : Tunjukkan gerakan yang dihasilkan.

**Penting:** Tidak ada `-r` opsi karena `mv` perintah memindahkan direktori secara default.

## 6.4 MEMBUAT FILE BARU

Ada beberapa cara untuk membuat file baru, termasuk menggunakan program yang dirancang untuk mengedit file (*text editor*).

Ada juga cara membuat file kosong yang bisa diisi data di lain waktu. Fitur ini berguna untuk beberapa sistem operasi karena keberadaan file dapat mengubah cara kerja perintah atau layanan. Ini juga berguna untuk membuat file sebagai "*placeholder*" untuk mengingatkan Anda untuk membuat konten file di lain waktu.

Untuk membuat file kosong, gunakan *command touch* seperti yang ditunjukkan di bawah ini:

```
sysadmin@localhost:~$ touch sample
sysadmin@localhost:~$ ls -l sample
-rw-rw-r-- 1 sysadmin sysadmin 0 Nov  9 16:48 sample
```

Perhatikan, ukuran file baru adalah 0 byte. Seperti yang disebutkan sebelumnya, *perintah touch* tidak menempatkan data apa pun di dalam file baru.

## 6.5 MENGHAPUS FILE

Untuk menghapus file, gunakan *command rm*:

```
sysadmin@localhost:~$ ls
Desktop  Downloads  Pictures  Templates  hosts.copy
Documents  Music      Public    Videos     sample
sysadmin@localhost:~$ rm sample
sysadmin@localhost:~$ rm hosts.copy
sysadmin@localhost:~$ ls
Desktop  Documents  Downloads  Music      Pictures  Public  Templates  Videos
```

Perhatikan bahwa file telah dihapus tanpa ada pertanyaan. Ini dapat menyebabkan masalah saat menghapus banyak file dengan menggunakan karakter *glob*. Karena file-file ini dihapus tanpa pertanyaan, pengguna bisa jadi akhirnya menghapus file yang tidak dimaksudkan untuk dihapus.

**Peringatan:** File-file tersebut dihapus secara permanen. Tidak ada perintah untuk membatalkan penghapusan file dan tidak ada tempat sampah untuk memulihkan file yang dihapus.

Sebagai tindakan pencegahan, pengguna harus menggunakan option *-i* saat menghapus beberapa file:

```
sysadmin@localhost:~$ touch sample.txt example.txt test.txt
sysadmin@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  example.txt  test.txt
Documents  Music     Public    Videos     sample.txt
sysadmin@localhost:~$ rm -i *.txt
rm: remove regular empty file `example.txt'? y
rm: remove regular empty file `sample.txt'? n
rm: remove regular empty file `test.txt'? y
sysadmin@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  sample.txt
Documents  Music     Public    Videos
```

### 6.5.1 MENGHAPUS DIREKTORI

Anda dapat menghapus direktori menggunakan *command* `rm`. Namun, apabila default (tanpa option) dari *command* `rm` tersebut tidak dapat menghapus direktori:

```
sysadmin@localhost:~$ rm Videos
rm: cannot remove 'Videos': Is a directory
```

Untuk menghapus direktori dengan perintah `rm`, gunakan option `-r` (*recursive*):

```
sysadmin@localhost:~$ ls
Desktop    Downloads  Pictures  Templates  sample.txt
Documents  Music     Public    Videos
sysadmin@localhost:~$ rm -r Videos
sysadmin@localhost:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  sample.txt
```

Ketika pengguna menghapus direktori, semua file dan subdirektori dihapus tanpa pertanyaan interaktif. Cara terbaik adalah menggunakan `-i` opsi dengan `rm` perintah.

Anda juga dapat menghapus direktori dengan *command* `rmdir`, tetapi hanya jika direktori tersebut kosong.

```
sysadmin@localhost:~$ rmdir Documents
rmdir: failed to remove 'Documents': Directory not empty
```

### 6.6 MEMBUAT DIREKTORI

Untuk membuat direktori, gunakan *command* `mkdir`:

```
sysadmin@localhost:~$ ls
Desktop Documents Downloads Music Pictures Public Templates sample.txt
sysadmin@localhost:~$ mkdir test
sysadmin@localhost:~$ ls
Desktop Downloads Pictures Templates test
Documents Music Public sample.txt
```

## 6.7 KOMPRESI FILE

Kompresi adalah mengurangi jumlah data yang diperlukan untuk menyimpan atau mengirimkan file sambil menyimpannya sedemikian rupa sehingga file tersebut dapat dipulihkan. File dengan teks yang dapat dibaca manusia mungkin sering menggunakan kata-kata yang diganti dengan sesuatu yang lebih kecil, atau gambar dengan latar belakang yang solid mungkin menunjukkan tambalan warna tersebut dengan sebuah kode. Versi file yang dikompresi biasanya tidak dilihat atau digunakan, melainkan didekompresi sebelum digunakan.

Algoritma *compressing* adalah prosedur penggunaan komputer untuk menyandikan file asli, dan sebagai hasilnya, membuatnya lebih kecil. Ilmuwan komputer meneliti algoritma ini dan menghasilkan algoritma yang lebih baik yang dapat bekerja lebih cepat atau memperkecil file masukan.

Saat berbicara tentang kompresi, ada dua jenis:

- *Lossless* : Tidak ada informasi yang dihapus dari file. Mengompresi file dan mendekompresinya meninggalkan sesuatu yang identik dengan aslinya.
- *Lossy* : Informasi mungkin dihapus dari file. Ini di kompresi sedemikian rupa sehingga file yang tidak di kompresi akan menghasilkan file yang sedikit berbeda dari aslinya. Misalnya, gambar dengan dua warna hijau yang sedikit berbeda dapat dibuat lebih kecil dengan memperlakukan kedua warna tersebut sebagai sama. Seringkali, mata tidak bisa membedakannya.

Secara umum, mata dan telinga manusia tidak melihat adanya sedikit ketidak sempurnaan dalam gambar dan audio, terutama saat ditampilkan di monitor atau diputar di speaker. Kompresi *lossy* sering menguntungkan media

karena menghasilkan ukuran file yang lebih kecil dan orang tidak dapat membedakan antara yang asli dan versi dengan data yang diubah. Untuk hal-hal yang harus tetap utuh, seperti dokumen, *log*, dan perangkat lunak, Anda memerlukan kompresi *lossless*.

Sebagian besar format gambar, seperti GIF, PNG, dan JPEG, menerapkan beberapa bentuk kompresi *lossy*. Anda biasanya dapat memutuskan seberapa banyak kualitas yang ingin Anda pertahankan. Kualitas yang lebih rendah menghasilkan file yang lebih kecil, tetapi setelah dekompresi, Anda mungkin melihat artefak seperti tepi kasar atau perubahan warna. Kualitas tinggi akan terlihat seperti gambar aslinya, tetapi ukuran file akan lebih mendekati aslinya.

Mengompresi file yang sudah dikompresi tidak akan membuatnya lebih kecil. Fakta ini sering dilupakan dalam hal gambar karena sudah disimpan dalam format terkompresi. Dengan kompresi *lossless*, kompresi *multiple* ini tidak menjadi masalah, tetapi jika Anda memampatkan dan mendekompresi file beberapa kali menggunakan algoritma *lossy*, pada akhirnya Anda akan memiliki sesuatu yang tidak dapat dikenali.

Linux menyediakan beberapa alat untuk mengompres file; yang paling umum adalah `gzip`. Di sini kami menampilkan file sebelum dan sesudah kompresi:

```
sysadmin@localhost:~$ cd Documents
sysadmin@localhost:~/Documents$ ls -l longfile*
-rw-r--r-- 1 sysadmin sysadmin 66540 Dec 20 2017 longfile.txt
sysadmin@localhost:~/Documents$ gzip longfile.txt
sysadmin@localhost:~/Documents$ ls -l longfile*
-rw-r--r-- 1 sysadmin sysadmin 341 Dec 20 2017 longfile.txt.gz
```

Dalam contoh sebelumnya, ada sebuah file bernama `longfile.txt` yang berukuran 66540 byte. File dikompresi dengan menjalankan perintah `gzip` dengan nama file sebagai satu-satunya argumen. Setelah perintah itu selesai, file asli hilang, dan versi terkompresi dengan ekstensi file `.gz` dibiarkan di tempatnya. Ukuran file sekarang adalah 341 byte.

Perintah `gzip` akan memberikan informasi ini, dengan menggunakan option `-l`, seperti yang ditunjukkan di sini:

```
sysadmin@localhost:~/Documents$ gzip -l longfile.txt.gz
      compressed      uncompressed   ratio uncompressed_name
            341              66540  99.5%  longfile.txt
```

Rasio kompresi diberikan sebagai 99.5%, pengurangan yang mengesankan dibantu oleh informasi berulang dalam file aslinya. Selain itu, ketika file didekompresi, itu akan dipanggil lagi `longfile.txt`.

File yang dikompresi dapat dikembalikan ke bentuk aslinya menggunakan perintah `gunzip` atau perintah `gzip -d`. Proses ini disebut *dekompresi*. Setelah `gunzip` melakukan tugasnya, file `longfile.txt` tersebut dikembalikan ke ukuran dan nama file aslinya.

```
sysadmin@localhost:~/Documents$ gunzip longfile.txt.gz
sysadmin@localhost:~/Documents$ ls -l longfile*
-rw-r--r-- 1 sysadmin sysadmin 66540 Dec 20 2017 longfile.txt
```

Ada perintah lain yang beroperasi secara hampir identik dengan `gzip` dan `gunzip`. Ada `bzip2` dan `bunzip2`, serta `xz` dan `unxz`.

Perintah `gzip` menggunakan **Lempel-Ziv** Data *algoritma kompresi*, sedangkan utilitas `bzip` menggunakan algoritma kompresi yang berbeda yang disebut **Burrows-Wheeler** blok penyortiran, yang dapat memampatkan file yang lebih kecil daripada `gzip` dengan mengorbankan lebih banyak waktu CPU. File-file ini dapat dikenali karena memiliki ekstensi `.bz` atau `.bz2` bukan ekstensi `.gz`.

Alat `xz` dan `unxz` secara fungsional mirip dengan `gzip` dan `gunzip` karena mereka menggunakan algoritme rantai **Lempel-Ziv-Markov (LZMA)**, yang dapat menghasilkan waktu dekompresi CPU yang lebih rendah yang setara `gzip` sambil memberikan rasio kompresi yang lebih baik yang biasanya dikaitkan dengan alat `bzip2`. File yang dikompresi dengan perintah `xz` menggunakan ekstensi `.xz`.

## 6.8 MENGARSIPKAN FILE

Jika Anda memiliki beberapa file untuk dikirim ke seseorang, Anda dapat memilih untuk mengompres setiap file satu per satu. Anda akan memiliki jumlah data yang lebih kecil secara total daripada jika Anda mengirim file yang tidak terkompresi, namun, Anda masih harus berurusan dengan banyak file sekaligus.

Pengarsipan adalah solusi untuk masalah ini. Utilitas UNIX tradisional untuk mengarsipkan file disebut `tar`, yang merupakan kependekan dari **T**ape **a**rchive . Itu digunakan untuk mengalirkan banyak *file* ke *tape* untuk *backup* atau transfer file. Perintah `tar` mengambil dalam beberapa file dan menciptakan file output tunggal yang dapat dibagi lagi ke dalam file asli di ujung transmisi.

Perintah `tar` memiliki tiga mode yang membantu untuk menjadi akrab dengan:

- *Buat*: Buat arsip baru dari serangkaian file.
- *Ekstrak*: Tarik satu atau lebih file dari arsip.
- *Daftar*: Menampilkan isi arsip tanpa mengekstrak.

Mengingat mode adalah kunci untuk mengetahui option baris perintah yang diperlukan untuk melakukan apa yang Anda inginkan. Selain mode, ingat di mana menentukan nama arsip, karena Anda mungkin memasukkan beberapa nama file pada baris perintah.

### 6.8.1 CREATE MODE

```
tar -c [-f ARCHIVE] [OPTIONS] [FILE...]
```

Membuat arsip dengan perintah `tar` membutuhkan dua option bernama:

Pilihan	Fungsi
<code>-c</code>	Buat arsip.
<code>-f ARCHIVE</code>	Gunakan file arsip. Argumennya <code>ARCHIVE</code> adalah nama file arsip yang dihasilkan.

Semua argumen yang tersisa dianggap sebagai nama file masukan, baik sebagai karakter pengganti, daftar file, atau keduanya.

Contoh berikut menunjukkan *file tar*, yang juga disebut *tarball*, dibuat dari banyak file. Argumen pertama membuat arsip bernama `alpha_files.tar`. Option wildcard `*` digunakan untuk menyertakan semua file yang dimulai dengan nama arsip `alpha`:

```
sysadmin@localhost:~/Documents$ tar -cf alpha_files.tar alpha*
sysadmin@localhost:~/Documents$ ls -l alpha_files.tar
-rw-rw-r-- 1 sysadmin sysadmin 10240 Oct 31 17:07 alpha_files.tar
```

Ukuran terakhir `alpha_files.tar` adalah 10240 byte. Biasanya, file tarball sedikit lebih besar dari file input gabungan karena informasi overhead untuk membuat ulang file asli. Tarballs dapat dikompresi untuk transportasi yang lebih mudah, baik dengan menggunakan `gzip` di arsip atau dengan `tar` ditambah option `-z`.

Pilihan	Fungsi
<code>-z</code>	Kompres (atau dekompresi) arsip menggunakan <code>gzip</code> perintah.

Contoh selanjutnya menunjukkan perintah yang sama seperti contoh sebelumnya, tetapi dengan tambahan option `-z`.

```
sysadmin@localhost:~/Documents$ tar -czf alpha_files.tar.gz alpha*
sysadmin@localhost:~/Documents$ ls -l alpha_files.tar.gz
-rw-rw-r-- 1 sysadmin sysadmin 417 Oct 31 17:15 alpha_files.tar.gz
```

Outputnya jauh lebih kecil daripada tarball itu sendiri, dan file yang dihasilkan kompatibel dengannya `gzip`, yang dapat digunakan untuk melihat detail kompresi. File yang tidak dikompresi memiliki ukuran yang sama seperti jika Anda mem-tar-nya dalam langkah terpisah:

```
sysadmin@localhost:~/Documents$ gzip -l alpha_files.tar.gz
      compressed      uncompressed   ratio uncompressed_name
          417              10240    96.1%  alpha_files.tar
```

Meskipun ekstensi file tidak memengaruhi cara file diperlakukan, konvensi digunakan `.tar` untuk tarball, dan `.tar.gz` atau `.tgz` untuk tarball terkompresi.

Kompresi bzip2 dapat digunakan sebagai pengganti gzip dengan menggantikan option `-j` pada option `-z` dan menggunakan `.tar.bz2`, `.tbz` atau `.tbz2` sebagai ekstensi file.

Pilihan	Fungsi
<code>-j</code>	Kompres (atau dekompresi) arsip menggunakan <code>bzip2</code> perintah.

Misalnya, untuk mengarsipkan dan mengompresi direktori School:

```
sysadmin@localhost:~/Documents$ tar -cjf folders.tbz School
```

### 6.8.2 LIST MODE

```
tar -t [-f ARCHIVE] [OPTIONS]
```

Diberikan arsip tar, dikompresi atau tidak, Anda dapat melihat apa yang ada di dalamnya dengan menggunakan option `-t`. Contoh selanjutnya menggunakan tiga option:

Pilihan	Fungsi
<code>-t</code>	Buat daftar file dalam arsip.
<code>-j</code>	Dekompresi dengan sebuah <code>bzip2</code> perintah.
<code>-f ARCHIVE</code>	Operasikan di arsip yang diberikan.

Untuk membuat daftar isi folders arsip `.tbz`:

```
sysadmin@localhost:~/Documents$ tar -tjf folders.tbz
School/
School/Engineering/
School/Engineering/hello.sh
School/Art/
School/Art/linux.txt
School/Math/
School/Math/numbers.txt
```

Dalam contoh, direktori School/ diawali dengan file. Perintah tar akan recurse ke dalam subdirektori secara otomatis ketika mengompresi dan akan menyimpan info path dalam arsip.

#### Perlu diperhatikan!

Untuk membuat daftar konten file dalam dua langkah, gunakan pipeline, “|” karakter.

```
sysadmin@localhost:~/Documents$ bunzip2 -c folders.tbz | tar -t
School/
School/Engineering/
School/Engineering/hello.sh
School/Art/
School/Art/linux.txt
School/Math/
School/Math/numbers.txt
```

Sisi kiri pipeline adalah bunzip2 -c folders.tbz, yang mendekompresi file, tetapi option -c mengirimkan output ke layar. Outputnya dialihkan ke tar -t. Jika Anda tidak menentukan file dengan -f maka tar akan membaca dari input standar, yang dalam hal ini adalah file yang tidak terkompresi.

**Catatan :** Contoh di atas dirancang untuk mendemonstrasikan kemampuan tar perintah untuk berulang ke subdirektori.

### 6.8.3 EXTRACT MODE

```
tar -x [-f ARCHIVE] [OPTIONS]
```

Membuat arsip sering kali digunakan untuk membuat banyak file lebih mudah dipindahkan. Sebelum mengekstrak file, pindah ke direktori Downloads:

```
sysadmin@localhost:~/Documents$ cd ~
sysadmin@localhost:~$ cp Documents/folders.tbz Downloads/folders.tbz
sysadmin@localhost:~$ cd Downloads
```

Terakhir, Anda dapat mengekstrak arsip dengan option -x setelah disalin ke direktori yang berbeda. Contoh berikut menggunakan pola yang sama seperti sebelumnya, menentukan operasi, kompresi, dan nama file yang akan dioperasikan.

Pilihan	Fungsi
<code>-x</code>	Ekstrak file dari arsip.
<code>-j</code>	Dekompresi dengan <code>bzip2</code> perintah.
<code>-f ARCHIVE</code>	Operasikan di arsip yang diberikan.

```
sysadmin@localhost:~/Downloads$ tar -xjf folders.tbz
sysadmin@localhost:~/Downloads$ ls -l
total 8
drwx----- 5 sysadmin sysadmin 4096 Dec 20 2017 School
-rw-rw-r-- 1 sysadmin sysadmin 413 Oct 31 18:37 folders.tbz
```

File asli tidak tersentuh, dan direktori baru dibuat. Di dalam direktori, ada direktori dan file asli.

<code>sysadmin@localhost:~/Downloads\$ cd School</code>	
<code>sysadmin@localhost:~/Downloads/School\$ ls -l</code>	
total 12	
<code>drwx----- 2 sysadmin sysadmin 4096 Oct 31 17:45 Art</code>	
<code>drwx----- 2 sysadmin sysadmin 4096 Oct 31 17:47 Engineering</code>	
<code>drwx----- 2 sysadmin sysadmin 4096 Oct 31 17:46 Math</code>	

Tambahkan `-v` dan Anda akan mendapatkan hasil verbose dari file yang diproses, membuatnya lebih mudah untuk melacak apa yang terjadi:

Pilihan	Fungsi
<code>-v</code>	Buat daftar file yang diproses secara verbal.

Contoh berikutnya mengulangi contoh sebelumnya, tetapi dengan tambahan option `-v`:

<code>sysadmin@localhost:~/Downloads\$ tar -xjvf folders.tbz</code>	
<code>School/</code>	
<code>School/Engineering/</code>	
<code>School/Engineering/hello.sh</code>	
<code>School/Art/</code>	
<code>School/Art/linux.txt</code>	
<code>School/Math/</code>	
<code>School/Math/numbers.txt</code>	

Penting untuk menjaga flag `-f` di akhir, karena `tar` mengasumsikan apapun yang mengikuti option ini adalah nama file. Dalam contoh berikutnya, flag `-f`

dan `-v` dialihkan, mengarah ke `tar` interpretasi perintah sebagai operasi pada file yang dipanggil `v`, yang tidak ada.

```
sysadmin@localhost:~/Downloads$ tar -xjfv folders.tbz
tar (child) : v: Cannot open: No such file or directory
tar (child) : Error is not recoverable: exiting now
tar: Child returned status 2
tar: Error is not recoverable: exiting now
```

Jika Anda hanya ingin beberapa file keluar dari arsip, tambahkan namanya di akhir perintah, tetapi secara default, mereka harus sama persis dengan nama dalam arsip, atau menggunakan pola.

Contoh berikut menunjukkan arsip yang sama seperti sebelumnya, tetapi hanya mengekstrak file `School/Art/linux.txt`. Output dari perintah (karena mode verbose diminta dengan `-v`) hanya menunjukkan satu file yang telah diekstraksi:

```
sysadmin@localhost:~/Downloads$ tar -xjvf folders.tbz School/Art/linux.txt
School/Art/linux.txt
```

Perintah `tar` memiliki lebih banyak fitur, seperti kemampuan untuk menggunakan pola ketika mengekstrak file, termasuk file tertentu, atau keluaran file diekstrak ke layar bukan disk. Dokumentasi untuk `tar` memiliki informasi yang mendalam.

## 6.9 ZIP FILE

Utilitas pengarsipan de facto di Microsoft adalah file ZIP. ZIP tidak lazim di Linux tetapi didukung dengan baik oleh perintah `zip` dan `unzip`. Meskipun, dengan `tar` dan perintah `gzip/gunzip` dan option yang sama dapat digunakan secara bergantian untuk melakukan pembuatan dan ekstraksi, tetapi ini tidak terjadi dengan `zip`. Option yang sama memiliki arti berbeda untuk dua perintah berbeda.

Mode default `zip` adalah menambahkan file ke arsip dan mengkompresnya.

```
zip [OPSI] [zipfile [file...]]
```

Argumen pertama `zipfile` adalah nama arsip yang akan dibuat, setelah itu daftar file yang akan ditambahkan. Contoh berikut menunjukkan arsip terkompresi yang disebut `alpha_files.zip` yang sedang dibuat:

```
sysadmin@localhost:~/Documents$ zip alpha_files.zip alpha*
    adding: alpha-first.txt (deflated 32%)
    adding: alpha-second.txt (deflated 36%)
    adding: alpha-third.txt (deflated 48%)
    adding: alpha.txt (deflated 53%)
    adding: alpha_files.tar.gz (stored 0%)
```

Outputnya menunjukkan file dan rasio kompresi.

Perlu dicatat bahwa `tar` memerlukan option `-f` untuk menunjukkan nama file sedang dikirim, sementara `zip` dan `unzip` memerlukan nama file dan oleh karena itu Anda tidak perlu memberi tahu perintah bahwa nama file sedang diteruskan.

Perintah `zip` tidak akan recurse ke dalam subdirektori secara default, yang perilaku berbeda dari perintah `tar`. Artinya, menambahkan `School` hanya akan menambah direktori kosong dan bukan file didalamnya. Jika Anda menginginkan `tar` perilaku seperti itu, Anda harus menggunakan option `-r` untuk menunjukkan rekursi yang akan digunakan:

```
sysadmin@localhost:~/Documents$ zip -r School.zip School
updating: School/ (stored 0%)
updating: School/Engineering/ (stored 0%)
updating: School/Engineering/hello.sh (deflated 88%)
updating: School/Art/ (stored 0%)
updating: School/Art/linux.txt (deflated 49%)
updating: School/Math/ (stored 0%)
updating: School/Math/numbers.txt (stored 0%)
adding: School/Art/red.txt (deflated 33%)
adding: School/Art/hidden.txt (deflated 1%)
adding: School/Art/animals.txt (deflated 2%)
```

Pada contoh di atas, semua file di bawah direktori `School` ditambahkan karena menggunakan option `-r`. Baris pertama keluaran menunjukkan bahwa direktori telah ditambahkan ke arsip, tetapi jika tidak, keluarannya mirip dengan contoh sebelumnya.

Option `-l` dari `unzip` di arsip `.zip`:

```
sysadmin@localhost:~/Documents$ unzip -l School.zip
Archive: School.zip
      Length      Date    Time     Name
-----  -----
          0  2017-12-20 16:46  School/
          0  2018-10-31 17:47  School/Engineering/
        647  2018-10-31 17:47  School/Engineering/hello.sh
          0  2018-10-31 19:31  School/Art/
         83  2018-10-31 17:45  School/Art/linux.txt
          0  2018-10-31 17:46  School/Math/
         10  2018-10-31 17:46  School/Math/numbers.txt
         51  2018-10-31 19:31  School/Art/red.txt
         67  2018-10-31 19:30  School/Art/hidden.txt
         42  2018-10-31 19:31  School/Art/animals.txt
-----  -----
        900                               10 files
          0  2018-10-31 17:46  School/Math/
         10  2018-10-31 17:46  School/Math/numbers.txt
-----  -----
        740                               7 files
```

Mengekstrak file sama seperti membuat arsip, karena operasi default dari perintah `unzip` ini adalah mengekstrak. Ini memberikan beberapa option jika membuka ritsleting file akan menimpa yang sudah ada:

```
sysadmin@localhost:~/Documents$ unzip School.zip
Archive: School.zip
replace School/Engineering/hello.sh? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
replace School/Art/linux.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
replace School/Math/numbers.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
replace School/Art/red.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
replace School/Art/hidden.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
replace School/Art/animals.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
```

Ini dapat dihindari dengan menyalin file zip ke direktori baru:

```
sysadmin@localhost:~/Documents$ mkdir tmp
sysadmin@localhost:~/Documents$ cp School.zip tmp/School.zip
sysadmin@localhost:~/Documents$ cd tmp
sysadmin@localhost:~/Documents/tmp$ unzip School.zip
Archive: School.zip
  creating: School/
  creating: School/Engineering/
  inflating: School/Engineering/hello.sh
  creating: School/Art/
  inflating: School/Art/linux.txt
  creating: School/Math/
  extracting: School/Math/numbers.txt
  inflating: School/Art/red.txt
  inflating: School/Art/hidden.txt
  inflating: School/Art/animals.txt
```

Di sini, kami mengekstrak semua file dalam arsip ke direktori saat ini. Sama seperti tar, Anda dapat memberikan nama file pada baris perintah. Contoh di bawah ini menunjukkan tiga upaya berbeda untuk mengekstrak file.

Pertama, hanya nama file yang diteruskan tanpa komponen direktori. Seperti tar, file tersebut tidak cocok.

```
sysadmin@localhost:~/Documents/tmp$ unzip School.zip linux.txt
Archive: School.zip
caution: filename not matched: linux.txt
```

Upaya kedua melewati komponen direktori bersama dengan nama file, yang hanya mengekstrak file itu.

```
sysadmin@localhost:~/Documents/tmp$ unzip School.zip School/Math/numbers.txt
Archive: School.zip
extracting: School/Math/numbers.txt
```

Versi ketiga menggunakan wildcard, yang mengekstrak empat file yang cocok dengan pola, sama seperti tar.

```
sysadmin@localhost:~/Documents/tmp$ unzip School.zip School/Art/*t
Archive: School.zip
  inflating: School/Art/linux.txt
  inflating: School/Art/red.txt
  inflating: School/Art/hidden.txt
  inflating: School/Art/animals.txt
```

zip dan unzip halaman man menjelaskan hal-hal lain yang dapat Anda lakukan dengan alat ini, seperti mengganti file dalam arsip, menggunakan tingkat kompresi yang berbeda, dan enkripsi digunakan bahkan.