

Menggunakan Filesystem RAM-Based

Jika anda membuat tempat penyimpanan untuk sistem yang kecil, sementara, filesystem yang kecil di RAM – istilahnya *ramfs* – dapat memberikan akses kecepatan tinggi. Filesystem ini relatif kecil (defaultnya) karena maksimum RAM yang digunakan *ramfs* adalah setengah dari total RAM pada sistem anda. Jadi jika anda memiliki 2 GB RAM, *ramfs* hanya dapat menggunakan 1 GB, *ramfs* sesuai untuk sistem file kecil yang harus diakses dengan cepat. Sebagai contoh, ramfs digunakan untuk gambar berukuran kecil pada web site dengan trafik yang sangat tinggi.

Untuk menggunakan ramfs, anda harus meng'enable'-kan nya di kernel:

1. Download source kernel linux terakhir dari www.kernel.org dan ekstrak di direktori `/usr/src/linux-version`.
2. Pilih File systems submenu. Gunakan spacebar, pilih *Simple RAM-based file system support* agar di masukkan ke kernel sebagai module dan exit dari submenu.
3. Pastikan feature kernel lainnya yang anda gunakan juga di pilih seperti biasanya (make oldconfig).
4. Exit dari main menu dan simpan konfigurasi kernel.
5. Jalankan perintah `make dep`, `make bzImage`, `make modules`, `make modules_install` untuk meng'install kernel baru di lokasi yang tepat.
6. Rubah direktori ke `arch/i386/boot` , copy bzImage ke `/boot/vmlinuz-baru` dan edit file `/etc/lilo.conf` untuk memasukkan konfigurasi baru sebagai berikut :

```
image = /boot/vmlinuz-baru
label = linux2
read-only
root = /dev/hda1
```

7. Jalankan perintah `/sbin/lilo` untuk reconfigure LILO dan reboot mesin anda. Pada lilo prompt enter linux2 dan boot ke kernel baru. Jika terjadi masalah anda dapat me'reboot menggunakan kernel default (standar linux) dan modifikasi kembali konfigurasi kernel.
8. Setelah boot dengan kernel baru, sistem anda telah siap untuk menggunakan kemampuan ramfs. Buatlah direktori ramdrive, `mkdir /ramdrive`.
9. Mount filesystem ramfs :
`mount -t ramfs none /ramdrive`

Kini anda dapat menulis/baca ke `/ramdrive` seperti biasa.

Catatan : Ketika system direboot atau di unmount filesystem, seluruh isi */ramdrive* akan hilang. Ini sebabnya mengapa menjadi tempat sementara untuk akses kecepatan tinggi. Karena ramfs sebenarnya bukan block device, seperti program `df` dan `du` tidak dapat melihatnya. Anda dapat memverifikasi jika anda benar-benar menggunakan RAM dengan menjalankan perintah `cat /proc/mounts` dan lihat entri berikut :
`none /ram ramfs rw 0 0`

Tips : Anda dapat menggunakan opsi `-O` ketika meng'mount filesystem seperti mounting filesystem disk-based biasa. Contohnya, untuk meng'mount filesystem ramfs dengan mode read-only, gunakan opsi `-O ro`. Juga opsi khusus seperti `maxsize=n` dimana `n` adalah kilobytes untuk alokasi filesystem RAM; `maxfiles=n` dimana `n` adalah jumlah seluruh file yang dibolehkan masuk di filesystem RAM; `maxinodes=n` dimana `n` adalah maksimum jumlah dari inodes (default nya `0 = no limits`).

Jika anda menjalankan Web server, anda dapat mendapatkan banyak manfaat dengan menggunakan filesystem RAM. Elemen-elemen seperti gambar dan file-file yang berukuran tidak terlalu besar dapat disimpan di ramfs filesystem. Anda dapat menulis shell script sederhana untuk meng'copy isi dari lokasi asli ke ramdrive setiap reboot.

```
#!/bin/sh
#
# Simply script to create a ramfs filesystem
# on $MOUNTPOINT (which must exists).
#
# It copies files from $ORIG_DIR to $MOUNTPOINT
# and changes ownership of $MOUNTPOINT to
# $USER and $GROUP
#
# Change values for these variables to suit
# your needs.
MOUNTPOINT=/ram
ORIG_DIR=/www/commonfiles
USER=httpd
GROUP=httpd
MOUNTCMD=/bin/mount
CHOWN=/bin/chown
CP=/bin/cp
echo -n "Creating ramfs filesystem in $MOUNTPOINT ";
$MOUNTCMD -t ramfs none $MOUNTPOINT
echo "done.";
echo -n "Copying $ORIG_DIR to $MOUNTPOINT ... ";
$CP -r $ORIG_DIR $MOUNTPOINT
echo "done.";
echo -n "Changing ownership to $USER:$GROUP for $MOUNTPOINT ...";
$CHOWN -R $USER:$GROUP $MOUNTPOINT
echo "done.";
```

Untuk menggunakan script tersebut lakukan prosedur sebagai berikut :

1. Buatlah script tersebut, *make_ramfs.sh* di direktori */usr/local/scripts*. Buat */usr/local/scripts* direktori nya terlebih dahulu jika tak ada.
2. Edit file */etc/rc.d/rc.local* dan lampirkan baris berikut :
/usr/local/scripts/make_ramfs.sh
3. Buat direktori */ram*, jika anda menyimpan file-file yang di inginkan ke RAM pada tempat lain, modifikasi variabel *ORIG_DIR* di script. Contoh, jika file tempatnya di direktori */www/mydomain/htdocs/common*, lalu set variabel ini ke direktori tersebut.
4. Jika anda menjalankan Web server menggunakan username dan gorup selain *httpd*, rubahlah variabel *USER* dan *GROUP* . Contoh jika anda menjalankan Apache sebagai *nobody* (user dan goup), lalu rubahlah menjadi
USER=nobody dan *GROUP=nobody*
5. Asumsi anda menggunakan Apache Web server, buat alias file di *httpd.conf* sebagai berikut :
Alias */commonfiles/* *"/ram/commonfile/"*

Jika sewaktu-waktu Web server memerlukan akses ke */commonfiles/**, kini telah menggunakan versi di RAM, dimana seharusnya substansial lebih cepat daripada disimpan di lokasi asli. Perlu di ingat, versi RAM-based akan hilang jika anda me'reboot atau unmount filesystem. Jadi jangan pernah meng'update kecuali anda juga meng'copy balik isinya ke direktori berbasis disk.

Referensi :

RedHat Press - RedHat Linux Security and Optimization
by Mohammed J. Kabir
www.hungryminds.com copyright 2002.

Semoga Bermanfaat.

Faiz