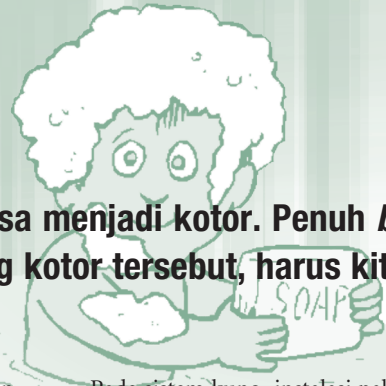


Sistem yang Bersih

Setelah setiap hari digunakan dan diutak-utik, sistem bisa menjadi kotor. Penuh *broken link*, paket yang konflik dan lain sebagainya. Sistem yang kotor tersebut, harus kita bersihkan agar tetap prima.



Komputer adalah seperti manusia. Butuh mandi. Tentu saja kita tidak bisa membayangkan bagaimana kalau tidak mandi selama satu bulan. Setiap hari beraktivitas, dan menjadikan kita keringatan. Terkadang ada makanan tumpah di baju ketika ceroboh pada saat makan. Bisa dibayangkan bagaimana kita bisa tampil prima ketika satu bulan ke depan kita tidak juga membersihkan badan.

Sama juga dengan komputer. Setiap hari kita bekerja dengan komputer. Kita mungkin meninggalkan banyak sampah sistem seperti file *temporary* yang tidak dihapus oleh programnya.

Ada kalanya kita menginstal program, dan membiarkan terkatung-katung tersebar di berbagai direktori sistem. Pada sistem berbasis RPM (Red Hat/Fedora dan SUSE misalnya), kita mungkin pernah menginstal paket dengan opsi *-force*, yang mungkin menyebabkan inkonsistensi paket sistem, dan lain sebagainya.

Hal-hal lain juga bisa mengotori sistem. Terkadang kita membuat variabel sistem/shell tertentu, dan menjalankannya setiap kali login, namun tidak pernah ingat untuk melakukan *unset* variabel tersebut.

Atau, kita mungkin menyimpan data kita secara tersebar dan karena sudah terlalu kacau, maka tidak berani sama sekali menghapus salah satu file di suatu partisi atau direktori karena takut ada data penting yang akan terhapus.

Gambar-gambar yang kita *download* pun tersebar di mana-mana dan kita menjadi tak berdaya untuk membereskannya.

Setiap hari merasa tak berdaya, dan hal tersebut dijalankan dalam waktu yang lama. Niscaya, sistem kita dapat menjadi kotor sekali.

Cara yang termudah tentu saja instal ulang. Tapi, hal tersebut bisa dilakukan apabila kita telah mengorganisasi data secara baik. Selain

itu, instal ulang tidak akan melatih kita untuk menjaga sistem. Dan yang penting, instal ulang bukan selalu merupakan solusi yang tepat. Sama seperti membeli server *branded* padahal tidak membutuhkannya.

Kalau sistem bisa ngomong, mungkin kita tidak akan berani mendengar keluhannya. Bisa-bisa terlalu banyak. Apabila sistem Anda kacau berantakan, mari bersama-sama untuk membereskannya. Bersama-sama membahas cara memperlakukan sistem agar tetap bersih.

Agar sistem dapat kembali tampil prima, seprima manusia yang baru mandi ketika tidak mandi selama satu bulan.

Paket yang konflik

Dengan ribuan program yang tersebar di dunia *free software*, bukan salah kita apabila kita selalu tergoda untuk menginstal program-program baru. Apalagi ketika sang *developer* menjadi sangat aktif untuk menghasilkan aplikasi kelas satu.

Syukur apabila *maintainer* atau *developer*-nya memaketkan software yang mereka buat ke dalam paket yang bisa diterima distro kita. Kita bisa dengan mudah mengambil dan kemudian menginstalnya.

Di satu sisi, hal ini membuat kita tidak perlu memaksakan instalasi dengan paket untuk distro lain misalnya. Atau mengompilasi sendiri dari *source code* yang menempatkan *prefix default* yang akan menjadikan banyak file tanpa induk tersebar di sistem. Atau bahkan dengan cara-cara lain yang lebih aneh lagi, misalnya dengan mengekstrak isi paket RPM, kemudian mencomot library dari distro lain dan kemudian membuat link file, yang akhirnya lupa dihapus sehingga menjadi *broken link*.

Ada beberapa pendekatan yang bisa dilakukan agar sistem tetap bersih. Kita bagi dua besar menjadi sistem kuno dan sistem modern. Ini hanyalah istilah penulis.

Pada sistem kuno, instalasi paket benar-benar hanya dilakukan apabila penting. Setiap kali melakukan instalasi paket, atau setiap jangka waktu tertentu, kita mencatat paket-paket apa saja yang sudah diinstal. Kalau perlu, instalasi dilakukan tidak secara *system wide*, namun dengan membuat sebuah root palsu dimana program eksperimental diinstal ke dalamnya.

Sistem kuno ini berhati-hati dalam instalasi paket. Hanya menginstal paket untuk distro yang digunakan. Pantang dalam mencomot paket yang bukan untuk distronya. Pantang sekali melakukan konversi paket (dengan alien misalnya).

Apabila kompilasi dari *source code* harus dilakukan, maka *prefix* harus diatur ke direktori root palsu tersebut (jangan ke */usr/local*) dan instal semua file yang dibutuhkan dalam satu direktori. Apabila hal ini memicu banyak file yang berganda, biarkan saja. Yang penting, satu program beserta semua file pendukungnya dalam satu direktori. Tidak menyebar ke mana-mana. Apabila ingin dihapus, direktori yang bersangkutan tinggal dihapus saja.

Sistem kuno sangat menolak instalasi hibrid. Mencomot satu hal dari sistem lain, dan mencomot hal lain dari sistem lain.

Pada hakikatnya, sistem kuno ini berusaha sebisa mungkin melakukan kompromi terbaik di satu sistem. Audit akan dilakukan secara berkala untuk memastikan tidak ada paket yang konflik atau *broken* (pada SUSE, apabila ada paket yang bermasalah, pada saat menjalankan YaST untuk instalasi program, sebuah pesan akan ditampilkan, lengkap dengan informasi paketnya).

Kelemahan dari sistem ini adalah sangat membutuhkan orang yang rajin untuk mengawasi sistem. Atau, sangat membutuhkan perhatian. Namun, cara ini cocok digunakan apabila Anda termasuk orang yang senang sistem yang sederhana, yang konservatif, dan terjaga penuh oleh Anda.

Sistem modern lain lagi. Dan biasanya, sistem ini membutuhkan pengorbanan. Umumnya, di sistem ini, pengguna menginstal satu sistem yang benar-benar bersih, kemudian menggunakan cara lain dalam menginstal program.

Ada yang menggunakan VMWare untuk menginstal distro di atas distro yang berjalan, kemudian mengizinkan untuk saling mengakses file sistem. Sistem ini banyak kekurangan, walaupun ada yang melakukannya.

Ada yang menggunakan User Mode Linux. Cara ini dipertimbangkan sebagai cara yang modern dan canggih, dengan pengorbanan tidak sebesar penggunaan VMWare. Pada penggunaan UML, semuanya dilakukan dalam *image*. Kali pertama, *image master* yang bersih dibuat. Kemudian *image* lain dibuat berdasarkan *image master* ini. Apabila suatu *image* menjadi kotor, maka *image* baru tinggal dibuat dari *image master*. Dan hebatnya, sistem ini memungkinkan adanya sistem multi distro.

Pada hakikatnya, sistem modern tidak mengizinkan adanya sentuhan pada sistem dasar. Semua perubahan dilakukan pada sistem di atasnya. Cara yang satu ini setidaknya memiliki satu kekurangan, yaitu resource yang dibutuhkan lebih besar. Lebih rakus. Seiring dengan perkembangan zaman, sistem modern yang satu ini lebih banyak dilirik daripada sistem kuno.

Perhatikan sistem yang ingin Anda ikuti. Anda bisa mengawinkan kedua sistem tersebut. Yang penting, kontrol mengontrol paket sistem tidak memakan terlalu banyak waktu Anda. Apabila bisa diotomatisasi dengan script, lakukanlah. Selalu jaga sistem agar tetap bersih. Masalah *dependency* adalah masalah yang menyebabkan. Jangan biarkan ini terjadi pada Anda.

Permasalahan /tmp

Direktori /tmp adalah direktori paling baik hati di sistem Linux. Proses mana saja yang ingin curhat, silahkan. Proses mana saja yang ingin meminjam space, silakan. Proses yang meminjam *space* lantas tidak mengembalikan *space*, juga silakan.

Karena terlalu baik hati, maka /tmp seringkali menjadi korban. Coba periksa /tmp Anda dengan perintah berikut ini:

```
du -sh /tmp
```

Apabila Anda hanya menjalankan KDE atau GNOME dan menggunakan komputer dalam pengertian penggunaan desktop, maka angka 1 atau 2 MB masih normal. Tapi kalau sudah di atas 2 MB, maka harus dipertanyakan.

Distro Debian boleh mendapatkan acungan jempol dalam hal ini. Setiap kali booting, /tmp akan dibersihkan. Oleh karena itu, jangan coba-coba untuk menyimpan file sementara di /tmp dan melakukan *rebooting*, karena bisa dihapus.

Distro lain memiliki mekanisme tersendiri. Namun, apabila Anda ingin cara bar-bar dalam membersihkan /tmp, lakukanlah langkah-langkah berikut ini:

- Tutup semua aplikasi.
- Reboot.
- Matikan sebanyak mungkin proses.
- Hapus semua isi /tmp.
- Reboot.

Apabila Anda membenci reboot, maka cukup tutup semua aplikasi dan matikan sebanyak mungkin proses, kemudian hapus isi di /tmp.

Kenapa /tmp menjadi penting? Tidak ada sentimen negatif tertentu untuk direktori yang baik hati seperti /tmp. Yang menjadi permasalahan adalah kemungkinan DoS dengan memenuhi isi /tmp.

Sebuah proses terkadang harus menulis ke /tmp. Dan bisa Anda bayangkan kalau /tmp menjadi penuh gara-gara ada proses nakal yang terus mengisi /tmp dengan file sampah? Untuk minimasi risiko, apabila mengelola server yang melayani banyak orang iseng, pastikan memisahkan /tmp pada partisi tersendiri.

Broken link

Namanya mirip dengan *broken home*. Dan, akibatnya juga mirip-mirip. Terkadang, untuk memudahkan akses atau dengan alasan lainnya, beberapa proses membuat symlink. Pengguna pun terkadang menggunakan symlink untuk mempermudah akses file atau sekadar menjadikan suatu program tampak lebih umum.

Masalahnya adalah ketika suatu proses atau pengguna tidak bertanggung jawab. Bisa Anda bayangkan kalau terdapat banyak link yang tidak mengacu kepada file mana-pun juga di sistem Anda?

Penulis menyebut ini sebagai *broken link syndrome*. Walau tidak separah *syndrome* lainnya, hal ini cukup merepotkan sistem. Karena, walaupun tidak banyak memakan *space* banyak, symlink tetaplah sebuah file.

Distro yang populer sekalipun, tidak terlepas dari masalah ini. Terkadang bukan sengaja ingin membuat symlink lantas tidak bertanggungjawab, namun mungkin karena lepas dari sistem QC.

Penulis membuat sebuah script kecil yang dapat melaporkan symlink-symlink mana saja yang broken di sistem Anda. Pencarian bisa dilakukan secara rekursif. Berikut ini adalah isi script tersebut:

```
#!/bin/sh

#find broken link on specified
directory
#(c) Noprianto, August 18 2004
#GPL

[ $# -lt 1 ] && echo `basename
$0` \<dir\> [-v] && exit 1
test ! -d $1 && echo "$1 is not
a directory" && exit 2

REPORT=REPORT

echo "please wait..."
echo "start scanning: `date`" >
$REPORT
echo "-- start of broken link in
$1 --" >> $REPORT

for link in `find $1 -type l
2>/dev/null`
do
    test ! -z $2 && test $2 = -v
    && echo "scanning $link..."
    file $link 2>/dev/null | grep
broken\ symbolic\ link\ to >>
$REPORT
done

echo "-- end of broken link in
$1 --" >> $REPORT
echo "end scanning: `date`" >>
$REPORT

echo "COUNT: `cat $REPORT | grep
broken\ symbolic\ link\ to | wc
-l` broken links" >> $REPORT
echo "done."
```

Berikut ini adalah contoh laporannya:

```
start scanning: Wed Aug 18
23:23:15 WIT 2004
-- start of broken link in /opt/
kde3/ --
/opt/kde3/share/doc/HTML/da/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/de/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/de/
digikam/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/de/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/es/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/es/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/et/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/et/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/fi/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/fr/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/fr/
digikam/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/fr/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/it/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/pt/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/ru/k3b/
common: broken symbolic link to
`../common'
/opt/kde3/share/doc/HTML/ru/
knights/common: broken symbolic
link to `../common'
/opt/kde3/share/doc/HTML/sv/k3b/
```

```
common: broken symbolic link to
`../common'
-- end of broken link in /opt/
kde3/ --
end scanning: Wed Aug 18
23:23:34 WIT 2004
COUNT: 17 broken links
```

Script tersebut dapat dijalankan dengan cara berikut:

```
./find-broken-link.sh <dir> [-v]
```

Apabila <dir> ditentukan, maka pemeriksaan akan dilakukan secara rekursif. Apabila -v diberikan, maka tampilan akan dibuat verbose.

Isi laporan dalam file REPORT tersebut kemudian bisa di tampilkan dan dilewatkan ke program rm untuk dihapus. Dengan demikian, sistem Anda akan terbebas dari symlink yang kehilangan induk.

Prinsip kerja program ini sebenarnya sederhana. Program akan mencari file dengan tipe symlink dan kemudian memeriksa file acuannya. Apabila tidak ditemukan, maka dianggap sebagai broken link. Program ini bekerja memanfaatkan program find dan file.

Home directory yang kacau balau

Ini sedikit berbeda dengan rumah tangga yang kacau balau, walaupun bisa dimirip-miripkan. Hal ini bisa menimbulkan masalah yang serius. Periksa file-file di dalam homedir Anda, dan periksa jugalah space yang dimakan oleh homedir Anda tersebut.

Terkadang, Anda ingin mencoba program-program yang terinstal di sistem. Program-program tersebut lantas menyimpan konfigurasi di dalam homedir Anda dalam file atau direktori berawalan titik sehingga luput dari perintah ls.

Apabila dibiarkan, homedir Anda akan menjadi begitu berantakan dan menjadikan rumah Anda tidak seru untuk ditinggali.

Apabila Anda senang mencoba-coba program, lakukan trik berikut untuk menjaga agar homedir Anda tetap bersih walaupun Anda mencoba puluhan software setiap hari:

- Buat *template* yang bersih, lengkap dengan segala konfigurasi desktop.
- Apabila dirasa perlu, buat beberapa *template*.

- Cobalah setiap software yang Anda inginkan.
- *Restore* secara berkala dengan *template* yang paling Anda senangi.

Jangan sampai suatu hari Anda menjadi bingung untuk membersihkan rumah Anda sendiri karena terlalu banyak file di dalamnya.

Variabel shell/sistem

Dari ribuan *free software* yang bisa kita temukan, sebagian diantaranya dapat bertingkah laku berbeda apabila terdapat variabel shell yang diharapkan.

Beberapa di antaranya bahkan dengan nakalnya mengubah ~/.profile dan menambahkan berbagai variabel shell sesuai keinginannya.

Kenapa hal ini bisa menjadi masalah di masa depan? Perhatikan skenario berikut. Ada sebuah program yang kita sukai, namun program tersebut sudah cukup kuno. Ia membutuhkan pustaka-pustaka yang kuno. Dan untuk mendukung agar kebutuhannya terpenuhi, kita mengatur variabel LD_LIBRARY_PATH yang mengacu kepada lokasi pustaka yang dibutuhkan.

Suatu saat, kita menginstal program baru yang secara kebetulan membutuhkan pustaka yang sama, hanya beda versi. Sayangnya, tidak ada pemeriksaan versi pustaka terlebih dahulu oleh program baru tersebut. Begitu ditemukan pustaka dengan nama yang diharapkan, pustaka kuno tersebut, langsung dibuka. Dan rupanya, hasilnya tidak sesuai harapan karena ada saja yang fungsi yang tidak ditemukan. Padahal, sebenarnya, pustaka versi barunya duduk manis di lokasi lain.

Hal tersebut adalah bisa berujung pada debugging yang tidak perlu. Padahal, semua terjadi karena kita lupa membersihkan satu variabel saja.

Selalu periksa ~/.profile atau /etc/profile untuk berbagai variabel buatan sendiri yang sebenarnya tidak diperlukan lagi.

Simpan data Anda!

Ada pepatah yang mengatakan jangan simpan semua telur dalam satu keranjang. Namun, ada pepatah lain yang mengatakan sebaliknya. Simpan semua dalam satu keranjang agar kamu dapat mengawasinya dengan konsentrasi.

Apapun yang Anda anut, Anda perlu memperhatikan lokasi penyimpanan data Anda. Terutama kalau Anda termasuk orang yang senang kebersihan sistem. Atau, Anda senang menghancurkan sistem untuk kemudian melakukan instalasi ulang.

Penulis selalu membiasakan untuk menyimpan semua data dalam partisi terpisah. Kalau terjadi apa-apa dengan sistem, risiko dapat diminimasi. Tentunya, *back-up* teratur juga harus dilakukan.

Ada kalanya, penulis menyimpan data dalam format image ISO 9660, yang siap di-burn ke CDROM. Terdengar aneh? Mungkin. Tapi, penulis senang dengan data yang tidak tersebar-sebar sehingga memudahkan *back-up*.

Apabila disimpan dalam kompresi, penulis anggap merepotkan ketika ingin mengakses isinya karena butuh dekompresi yang memakan waktu dan space hard-disk. Dengan menyimpannya dalam format image ISO 9660, selain tidak tersebar, dapat di-mount *loopback* dengan mudah dan cepat.

Untuk mount *loopback* sebuah image, berikan perintah berikut sebagai root:

```
mount -o loop <IMAGE> <MOUNT_
POINT>
```

Apapun cara yang Anda pilih, sekali lagi, jagalah data Anda.

Security

Security adalah hal yang jelas membutuhkan banyak perhatian dari sisi manusia. Seaman-amannya sistem, apabila manusianya tidak memiliki pandangan yang tinggi soal keamanan—setidaknya peduli terhadap keamanan—maka keamanan sudah sekali untuk diwujudkan.

Lantas, apa hubungannya dengan kebersihan sistem? Apabila Anda seorang admin jaringan atau developer, seberapa sering Anda menjalankan daemon yang eksperimental? Atau seberapa sering Anda membuka port tertentu untuk melakukan eksperimen?


Kegiatan-kegiatan tersebut dapat memicu masalah lain yang lebih serius, jauh lebih serius dibanding sistem kotor.

Pastikan Anda memeriksa keamanan sistem dengan program yang terpercaya. Gunakan *Intrusion Detection System* yang andal, dan sesekali, apabila Anda sering menjalankan daemon eksperimental, jalankanlah port scanning dengan *nmap*.

Menjadi pengguna yang paranoid?

Menjadi orang yang serba takut tentu bukan hal yang membahagiakan. Sedikit tidak beres, lantas ketakutan. Sedikit kacau, bisa-bisa kalang kabut.

Tulisan ini tidak bermaksud mempengaruhi Anda agar menjadi pengguna yang serba resik dan paranoid. Hanya, sistem yang lebih bersih akan menjadikan waktu maintenance lebih sedikit (*cost cutting*) dan Anda dapat mempercayakan sistem untuk melayani Anda tanpa harus memendam rasa khawatir. Dan, bukanlah sistem yang bersih akan mampu melayani Anda lebih prima?

Demikian pembahasan kita soal sistem yang bersih. Terapkan sesuai cara yang Anda senangi untuk menjaga sistem Anda! 
Noprianto (noprianto@infolinux.co.id)

MELAYANI
FULL MIGRASI
KE LINUX

Program Intensif LINUX Profesional Linux Server Development



MATA KULIAH DASAR

- **Pengantar Komputer**
(Komponen & Konfigurasi Komputer dan Jaringan Komputer / LAN)
- **Linux Fundamental**
(Basic User, X-window, System Administrator, Networking)
- **Internet & Aplikasi WEB**
(Browser, Search Engine, Email, FTP, HTML, CSS & JavaScript)



MATA KULIAH INTI

- Shell Programming
- Advanced System Administration
- Security & Advanced Networking
- PHP & MySQL

FASILITAS:

- ☺ Minimal PIII+ Networking
- ☺ 1 Komputer Per Peserta
- ☺ Ruang Kuliah Full AC
- ☺ Buku Panduan Belajar
- ☺ Souvenir Eksklusif
- ☺ Sertifikat
- ☺ Konsultasi
- ☺ Disket
- ☺ CD Linux
- ☺ Block Notes

WAKTU KULIAH:

- ☺ Senin s.d. Kamis
- ☺ Sabtu & Minggu

Exclusive
Class
For Corporate

- ☺ Linux Terpadu
- ☺ Web Terpadu
- ☺ PHP & MySQL
- ☺ PHP & PostgreSQL

FASILITAS:

- ☺ LCD Projector
- ☺ Komputer Minimal PIII
- ☺ Sertifikat
- ☺ Block Notes
- ☺ Snack (Coffee Break)
- ☺ Makan Siang

LEMBAGA PENDIDIKAN KOMPUTER
NURUL FIKRI

- Jl. Margonda Raya No.522 - Depok 16424 (/Fax. (021) 7874223, 7874224, 27609039
- Jl. Mampang Prapatan X/4 Jakarta 12790 ((021) 7975235, 7947115, 7901205
- Sentra Niaga Blok B.I/12 - Jl. A. Yani Bekasi (Samping HERO/Mc.D) ((021) 8853537

<http://www.nurulfikri.com> | email: info@nurulfikri.com

