

# Memahami RPM

**RPM adalah salah satu *package manager* yang sangat terkenal di Linux. Hadirnya RPM membuat instalasi program dan manajemen program menjadi lebih mudah. Memahami RPM akan membantu kita lebih menguasai manajemen program sistem kita.**

Linux pada awalnya terkenal sebagai sistem operasi untuk kalangan pecinta komputer yang senang mengutak atik sistem. Pada awalnya, kita pun tidak mengenal istilah distro Linux sehingga perakitan sistem Linux memang dilakukan secara manual. Merakit suatu sistem di atas suatu sistem mau tidak mau harus dijalani untuk mendapatkan sistem Linux sendiri. Hal ini bukan pekerjaan yang mudah. Dengan kondisi seperti ini, pengguna Linux akan terkelompok secara eksklusif.

Beberapa inisiator distro kemudian melahirkan satu bundel sistem Linux dan aplikasinya yang dikenal saat ini sebagai distro Linux. Contoh terbaik untuk distribusi-distribusi Linux kuno yang tetap bertahan sampai saat ini adalah Slackware, Debian dan Red Hat Linux. Dalam perjalanan Linux sampai hari ini, ratusan distro Linux, baik yang sempat terkenal ataupun tidak, telah berusaha mengemas Linux agar menjadi sistem yang lebih mudah digunakan.

Untuk bertahan, distro harus berjuang keras. Paling tidak, untuk distro umum misalnya, pembuatnya harus memeriksa suatu program, menyesuaikan dengan distronya, memeriksa update terbaru, mengaplikasikan patch, menyediakan kemungkinan komunitas, menyediakan update, dan lain sebagainya. Bukan pekerjaan yang mudah untuk tetap bertahan. Suatu distro yang baik harus pula sesuai dengan standar Linux agar penggunaanya tidak kebingungan manakala berpindah dari distro lain atau ingin berpindah ke distro lain.

Manajemen paket program harus pula diperhatikan. Sistem Linux lebih menyukai sistem yang bersih, yang kalau bisa pustakanya tersentral dan bisa digunakan oleh siapa saja. Oleh karena itu, pendekatan instalasi dan manajemen program seperti Windows tidak disukai. Mudah digunakan, namun ada kemungkinan terjadinya redun-

dansi pustaka dan berbagai hal lainnya.

Dahulu kala, manajemen program sangat ditentukan oleh penggunanya. Tidak ada prosedur resmi untuk manajemen program. Pengguna bisa mengambil *source code free software* di internet dan melakukan kompilasi sendiri, dan setelah itu bisa melakukan instalasi. Rangkaian tindakan ambil, kompilasi, instalasi dan uninstalasi bisa menyebabkan sistem menjadi kotor dan tidak terintegrasi.

Untunglah para pengembang distro kemudian memikirkan cara yang lebih baik dalam mengatur instalasi dan manajemen program. Waktu itu, instalasi program di Windows pun tidak semudah saat ini. Boleh dikatakan, mereka memikirkan sendiri, atau sebisa mungkin melihat dari contoh yang ada di dunia UNIX lain.

Istilah paket program pun lahir. Secara sederhana, paket program adalah kumpulan file-file program dan *script* yang dikemas dalam satu file khusus. File khusus tersebutlah yang disebut sebagai paket program. Karena ingin menjaga sistem sekompak mungkin, maka paket haruslah sederhana dan hanya mengandung apa yang dibutuhkan. Oleh karena itu, suatu paket mungkin membutuhkan paket lain agar dapat bekerja. Kita menyebutnya sebagai *dependency*. Sebagai contoh, aplikasi *web browser* membutuhkan paket pustaka penanganan jaringan. Web browser tersebut tidak memasukkan sekaligus paket pustaka penanganan jaringan karena akan menyebabkan paket tidak kompak dan sistem mungkin tidak terintegrasi lagi. Dan, seperti yang disebutkan sebelumnya, suatu paket juga berisi script-script yang bisa dimanfaatkan untuk menyempurnakan proses instalasi atau uninstalasi paket program. Instalasi paket harus tidak menyebabkan sistem menjadi konflik karena pustaka misalnya. Uninstalasi paket harus tidak menyisakan sampah

yang akhirnya bisa menyebabkan sistem kotor atau konflik. *Upgrade* paket harus memungkinkan dan pintar.

Lantas, lahirlah RPP, PMS dan PM yang menjadi cikal bakal RPM saat ini. Dari dunia Debian, manajemen DEB juga dikembangkan.

Bicara soal RPM dan sejarahnya yang panjang dan rumit, banyak hal menarik yang bisa dipelajari. Mari kita melihat sejenak sejarah singkatnya.

RPP adalah cikal bakal awal RPM yang digunakan pada distribusi-distribusi Red Hat yang pertama. Saat itu, RPP tampil cukup inovatif dengan fitur berikut:

- Sederhana dalam instalasi dan uninstalasi paket.
- Memungkinkan script untuk dijalankan sebelum dan sesudah instalasi dan uninstalasi paket.
- Verifikasi paket untuk memeriksa keabsahan suatu paket.
- Memiliki fasilitas query.

PMS kemudian hadir untuk menyempurnakan RPP. Namun, PMS dikembangkan oleh pihak lain, ketika Red Hat sangat disibukkan oleh RPP. PMS mengenal konsep membangun paket dari source asal dan kemudian memberikan *patch*. Suatu konsep yang tidak dimiliki oleh RPP saat itu. Hal ini pulalah yang pada akhirnya memberikan kontribusi penting untuk RPM. Sayangnya, PMS memiliki banyak kekurangan seperti lemahnya *query*, tidak adanya verifikasi paket, tidak memungkinkan *multi architecture* dan desain database paket sistem yang lemah.

Rick Faith, yang memimpin pengembangan PMS, bersama Doug Hoffman, kemudian dikontak oleh Red Hat Software untuk mengembangkan PM. Tujuannya adalah mengkombinasikan yang terbaik terbaik dari RPP dan PMS. Sayangnya, pada ak-

hirnya, PM masih menyisakan kekurangan desain database paket sistem yang buruk dan tidak memungkinkan adanya multi architecture. Kabar yang buruk sekali adalah: PM tidak pernah digunakan dalam distribusi komersial. Sudah bagus, namun belum cukup bagus.

Dedengkot Red Hat, Marc Ewing dan Erik Troan pun turun tangan. Mereka menyebut percobaan lanjutan ini sebagai Red Hat Package Manager. Mereka mengambil yang terbaik-terbaik dari RPP, PMS, dan PM dan berhasil mengembangkan sistem yang lebih baik. RPM versi 1 ini ditulis dalam bahasa Perl untuk kecepatan pengembangan. Berikut ini adalah beberapa fitur penting RPM v1:

- Penanganan file konfigurasi. Hal ini diperlukan terutama dalam upgrade dan uninstallasi.
- Mudah untuk digunakan, dan sekaligus mudah dalam pembuatan paket program.

Sayangnya, yang satu ini juga memiliki banyak kekurangan. Tercatat adalah:

- Lambat karena menggunakan bahasa Perl. Penulisan ulang dalam bahasa C dibutuhkan.
- Masih juga dengan desain database yang lemah.
- Terlalu besar dan boros karena membutuhkan Perl. Hal ini paling terasa dalam instalasi sistem karena dahulu, instalasi dilakukan dari disket.
- Tidak mendukung multi architecture.
- Format paket yang kaku.

RPM versi 2 pun dikembangkan dan memiliki fitur berikut ini:

- Cepat. Ditulis ulang dengan bahasa C.
- Kecil karena tidak lagi membutuhkan Perl.
- Lahirnya rpmlib.
- Format paket yang lebih baik.
- Memungkinkan multi architecture.
- Desain database yang lebih disempurnakan.

Namun, tercatat RPM versi 2 pun memiliki kekurangan. RPM versi 3 dan 4 lahir menyempurnakan semua kekurangan RPM. Saat ini, RPM versi 4 telah digunakan secara meluas.

Dunia paket DEB pun memiliki banyak perbaikan. Perbaikan paling sukses adalah APT. Dengan adanya APT, dependency menjadi tidak masalah. Saat ini, APT tidak hanya dapat digunakan di Debian, namun juga dapat digunakan untuk RPM. Walau banyak distro besar berbasis RPM tidak memerlukan APT secara langsung karena konsepnya telah diimplementasikan dalam tool-tool spesifik distro. SUSE misalnya. Dengan YaST yang super canggih, auto resolve untuk dependency bukan masalah sama sekali.

Dari sisi penggunaan, perkembangan RPM terakhir kemudian memisahkan RPM dengan RPMBUILD. Hal ini sekaligus menjadikan RPM lebih modular. RPM dikhususkan untuk pengguna (pengguna akhir dan sysadmin). Sementara, RPMBUILD dikhususkan untuk pengembang paket. Kali ini, kita akan membahas penggunaan RPM.

## Menggunakan RPM

Kita akan membahas penggunaan RPM secara umum untuk instalasi, uninstallasi, upgrade dan query paket. Untuk instalasi, uninstallasi dan upgrade, Anda harus login sebagai root. Sementara, untuk query, hak root tidak dibutuhkan.

Dalam membahas instalasi atau upgrade paket, harap perhatikan untuk menggunakan paket yang benar dan cocok untuk distro Anda. Sebagai contoh, apabila menggunakan SUSE 9.1 misalnya, carilah paket untuk SUSE 9.1. Jangan cari paket untuk Red Hat 9.0 atau SUSE 9.0 (sebisanya mungkin).

Hal ini disebabkan karena pemaketan untuk setiap distro berbeda. Peletakan file setiap paket umumnya sudah sama, karena distro yang baik akan merujuk pada standar Linux. Namun, ada kalanya ada perbedaan pustaka. Ada kalanya juga perbedaan hal-hal spesial untuk distro. Sebagai contoh, script dalam paket di Red Hat berbeda dengan di SUSE.

Contoh kali ini akan mempergunakan paket contoh hellonop-0.1-0.i586.rpm.

## Instalasi program

Instalasi program secara sederhana dapat dilakukan dengan memberikan opsi `-i` untuk program rpm. Sebagai contoh:

```
rpm -i hellonop-0.1-0.i586.rpm
```

Anda bisa menambahkan opsi `-v` untuk menampilkan informasi lebih banyak lagi. Sebagai contoh:

```
rpm -iv hellonop-0.1-0.i586.rpm
```

Tambahkan lagi `-v` untuk informasi yang lebih banyak:

```
rpm -ivv hellonop-0.1-0.i586.rpm
```

Hanya, informasi yang terlalu banyak terkadang bisa membingungkan. Oleh karena itu, satu kali `-v` umumnya sudah cukup.

Pada paket berukuran besar, instalasi bisa berlangsung cukup lama. Apalagi jika paket berisikan sekian banyak script yang dijalankan. Untuk itu, Anda bisa memberikan indikator proses dengan menambahkan opsi `-h`:

```
rpm -ivh hellonop-0.1-0.i586.rpm
```

Umumnya, hanya opsi-opsi `-i`, `-v` dan `-h` tersebutlah yang perlu diberikan untuk instalasi paket program.

Berikut adalah beberapa opsi tambahan yang mungkin berguna dalam kasus tertentu.

- Pemberian opsi `--test` untuk menguji keberhasilan instalasi paket. Paket tidak akan benar-benar diinstall walaupun pesan sukses ditampilkan. Anda bisa menggunakan opsi ini untuk menguji berhasil tidaknya suatu paket diinstall di sistem Anda.
- Pemberian opsi `--excludedocs` untuk tidak menginstall dokumentasi paket. Apabila Anda hanya menginginkan *binary*, konfigurasi atau share data program namun tidak menginginkan dokumentasi paket, opsi ini bisa diberikan.
- Pemberian opsi `--noscripts` untuk tidak menjalankan script-script paket. Suatu paket bisa mengandung script preinstall (dijalankan sebelum instalasi), postinstall (dijalankan setelah instalasi), preuninstall (dijalankan sebelum uninstallasi) dan post-uninstall (dijalankan setelah uninstallasi). Apabila Anda benar-benar tahu apa yang Anda lakukan, opsi ini bisa sangat berguna. Beberapa paket melakukan pemeriksaan berlebihan dan terkadang menyebabkan. Paket hellonop-0.1-0.i586.rpm memiliki script preinstall dan postinstall. Apabila instalasi normal diberikan, berikut ini adalah contoh keluarannya:

```
rpm -ivh hellonop-0.1-0.i586.rpm
rpm
Preparing... ##### [100%]
#####
preparing for hellonop
instalation
1:hellonop ##### [100%]
#####
hello nop installed
```

Preinstall script dan postinstall script masing-masing akan mencetak preparing for hellonop instalation dan hello nop installed. Apabila diberikan opsi --noscripts, berikut ini adalah keluarannya:

```
# rpm -ivh --noscripts
hellonop-0.1-0.i586.rpm
Preparing... ##### [100%]
#####
1:hellonop ##### [100%]
#####
```

Pemberian opsi --nodeps untuk mengabaikan dependency. Harap hati-hati memberikan opsi ini. Terkadang berguna, namun lebih banyak tidak berguna. Sebagai contoh, ada kalanya kita tahu bahwa sebuah paket bisa bekerja dengan pustaka A versi 4. Sementara, paket yang kita inginkan cukup kuno dan menginginkan A versi 3. Karena kita benar-benar yakin paket yang kita install bisa bekerja dengan pustaka A versi 4, kita boleh saja melakukan instalasi dengan menggunakan opsi --nodeps. Hanya, perhatikan. Ini bukan alasan karena malas *resolve dependency* secara manual. Versi 4 tidak selalu backward compatibel dengan versi 3.

### Uninstalasi

Untuk uninstalasi, berikanlah opsi -e untuk rpm. Kita cukup menyebutkan nama paketnya saja. Atribut lain tidak perlu disebutkan. Contoh:

```
rpm -e hellonop
```

### Upgrade

Untuk upgrade, berikanlah opsi -U untuk rpm. Beberapa opsi lain mirip dengan opsi untuk instalasi program. Contoh:

```
rpm -Uvh hellonop-0.1-1.i586.rpm
```

### Query

Query adalah meminta informasi seputar paket atau database paket RPM. Seringkali,

```
Shell - Konsole
Session Edit View Bookmarks Settings Help
nop@linux:/usr/src/packages/RPMS/i586> rpm -qi hellonop-0.1-0.i586.rpm
Name       : hellonop      Relocations: /usr/local
Version    : 0.1          Vendor: Keant Systems
Release    : 0            Build Date: Sun 12 Sep 2004 12:33:28 AM WIT
Install date: (not installed) Build Host: linux.site
Group      : Amusements/Games/Toys Source RPM: hellonop-0.1-0.src.rpm
Size       : 2190349      License: GPL
Signature   : DSA/SHA1, Sun 12 Sep 2004 12:33:31 AM WIT, Key ID 404ffe07b11e98d0
Packager    : Noprianto <nop@keant.com>
Summary     : say hello to nop
Description :
This is a hellonop tool. Hellonop is tool for saying hello nop
to world. Its useful when people are being so busy and lost time
to say hello to its community.

Install this package to get hellonop message!
Distribution: Keant Tools
nop@linux:/usr/src/packages/RPMS/i586>
```

Query Paket RPM.

kita ingin tahu informasi detil suatu paket. Atau, kita ingin tahu apa saja file yang diinstall oleh suatu paket. Atau, paket A membutuhkan apa saja.

Anda perlu memberikan opsi -q terlebih dahulu, sebelum diikuti opsi lain.

Berikut ini adalah beberapa query yang umum dilakukan:

- Melihat informasi detil paket. Berikanlah opsi -q, lalu diikuti -i. Contoh:

```
$ rpm -qi hellonop
Name       : hellonop      Relocations: /usr/local
Version    : 0.1          Vendor: Keant Systems
Release    : 0            Build Date: Sun 12 Sep 2004 12:33:28 AM WIT
Install date: Sat 18 Sep 2004 02:00:43 PM WIT Build Host: linux.site
Group      : Amusements/Games/Toys
Source RPM: hellonop-0.1-0.src.rpm
Size       : 2190349      License: GPL
Signature   : DSA/SHA1, Sun 12 Sep 2004 12:33:31 AM WIT, Key ID 404ffe07b11e98d0
Packager    : Noprianto <nop@keant.com>
Summary     : say hello to nop
Description :
This is a hellonop tool.
Hellonop is tool for saying
hello nop
to world. Its useful when people
```

```
are being so busy and lost time
to say hello to its community.
```

```
Install this package to get
hellonop message!
```

```
Distribution: Keant Tools
```


- Melihat file apa saja yang diinstall oleh suatu paket. Berikanlah opsi -q, lalu ikuti dengan -l. Contoh:

```
$ rpm -ql hellonop
/usr/local/bin/hellonop
```

- Melihat apa saja yang dibutuhkan oleh suatu paket. Berikanlah opsi -q, lalu ikuti dengan --requires. Contoh:

```
$ rpm -q --requires hellonop
/bin/sh
/bin/sh
rpmLib(PayloadFilesHavePrefix)
<= 4.0-1
rpmLib(CompressedFileNames) <=
3.0.4-1
```

RPM adalah tool serbaguna dalam melakukan manajemen paket program. Pada kenyataannya, RPM adalah tool yang sangat hebat dan dirancang dengan hati-hati. Apabila vendor paket memiliki departmen QC yang baik, maka kombinasi dengan RPM akan membuat instalasi program di Linux sangat mudah untuk dilakukan.

Sampai di sini pembahasan kita untuk RPM. Selamat mencoba dan berhasil!   
Noprianto (noprianto@infolinux.co.id)