Noprianto

Instalasi Program di Linux

Seiring dengan semakin banyaknya Linux digunakan, teknologi yang memudahkan instalasi program di Linux pun bertambah. Kita akan membahasnya di tulisan ini.

Suatu sistem yang solid adalah impian semua pengguna komputer. Baik pengguna komputer di server besar sampai perangkat genggam. Permasalahannya adalah sistem operasi yang telah terinstal tidak dapat memenuhi kebutuhan semua pengguna. Baik saat pertama penggunaan ataupun di waktu yang akan datang. Oleh karena itu, sistem operasi modern mengizinkan instalasi aplikasi tambahan, baik yang berjalan sepenuhnya di *user space* ataupun yang menyentuh *kernel space*.

Instalasi program dapat dilakukan dengan berbagai cara. Di sistem operasi Windows misalnya, kita umum mengenal *installer* aplikasi yang datang di antaranya dengan program setup.exe (plus sejumlah file pendukung) ataupun satu *executable* tunggal yang *self-extract*. Kita juga umum menjumpai distribusi program yang dipaketkan dalam format Windows Installer (.msi).

Di berbagai distro Linux, kita juga mengenal banyak *package management*. Sebagai contoh, di sistem berbasis RPM, kita mengenal program yang dipaketkan sebagai file dengan ekstensi rpm yang dapat diinstal. Atau, di sistem berbasis DPKG, kita mengenal program yang dipaketkan sebagai file dengan ekstensi .deb. Dan masih banyak lagi.

Khususnya di Linux, masalah instalasi program menjadi sangat istimewa karena adanya keinginan untuk mengumpulkan semua pustaka di satu atau beberapa tempat yang disepakati bersama. Demikian juga dengan lokasi executable program. Di Linux, kita tidak disarankan untuk menggunakan lokasi instalasi a la Program Files di Windows, di mana sebagian besar aplikasi datang dengan file .exe dan .dll yang tersimpan di folder masing-masing. Akibatnya akan sangat mungkin terdapat redundansi file pustaka di sistem. Di Linux, kita ingin semua pustaka di simpan di /lib, /usr/lib atau lainnya. Binary executable disimpan di /bin, /usr/bin, dan lainnya. Aplikasi pun tidak disarankan untuk membawa pustaka sendiri-sendiri. Selain karena alasan tidak ingin adanya redundansi yang tidak diperlukan, juga karena kemungkinan terjadinya inkompatibilitas versi pustaka.

Oleh karena itu, sebuah paket barangkali akan membutuhkan paket lainnya. Dan, paket tersebut juga mungkin dibutuhkan oleh paket lainnya. Beberapa package management modern sudah mengatasi masalah *dependency* ini, namun sempat menjadi isu yang sangat rumit di beberapa waktu yang lalu ataupun di beberapa sistem saat ini.

Kebutuhan akan versi pustaka yang berbeda-beda juga bisa menjadi masalah besar. Dan, kerumitan masih harus ditambah dengan *user interface* package management yang terkadang masih harus dipelajari lagi.

Hal-hal seperti ini terkadang membuat user (terutama yang telah terbiasa dengan instalasi program yang relatif sederhana di Windows) menjadi susah untuk memahami Linux.

Seiring dengan waktu berjalan, komunitas pengguna, pengembang dan bisnis di

Linux telah mencoba untuk menghadirkan sistem instalasi program yang lebih mudah dilakukan di Linux. Tulisan ini mencoba untuk membahas beberapa teknologi yang bisa digunakan, disamping package management natif distribusi Linux yang digunakan.

Pembahasan akan kita mulai dengan package management natif. Package management yang akan dibahas adalah RPM, DPKG dan TGZ slackware.

Setelah itu, pembahasan akan kita lanjutkan dengan tool yang dapat menghasilkan installer program yang lebih mudah digunakan. Beberapa dari installer yang dihasilkan sudah semudah installer aplikasi di Windows. Namun, beberapa dari mereka juga membawa konsekuensi masing-masing.

Bagi *developer* yang lebih percaya dengan melakukan kompilasi sendiri, kami pun akan membahas berbagai trik agar instalasi program dapat dilakukan dengan lebih mudah dan terkontrol.

Sebagai penutup, kami juga akan membahas tool-tool yang bisa digunakan untuk bekerja dengan paket program berbagai distribusi.

Selamat membaca!

PACKAGE MANAGEMENT BERBAGAI DISTRO

Di dunia distribusi Linux, terdapat cukup banyak package management. Di antaranya adalah:

- RPM, merupakan singkatan dari RPM Package Manager (dulu: Red Hat Package Manager).
- DPKG, kependekan dari Debian package.
- Slackware package management.
- Portege.
- Dan lain sebagainya.

Setiap package management menawarkan fitur yang berbeda-beda. Walau demikian, pada dasarnya fungsinya adalah sama. Package management bertugas di antaranya untuk mengotomatisasi proses instalasi, *upgrade*, konfigurasi dan penghapusan paket.

Untuk setiap package management, kita akan membahas bagaimana mengetahui informasi yang terkandung di dalam suatu file paket, termasuk bagaimana melakukan instalasi, upgrade, dan menghapus paket.

Terakhir, kita juga akan membahas bagaimana resolusi dependency dilakukan untuk package management tersebut.

RPM

RPM merupakan package management yang sangat populer di dunia Linux. Banyak sekali distribusi Linux yang menggunakan package management ini. Bahkan, sistem yang dikembangkan awalnya oleh Red Hat ini sudah pula di port ke sistem operasi lain, seperti Novell Netware (versi 6.5 SP3) dan IBM AIX (versi 5).

Berikut ini adalah beberapa distribusi yang menggunakan RPM:

- RHEL (dan turunan seperti CentOS) dan Fedora.
- SLED, SLES dan OpenSUSE.
- Mandriva.
- PCLinuxOS.

RPM sendiri merujuk kepada format paket dan nama program untuk bekerja dengan file paket.

Nama file paket RPM

Berikut ini adalah pola nama file paket RPM:

```
<name>-<version>-<release>.<arch>.
rpm
```

Contoh nama file:

```
zoo-2.10-895.i586.rpm
```

Penjelasan field:

- <name> merupakan nama paket. Pada contoh tersebut, <name> adalah zoo.
- <version> merupakan versi dari *upstream* developer. Pada contoh tersebut, <version> adalah 2.10
- <release> merupakan rilis paket. Pada contoh tersebut, <release> adalah 895. Umumnya, bagian ini juga sering ditambahkan dengan kode distribusi, seperti fc4 (untuk fedora core 4), mdv (untuk mandriva), dan lain sebagainya.
- <arch> merupakan arsitektur target. Pada contoh tersebut, <arch> adalah i586. Contoh arsitektur lainnya adalah i386, i686, ppc, dan lainnya. Untuk paket yang tidak tergantung pada arsitektur, umumnya diisikan sebagai noarch. Contoh paket yang bersifat demikian adalah shell script, python script, file dokumentasi, gambar, dan lainnya.

Apabila source code dipaketkan sebagai paket rpm, maka pola nama file menjadi:

```
<name>-<version>-<release>.src.rpm
```

Informasi dari file paket

Informasi umum

Untuk mendapatkan informasi umum dari file paket, berikanlah perintah berikut:

```
rpm -qp --info <file_paket_rpm>
```

Contoh:

```
$ rpm -qp --info zoo-2.10-895.
i586.rpm

Name        : zoo
   Relocations: (not relocatable)
Version     : 2.10
   Vendor: SUSE LINUX Products
   GmbH, Nuernberg, Germany
Release     : 895
   Build Date: Sun 26 Nov 2006
   10:56:29 AM WIT
Install Date: (not installed)
   Build Host: cimarosa.suse.de
```

```
Group       : Productivity/
Archiving/Compression
   Source RPM: zoo-2.10-895.src.
rpm
Size        : 111912
   License: Public Domain, Freeware,
   Other License(s), see package
Signature   : DSA/SHA1, Sun 26
Nov 2006 11:07:29 AM WIT, Key ID
a84edae89c800aca
Packager    : http://bugs.
opensuse.org
Summary     : Pack Program
Description :
Zoo is a packer based on the
Lempel-Ziv algorithm. Lots of
files on DOS/AmigaDOS and TOS
systems used this packer for
their archives. The compression
rate of gzip is not reached, and
thus zoo should only be used for
decompressing old archives.

Authors:
--------
   Ian Phillipps   <igp@camcon.
co.uk>
   J. Brian Waters <jbwaters@bsu-
cs.bsu.edu>
   Paul Homchick
<rutgers!cgh!paul>
   Mark Alexander
<amdahl!drivax!alexande>
   Randal L. Barnes   <rlb@skyler.
mavd.honeywell.com>
Distribution: openSUSE 10.2 (i586)
```

Informasi kebutuhan

Untuk melihat apa yang dibutuhkan suatu paket, berikanlah perintah:

```
rpm -qp --requires <file_paket_
rpm>
```

Contoh:

```
$ rpm -qp --requires zoo-2.10-895.
i586.rpm
rpmlib(PayloadFilesHavePrefix) <=
4.0-1
rpmlib(CompressedFileNames) <=
3.0.4-1
libc.so.6
libc.so.6(GLIBC_2.0)
libc.so.6(GLIBC_2.1)
libc.so.6(GLIBC_2.3)
libc.so.6(GLIBC_2.3.4)
libc.so.6(GLIBC_2.4)
rpmlib(PayloadIsBzip2) <= 3.0.5-1
```

Instalasi dan upgrade paket

Untuk melakukan instalasi file paket, berikanlah perintah berikut:

```
rpm -i <file_paket_rpm>
```

contoh:

```
# rpm -i zoo-2.10-895.i586.rpm
```

Untuk melakukan upgrade file paket, berikanlah perintah berikut:

```
rpm -U <file_paket_rpm>
```

Menghapus paket terinstal

Untuk menghapus paket terinstal, berikanlah perintah berikut:

```
rpm -e <paket_terinstall>
```

Contoh:

```
# rpm -e zoo
```

Melihat file yang datang bersama paket

Untuk melihat file yang datang bersama suatu paket yang telah terinstall, berikanlah perintah berikut:

```
rpm -q --filesbypkg <paket_
terinstall>
```

Contoh:

```
$ rpm -q --filesbypkg zoo
zoo   /usr/bin/fiz
zoo   /usr/bin/zoo
zoo   /usr/share/doc/packages/zoo
zoo   /usr/share/doc/packages/zoo/
Copyright
zoo   /usr/share/man/man1/fiz.1.gz
zoo   /usr/share/man/man1/zoo.1.gz
```

Melihat paket pemilik file

Untuk melihat paket pemilik suatu file, berikanlah perintah berikut:

```
rpm -qf <path>
```

Contoh:

```
$ rpm -qf /bin/ls
coreutils-6.4-10
```

Resolusi dependency

Seperti kita bahas sebelumnya, sebuah paket bisa membutuhkan satu atau lebih paket (yang bisa membutuhkan satu atau lebih paket, dan seterusnya) serta dapat dibutuhkan oleh paket lainnya. Apabila hirarkinya menjadi sangat kompleks, maka akan sangat repot apabila resolusi dependency dilakukan secara manual.

Berikut ini adalah beberapa tool yang dapat digunakan untuk melakukan resolusi dependency secara otomatis:

- up2date (Red Hat Update Agent). Digunakan oleh RHEL, CentOS dan versi awal Fedora Core. Daftar pencarian paket disimpan di /etc/sysconfig/rhn/sources.
- Yum (Yellow dog Update, Modified). Digunakan, di antaranya di Fedora, CentOS, Scientific Linux, dan Yellow Dog Linux. Beberapa front end grafikal yang tersedia untuk yum adalah pup, pirut, dan yumex.
- YaST (Yet another setup tool). Digunakan di SUSE.
- Urpmi. Digunakan di Mandriva.
- Apt-rpm. Merupakan versi APT yang dimodifikasi untuk bekerja dengan RPM. Informasi selengkapnya: *http://apt-rpm.org/*.

DPKG

dpkg merupakan package management yang digunakan di distro debian GNU/Linux. Package management ini dikembangkan pada tahun 1993. Karena sangat banyak distribusi yang diturunkan dari Debian GNU/Linux, maka dengan sendirinya, banyak distribusi yang menggunakan package management ini.

Berikut ini adalah beberapa contoh turunan Debian GNU/Linux:

- Ubuntu dan variannya/turunannya.
- Knoppix dan turunannya.
- MEPIS.
- Xandros.

Ekstensi nama file paket Debian adalah .deb.

Nama file paket Debian

Berikut ini adalah pola nama file paket debian:

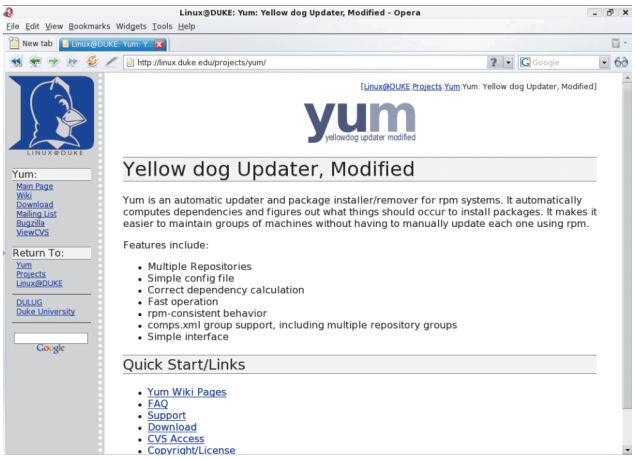
```
<name>_<version>-<debian-
revision>.deb
```

Contoh nama file:

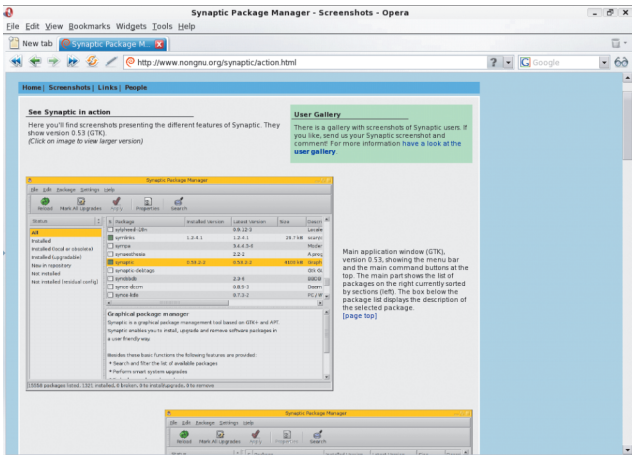
```
bzip2_1.0.2-7_i386.deb
```

Penjelasan field:

- <name> merupakan nama paket. Pada contoh tersebut, <name> adalah bzip.



Situs web yum.



Situs web synaptic.

- <version> merupakan versi dari upstream developer. Pada contoh tersebut, <version> adalah 1.0.2.
- <debian-revision> merupakan nomor revisi. Pada contoh tersebut, <debian-revision> adalah 7_i386.

Informasi dari file paket

Informasi umum

Untuk mendapatkan informasi umum dari file paket, berikanlah perintah berikut:

```
dpkg -I <file_paket_deb>
```

Contoh:

```
$ dpkg -I bzip2_1.0.2-7_i386.deb
 new debian package, version 2.0.
 size 233356 bytes: control
archive= 1324 bytes.
    1014 bytes,    25 lines
control
     582 bytes,    24 lines   *
postinst             #!/bin/sh
     543 bytes,    20 lines   *
preinst              #!/bin/sh
     292 bytes,    16 lines   *
prerm                #!/bin/sh
 Package: bzip2
 Version: 1.0.2-7
 Section: utils
 Priority: optional
 Architecture: i386
 Depends: libbz2-1.0, libc6 (>=
2.3.2.ds1-21)
 Replaces: libbz2 (<< 0.9.5d-3)
...
...
```

Informasi kebutuhan

Untuk melihat apa yang dibutuhkan suatu paket, berikanlah perintah:

```
dpkg -f <file_paket_deb> depends
```

Contoh:

```
$ dpkg -f bzip2_1.0.2-7_i386.deb
depends
libbz2-1.0, libc6 (>= 2.3.2.ds1-21)
```

Instalasi dan upgrade paket

Untuk melakukan instalasi/upgrade file paket, berikanlah perintah berikut:

```
dpkg -i <file_paket_deb>
```

contoh:

```
$ dpkg -i zip_2.31-1_i386.deb
Selecting previously deselected
package zip.
(Reading database ... 89845
files and directories currently
installed.)
Unpacking zip (from .../main/z/
zip/zip_2.31-1_i386.deb) ...
Setting up zip (2.31-1) ...
```

Menghapus paket terinstall

Untuk menghapus paket terinstal, berikanlah perintah berikut:

```
dpkg -r <paket_terinstall>
```

Contoh:

```
$ dpkg -r zip
(Reading database ... 89860
files and directories currently
installed.)
Removing zip ...
```

Untuk menghapus paket terinstal beserta file konfigurasinya, berikanlah perintah:

```
dpkg -P <paket_terinstall>
```

Contoh:

```
$ dpkg -P zip
(Reading database ... 89860
files and directories currently
installed.)
Removing zip ...
```

Melihat file yang datang bersama paket

Untuk melihat file yang datang bersama suatu paket yang telah terinstall, berikanlah perintah berikut:

```
dpkg -L <paket_terinstall>
```

Contoh:

```
$ dpkg -L bzip2
/.
/usr
/usr/bin
/usr/bin/bzip2
/usr/bin/bunzip2
/usr/bin/bzcat
/usr/bin/bzip2recover
/usr/bin/bzexe
/usr/bin/bzgrep
/usr/bin/bzegrep
/usr/bin/bzfgrep
/usr/bin/bzmore
...
...
```

Melihat paket pemilik file

Untuk melihat paket pemilik suatu file, berikanlah perintah berikut:

```
dpkg -S <path>
dpkg -S <pola_path>
```

Contoh:

```
$ dpkg -S /bin/ls
coreutils: /bin/ls
```

Resolusi dependency

Debian package management dikenal sebagai yang pertama yang datang dengan kemampuan *dependency resolution* melalui APT (*Advanced Packaging Tool*). APT dapat melakukan sangat banyak hal di luar dependency resolution.

Berbeda dengan dpkg yang juga bekerja dengan file paket, APT bekerja dengan nama paket. Daftar source paket tersimpan di /etc/apt/sources.list.

Berikut ini adalah beberapa contoh penggunaan APT.

Instalasi paket

apt-get install <pkgname1> [pkgname2 ...]

Contoh:

```
# apt-get install zip
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be
installed:
  zip
0 upgraded, 1 newly installed, 0
to remove and 0 not upgraded.
Need to get 0B/99.6kB of archives.
After unpacking 250kB of
additional disk space will be
used.
Selecting previously deselected
package zip.
(Reading database ... 89845
files and directories currently
installed.)
Unpacking zip (from .../archives/
zip_2.31-1_i386.deb) ...
Setting up zip (2.31-1) ...
```

Menghapus paket terinstal

apt-get remove <pkgname1> [pkgname2 ...]

Contoh:

```
# apt-get  remove zip
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be
REMOVED:
  zip
0 upgraded, 0 newly installed, 1
to remove and 0 not upgraded.
Need to get 0B of archives.
After unpacking 250kB disk space
will be freed.
Do you want to continue? [Y/n] y
(Reading database ... 89860
files and directories currently
installed.)
Removing zip ...
```

Untuk memudahkan penggunaan, user dapat menggunakan beberapa *front end* berikut:

- aptitude (Text user interface).
- Synaptic Package Manager (GUI).
- Adept Package Manager (GUI).

TGZ/Slackware package management

Dibandingkan dengan package management distro-distro lain, package management yang digunakan oleh Slackware jauh lebih sederhana. Selain digunakan oleh Slackware, package management ini digunakan oleh turunannya.

Berikut ini adalah beberapa contoh turunan Slackware:

- Vector Linux.
- Zenwalk Linux.
- Slax.

Ekstensi nama file paket slackware adalah .tgz. Paket slackware merupakan paket tar.gz biasa yang ditambahkan deskripsi paket dan shell script (post-install; dijalankan setelah instalasi selesai dilakukan).

Berbeda dengan paket distro lain, paket slackware umumnya merupakan paket monolitik, di mana satu paket telah mengandung semua yang dibutuhkan, termasuk *development* file (apabila ada), dokumentasi dan lainnya.

Nama file paket slackware

Berikut ini adalah pola nama file paket slackware:

```
<name>-<version>-<arch>-
<buildnumber>.tgz
```

Contoh nama file:

```
xmms-1.2.10-i486-3.tgz
```

Penjelasan field:

- <name> merupakan nama paket. Pada contoh tersebut, <name> adalah xmms.
- <version> merupakan versi dari upstream developer. Pada contoh tersebut, <version> adalah 1.2.10.
- <arch> adalah arsitektur. Pada contoh tersebut, <arch> adalah i486.
- <buildnumber> merupakan nomor build. Pada contoh tersebut, <buildnumber> adalah 3.

Informasi dari file paket

Untuk mendapatkan informasi umum dari file paket, shell script slackinfo.sh berikut ini bisa digunakan:

```
#!/bin/sh

#(c) Nop, GPLv2.

if [ -z "$1" ]
then
        echo "usage: $0 <package_
file>"
        exit 1
fi


PKGBASENAME=`basename "$1"`
PKGNAME=`echo $PKGBASENAME | cut
-d- -f1`

tar -xf "$1" install/slack-desc
[ $? -ne 0 ] && exit 2

cat install/slack-desc | grep
"$PKGNAME:"
[ $? -ne 0 ] && exit 3

rm -f install/slack-desc
rmdir install
```

Berikanlah hak akses executable dengan perintah:

```
$ chmod +x slackinfo.sh
```

Cara penggunaan:

```
slackinfo.sh <package_file>
```

Contoh:

```
$ ./slackinfo.sh xmms-1.2.10-i486-
3.tgz
xmms: xmms (X Multimedia System)
xmms:
xmms: XMMS is the X Multimedia
System.  It is used to play audio
and other
xmms: kinds of media files.  By
default XMMS can play MPEG audio,
Ogg
xmms: Vorbis, RIFF wav, most
module formats, and a few other
```

```
formats.  XMMS
xmms: can be extended through
plugins to play a number of other
audio and
xmms: video formats.
xmms:
xmms:
xmms:
xmms:
```

Instalasi dan upgrade paket

Untuk melakukan instalasi file paket, berikanlah perintah berikut:

```
installpkg  <file_paket_tgz>
```

contoh:

```
# installpkg xmms-1.2.10-i486-
3.tgz
Installing package xmms-1.2.10-
i486-3...
PACKAGE DESCRIPTION:
xmms: xmms (X Multimedia System)
xmms:
xmms: XMMS is the X Multimedia
System.  It is used to play audio
and other
xmms: kinds of media files.  By
default XMMS can play MPEG audio,
Ogg
xmms: Vorbis, RIFF wav, most
module formats, and a few other
formats.  XMMS
xmms: can be extended through
plugins to play a number of other
audio and
xmms: video formats.
xmms:
Executing install script for xmms-
1.2.10-i486-3...
```

Untuk melakukan upgrade file paket, gunakanlah program upgradepkg. Bacalah manual penggunaan untuk informasi selengkapnya.

Menghapus paket terinstal

Untuk menghapus paket terinstal, berikanlah perintah berikut:

```
removepkg <paket_terinstall>
```

Contoh:

```
# removepkg xmms

Removing package /var/log/
packages/xmms-1.2.10-i486-3...
Removing files:
  --> Deleting symlink /usr/lib/
libxmms.so
  --> Deleting symlink /usr/lib/
libxmms.so.1
  --> Deleting /usr/bin/wmxmms
  --> Deleting /usr/bin/xmms
  --> Deleting /usr/bin/xmms-arts-
helper
  --> Deleting /usr/bin/xmms-
config
...
...
```

Melihat file yang datang bersama paket

Untuk melihat file yang datang bersama suatu paket yang telah terinstal, bacalah file /var/log/packages/<name>-<version>-<arch>-<buildnumber>.

Resolusi dependency

Berbeda dengan distro lain, package management slackware tidak datang dengan penanganan dependency. Slackware menyerahkan sepenuhnya kepada user. Instalasi dapat dilakukan walaupun paket yang dibutuhkan belum terinstall. Hanya, ketika aplikasi dijalankan, barulah pesan kesalahan (bahwa terdapat pustaka yang tidak ditemukan misalnya) akan ditampilkan.

Walau demikian, terdapat beberapa project yang dapat digunakan untuk menyelesaikan masalah dependency. Berikut ini adalah beberapa di antaranya:

- Swaret, *http://swaret.sourceforge.net/*.
- slapt-get, *http://software.jaos.org/#slapt-get*.
- Emerde, *http://emerde.freaknet.org/*.
- slackpkg, *http://slackpkg.sourceforge.net/*.

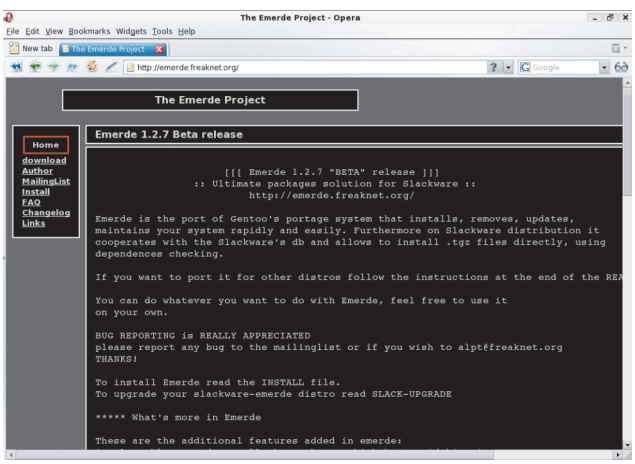
INSTALASI DENGAN TOOL PIHAK KETIGA

Di Linux, saat ini kita bisa menikmati berbagai tool yang bisa menghasilkan installer yang mudah dijalankan oleh user. Bahkan semudah installer aplikasi di Windows. User hanya tinggal klik *next*, next, next, *finish*, dan aplikasi sudah terinstal. Tool-tool tersebut bisa pula menghasilkan installer yang dapat dijalankan tanpa harus membutuhkan hak root. Dengan demikian, untuk sekedar menginstal word processor misalnya, user biasa pun bisa melakukannya.

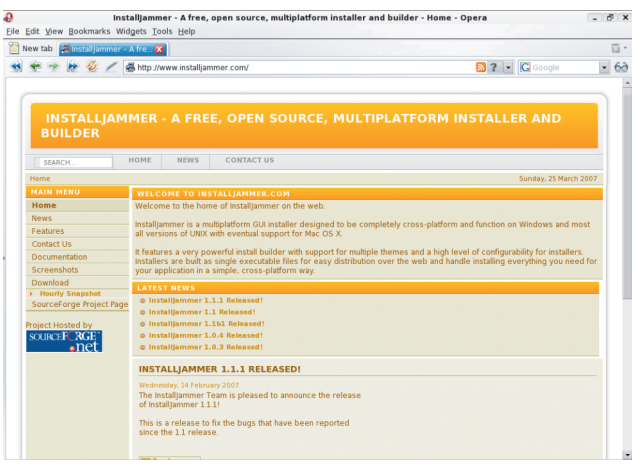
Di bagian ini, kita akan membahas beberapa dari tool-tool tersebut.

InstallJammer

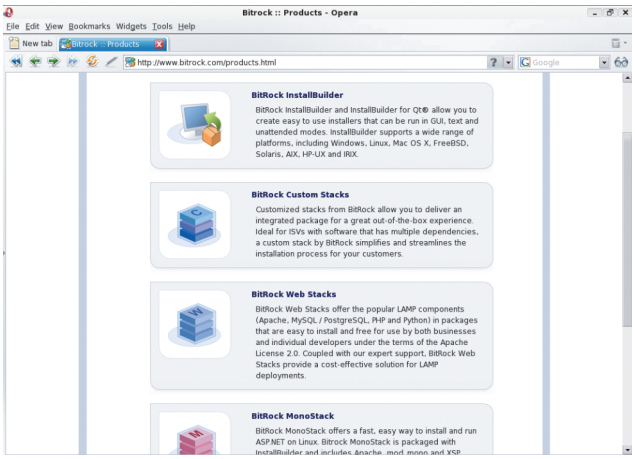
InstallJammer adalah *installation development tool* GPL yang dapat menghasilkan installer berbasis GUI. InstallJammer bekerja cross platform dan mendukung Windows, Linux, sebagian besar versi UNIX dan sistem operasi lainnya.



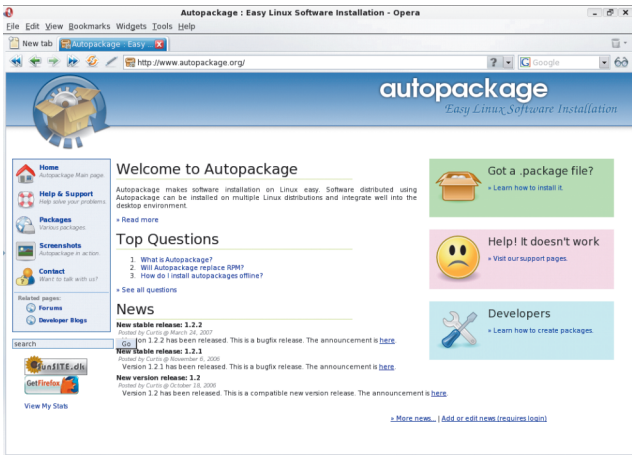
Situs web emerde.



Situs web installjammer.



Situs web bitrock installbuilder.



Situs web autopackage.

Di Linux, installer yang dihasilkan merupakan sebuah file *executable statically linked* dan *self-contained* yang dapat berjalan di hampir semua distribusi Linux. Secara default, installer berjalan di GUI, namun apabila tidak tersedia, dapat pula berjalan di *text mode*. Installer yang dihasilkan juga mendukung *silent install* dengan pemberian prefix instalasi.

User yang menjalankan installer dapat melakukan instalasi dengan sangat mudah karena hanya perlu memilih folder tujuan, klik next, next, finish dan program pun siap digunakan.

Berikut ini adalah beberapa fitur lain InstallJammer:

- datang dengan Install Builder yang *powerful* dan mudah digunakan.
- Mendukung kustomisasi installer.
- Mendukung banyak bahasa.
- Dapat menghasilkan installer yang kecil dan cepat.
- Mendukung berbagai aksi *built-in* seperti deteksi runtime Java, dan lainnya
- Mendukung pengeditan dialog setiap langkah instalasi.
- Mendukung *theme*.
- Mendukung pembuatan *uninstaller* otomatis.

Fitur selengkapnya bisa dibaca di website-nya, *http://www.installjammer.com*.

Berikut ini adalah keluaran program file untuk installer InstallJammer:

```
$ file InstallJammer-1.1.2-Linux-
x86-Install
InstallJammer-1.1.2-Linux-
x86-Install: ELF 32-bit LSB
executable, Intel 80386, version
1, statically linked, stripped
```

Keluaran tersebut memperlihatkan bahwa file tersebut adalah executable Linux dalam kondisi *statically linked* (di-link secara statis; tidak membutuhkan pustaka tambahan).

Penulis sudah beberapa saat lamanya menggunakan InstallJammer dan secara umum sangat puas dengan installer yang dihasilkan.

BitRock Install Builder

BitRock Install Builder dapat menghasilkan installer GUI aplikasi untuk berbagai platform populer, termasuk Windows, Linux, Mac OS X, dan Solaris. Installer yang dihasilkan dapat dijalankan dengan mudah, dimana user hanya perlu klik next, next dan seterusnya.

Installer yang dihasilkan berjalan di modus GUI, namun apabila GUI tidak tersedia, secara default akan dijalankan pada modus teks. Di Linux, installer yang dihasilkan merupakan sebuah file executable statically linked dan self-contained yang dapat berjalan di hampir semua distribusi Linux.

Stack yang tersedia adalah:

- InstallBuilder/InstallBuilder for Qt.
- Custom stack.
- Web stack, untuk aplikasi berbasis Apache, MySQL/PostgreSQL, PHP/Python.
- Mono stack, untuk komponen Mono.

Lisensi install builder yang tersedia:

- Lisensi komersial, untuk membangun installer dengan tujuan komersial.
- Lisensi akademis, untuk membangun installer dengan tujuan akademis.
- Lisensi open source, untuk membangun installer untuk aplikasi open source .
- Lisensi upgrade.

Berikut ini adalah beberapa fitur lain Install Builder:

- Integrasi desktop.
- Integrasi database RPM.
- Dapat menghasilkan installer yang teroptimasi.
- Datang dengan aksi builtin seperti deteksi runtime Java dan lainnya.
- Mendukung kustomisasi.
- Mendukung pembuatan uninstaller otomatis.
- Mendukung banyak bahasa.

Fitur selengkapnya bisa dibaca di website-nya, *http://www.bitrock.com*.

Berikut ini adalah keluaran program file untuk installer Install Builder:

```
$ file installbuilder-4.1.0-linux-
installer.bin
installbuilder-4.1.0-linux-
installer.bin: ELF 32-bit LSB
executable, Intel 80386, version
1, statically linked, stripped
```

Keluaran tersebut memperlihatkan bahwa file tersebut adalah executable Linux dalam kondisi statically linked (di-link secara statis; tidak membutuhkan pustaka tambahan).

Autopackage

Autopackage adalah tool open source yang dapat menghasilkan paket instalasi berbagai aplikasi Linux yang mudah diinstal oleh user.

Berikut ini adalah beberapa fitur autopackage:

- Dapat menghasilkan paket untuk berbagai distribusi (paket yang dihasilkan adalah *bash script*).
- Mendukung berbagai front-end, termasuk GUI.
- Mendukung banyak bahasa.
- Resolusi dependency otomatis.

Fitur selengkapnya bisa dibaca di website-nya, *http://www.autopackage.org*.

Berikut ini adalah keluaran program file untuk salah satu paket yang dihasilkan (contoh: pawn, sebuah scripting language):

```
$ file pawn-3.2.3664.package
pawn-3.2.3664.package: Bourne-
Again shell script text executable
```

Keluaran tersebut memperlihatkan bahwa file tersebut adalah script bash.

MELAKUKAN KOMPILASI SENDIRI

Bagi pengguna Linux dan open source, melakukan kompilasi sendiri program *free/open source* selalu menjadi salah satu pilihan yang dapat diambil apabila tidak ingin tergantung dengan package management distro.

Selain itu, melakukan kompilasi sendiri memungkinkan optimasi sesuai dengan konfigurasi *hardware* yang dimiliki.

Berikut ini adalah beberapa tips untuk pengguna yang melakukan kompilasi sendiri:

- Gunakanlah *prefix* yang memungkinkan sistem tetap *clean*. Hindari prefix /usr dan gunakan prefix /usr/local apabila memungkinkan. Beberapa source code datang dengan makefile yang mendukung fasilitas uninstall, namun tidak semuanya.
- Apabila Anda menggunakan prefix custom seperti /opt/<program>, pastikan Anda tetap mudah mengakses binary aplikasi, termasuk halaman manual.
- Apabila Anda menginstal pustaka, pastikan kompilasi program lain yang membutuhkan pustaka tersebut bisa menemukan file header yang dibutuhkan (terutama untuk penggunaan *prefix custom*).
- Simpanlah source code yang telah terkonfigur apabila memiliki ruang kosong yang cukup.
- Untuk redistribusi hasil kompilasi, bangunlah paket untuk distribusi yang diinginkan. Lihatlah pembahasan berikut.

TOOL BERGUNA

Di bagian ini, kita akan membahas dua tool yang berguna untuk bekerja dengan paket distribusi.

Checkinstall

Bagi Anda yang melakukan kompilasi sendiri dari source code, checkinstall adalah tool yang sangat berguna. Tool ini bisa membantu untuk membangun paket RPM, Debian atau slackware dari hasil kompilasi.

Tanpa checkinstall, kita umumnya akan memberikan perintah semacam make install (prefix instalasi diasumsikan /usr) dan setelah itu, file-file akan tersebar di seluruh sistem. Apabila source code datang dengan makefile yang mendukung uninstall, maka proses uninstall masih bisa kita lakukan dan sistem masih berpeluang untuk tetap bersih. Namun, apabila source code tidak datang makefile yang mendukung uninstalasi, maka uninstall akan sangat merepotkan untuk dilakukan.

Dengan menggunakan checkinstall, kita akan memberikan argumen berupa perintah yang diperlukan untuk melakukan instalasi dari hasil kompilasi (misal: make install), dan dengan langkah-langkah yang sangat mudah, checkinstall akan menghasilkan paket RPM, Debian atau Slackware. Sistem tidak akan terganggu.

Setelah paket dihasilkan, dengan package management distribusi, kita dapat melakukan instalasi dan uninstalasi dengan mudah.

Informasi selengkapnya tentang checkinstall bisa didapatkan di website-nya, *http://asic-linux.com.mx/~izto/checkinstall/*.

Alien

Alien adalah program yang melakukan konversi paket-paket RPM, DPKG, Stampede slp dan TGZ slackware. Dengan demikian, kita bisa mengonversi paket yang tersedia untuk distro lainnya ke paket untuk distro yang kita gunakan (dan sebaliknya), selama format paket didukung oleh alien.

Alien dimaksudkan untuk melakukan konversi paket-paket yang nonesensial. Disamping fiturnya yang sangat berguna, alien masih memiliki sejumlah keterbatasan. Dan, berhubung aturan pemaketan setiap distribusi bisa sangat berbeda, selalu berhati-hatilah dalam melakukan instalasi paket hasil konversi.

Informasi selengkapnya tentang alien bisa didapatkan di website-nya, *http://kitenet.net/~joey/code/alien.html*.

TIPS UMUM INSTALASI APLIKASI

Berikut ini adalah beberapa tips yang mungkin berguna dalam instalasi aplikasi di Linux:

- Selalu gunakan paket yang dikhususkan untuk distribusi Anda. Gunakanlah paket untuk versi yang sesuai. Apabila Anda menggunakan SUSE, jangan gunakan paket dari Fedora, walaupun sama-sama RPM. Apabila Anda menggunakan SUSE versi 10, jangan gunakan paket dari versi 9.
- Khusus untuk pengguna Debian, jangan pernah mencampuradukkan *stable, testing,* dan *unstable*. Pengguna stable dapat mempergunakan berbagai backport apabila membutuhkan versi terbaru/cukup baru. Informasi mengenai berbagai backport yang tersedia bisa didapatkan diantaranya di *http://www.backports.org*. Alternatif lain bagi yang tidak ingin menggunakan backport adalah dengan melakukan kompilasi sendiri. Seperti kita ketahui, Debian datang dengan sangat banyak paket, termasuk untuk kebutuhan *development*.
- Gunakanlah package management distro untuk paket-paket esensial. Ini adalah cara yang disarankan. Dan, gunakanlah paket dari repositori distro yang digunakan (dari sumber terpercaya). Untuk paket tambahan, seperti word processor, apabila memang tidak memiliki pilihan lain (seperti menggunakan paket distro), Anda mungkin bisa menggunakan installer yang ada (yang dibangun dengan InstallJammer atau Install Builder misalnya).

Dengan memahami instalasi software di Linux, diharapkan, semakin sedikit hambatan yang kita hadapi untuk menggunakan Linux. Dan, kita pun bisa semakin produktif. Selamat mencoba!