52

Ohjelmapakettien hallinta

Fedora ja Centos Linuxissa on useita työkaluja ohjelmapakettien hallintaan. Perustyökalu on `rpm`, mutta sen käyttö on hankalaa, koska se ei osaa itse ratkaista riippuvuuksia. `yum` on kehittyneempi työkalu, joka ottaa riippuvuudet huomioon ja osaa itse hakea ja asentaa riippuvuuksien takia tarvittavat ohjelmapaketit. `yum` on kuitenkin komentoriviohjelma ja graafinen `system-config-packages` eli `pirut` on helpompi varsinkin, jos järjestelmään lisätään tai poistetaan ohjelmia.

Fedorassa `system-config-packages` on korvattu PackageKit-ohjelmistolla versiosta 9 lähtien.

Aloitamme tutustumisen pakettien hallintaan perustyökalusta, joka on `rpm`.

52.1 Pakettien hallintaa RPM:llä

Red Hat **P**ackage **M**anager (RPM) on avoin paketointijärjestelmä, jota kuka tahansa voi käyttää, ja se toimii yhtä hyvin Red Hat Linuxin kuin muiden UNIX-järjestelmien kanssa. Red Hat Software rohkaisee muita yrityksiä tutustumaan RPM:ään ja käyttämään sitä omissa tuotteissaan. RPM on levitettävissä GPL:n mukaisesti.

Loppukäyttäjälle RPM tarjoaa useita ominaisuuksia, jotka tekevät järjestelmän hallinnan helpoksi. Ohjelmien asentaminen, poistaminen ja päivittäminen RPM-paketteina on yhden ko-

mentorivin pituinen toimenpide ja kaikki hankalat yksityiskohdat (paitsi riippuvuuksien selvittäminen ja niiden tarvitsemien lisäpakettien asennus) on jo hoidettu. RPM ylläpitää tietokantaa asennetuista paketeista sekä niiden tiedostoista ja mahdollistaa monipuoliset kyselyt sekä pakettien tarkistukset. Päivitettäessä RPM huolehtii konfigurointitiedostoista siten, että käyttäjän tekemät muutokset säilyvät. Vastaava olisi mahdotonta tavallisilla arkistotiedostoilla, kuten `.tar.gz`-tiedostoilla.

Sovelluskehityksessä RPM:n avulla voidaan sama lähdekoodi pakata lähdekoodipaketiksi sekä binääripaketeiksi loppukäyttäjille. Prosessi on melko yksinkertainen ja sitä ohjaa yksi tiedosto sekä valinnaiset korjaustiedostot, joita paketin kokoaja voi tehdä. Selvä jako koskemattomien lähdekooditiedostojen, mahdollisten korjaustiedostojen sekä käännöskäskyjen välillä helpottaa pakettien ylläpitoa, kun uusia versioita ohjelmista ilmestyy.

52.2 RPM:n suunnittelutavoitteet

RPM:n suunnittelutavoitteiden ymmärtäminen auttaa myös sen käytössä.

Päivitettävyys RPM:n avulla voit päivittää osia järjestelmästä ilman täysasennusta. Uuden RPM-pohjaisen käyttöjärjestelmän (kuten Red Hat Linux) ilmestyessä ei tarvitse asentaa kaikkea, kuten yleensä tarvitsee muihin paketointijärjestelmiin pohjautuvissa järjestelmissä. RPM tarjoaa älykkään, automatisoidun ja suoraan entisten tilalle tapahtuvan päivityksen. Konfigurointitiedostot, jotka liittyvät paketteihin, säästetään, eikä asetusten muutokseen tehtyä työtä menetetä.

Monipuoliset kyselyt RPM:ään suunniteltiin myös monipuoliset kyselyominaisuudet. Voit etsiä koko tietokannasta tiettyjä paketteja tai yksittäisiä tiedostoja. Tiedostoista voidaan kysyä, mihin pakettiin se kuuluu ja mistä se on peräisin. RPM-paketissa tiedostot ovat pakatussa muodossa ja paketin alussa ovat tiedot paketista ja sen sisällöstä, jonka perusteella pakettien tiedot saadaan helpolla ja nopeasti.

Pakettien tarkistus Pakettien tarkistus on usein käyttökelpoinen ominaisuus. Jos epäilet, että olet poistanut tärkeän, johonkin pakettiin kuuluvan tiedoston, tarkista paketti. Tarkistus ilmoittaa kaikesta epätavallisesta. Jos ongelmia löytyy, voit asentaa paketin uudelleen. Kaikki muutetut konfigurointitiedostot säilyvät uudelleenasennuksessa.

Koskemattomat lähdekooditiedostot Tärkeä suunnitteluperuste oli alkuperäisten lähdekooditiedostojen käyttö sellaisenaan. RPM erittelee alkuperäisen lähdekoodin, mahdolliset korjaukset ja käännösohjeet. Tämä tarjoaa useita etuja. Jos esimerkiksi ohjelmasta ilmestyy uusi versio, uuden paketin kokoamista ei välttämättä tarvitse aloittaa alusta. Korjaustiedostoista voi tarkistaa, mitä aikaisemmin tehtiin, ja päätellä, tarvitaanko samoja muutoksia jälleen. Kaikki oletusarvojen muutokset ja muut toimenpiteet, joita

tarvittiin valmiin ohjelman tekoon, säilytetään erikseen ja ovat sen takia selvästi nähtävillä.

Tämä tavoite saattaa vaikuttaa tärkeältä vain pakettien kokoajille, mutta sen ansiosta myös loppukäyttäjille tarkoitettujen ohjelmien taso nousee.

52.3 RPM:n käyttö

RPM:llä on viisi perustoimintoa: asennus, poisto, päivitys, kysely ja tarkistus. Pakettien kokoaminen on siirretty erilliseen `rpmbuild`-ohjelmaan. Täydellisemmät ohjeet saa käskyllä `rpm --help`, RPM:n man-sivulta ja englanninkielisestä RPM-kirjasta (ks. luku 52.5 sivulla 601).

52.3.1 Asentaminen

RPM-paketin nimi voi olla esimerkiksi `foo-1.0-1.i386`, joka sisältää paketin nimen (`foo`), version (`1.0`), jakeluversion (`1`) ja arkkitehtuurin (`i386`). Paketti asentuu käskyllä:

```
$ rpm -ivh foo-1.0-1.i386.rpm
foo                         ########################################
```

Kuten huomaat, RPM tulostaa ensin paketin nimen (joka ei välttämättä ole sama kuin tiedoston nimi) ja sitten #-merkkejä paketin asentuessa eräänlaisena matkamittarina.

Pakettien asennus on helppoa, mutta joskus voi tulla eteen virhetilanteita:

Paketti on jo asennettu

Jos paketti on jo asennettu, tulee seuraavantapainen virheilmoitus:

```
$ rpm -ivh foo-1.0-1.i386.rpm
foo                       package foo-1.0-1 is already installed
error: foo-1.0-1.i386.rpm cannot be installed
```

Jos todella haluat asentaa paketin kuitenkin, voit käyttää komentorivillä parametriä `--force`, joka käskee RPM:ää olemaan välittämättä virheestä.

```
# rpm -ivh --force foo-1.0-1.i386.rpm
foo                         ########################################
#
```

Tiedostoristiriita

Jos yrität asentaa pakettia, jossa on tiedosto, joka on jo asennettu, näet virheilmoituksen:

```
# rpm -ivh foo-1.0-1.i386.rpm
foo           /usr/bin/foo conflicts with file from bar-1.0-1
error: foo-1.0-1.i386.rpm cannot be installed
```

Jotta RPM ohittaisi tämän virheen, voit käyttää jälleen parametriä `--force` komentorivillä.

```
# rpm -ivh --force foo-1.0-1.i386.rpm
foo                         ##################################
#
```

Selvittämätön riippuvuus

RPM-paketit voivat olla "riippuvaisia" toisista paketeista, mikä tarkoittaa, että toimiakseen kunnolla ne vaativat, että muita paketteja on jo asennettu. Yrittäessäsi asentaa pakettia, jossa on selvittämätön riippuvuus, näet virheilmoituksen:

```
$ rpm -ivh bar-1.0-1.i386.rpm
failed dependencies:
        foo is needed by bar-1.0-1
```

Tämän virheen selvittämiseksi on syytä asentaa tarvittavat paketit. Jos haluat silti pakottaa asennuksen tapahtuvaksi (huono ajatus, koska paketin ohjelma ei todennäköisesti toimi kunnolla), käytä parametriä `--nodeps` komentorivillä.

52.3.2 Poisto

Paketin poisto tapahtuu yksinkertaisesti:

```
$ rpm -e foo
```

Huomaa, että käytimme paketin *nimeä* "`foo`", eikä paketin alkuperäistä *tiedostonimeä* "`foo-1.0-1.i386.rpm`".

Riippuvuusvirhe voi tulla vastaan myös pakettia poistettaessa, jos jonkin muun paketin toiminta riippuu poistettavasta paketista. Ilmoitus on seuraavan tapainen:

```
$ rpm -e foo
removing these packages would break dependencies:
        foo is needed by bar-1.0-1
```

Virhe voidaan ohittaa ja paketti poistaa käyttämällä komentorivillä parametriä `--nodeps`, vaikka sen poisto tekeekin toisen paketin toimimattomaksi.

52.3.3 Päivitys

Paketin päivittäminen käy lähes samoin kuin asentaminen.

```
$ rpm -Uvh foo-2.0-1.i386.rpm
foo                         ##################################
```

Mitä yllä ei näkynyt, oli, että RPM automaattisesti poisti jo mahdollisesti asennetun vanhan version paketista `foo`. Itse asiassa on järkevää käyttää aina parametriä `-U` pakettien asentamiseen, koska se toimii, vaikka paketista ei olisikaan vanhempia versioita asennettuna.

Koska RPM suorittaa älykkään pakettien konfigurointitiedostojen päivittämisen, voit ehkä nähdä seuraavan tapaisen ilmoituksen:

```
saving /etc/foo.conf as /etc/foo.conf.rpmsave
```

Tämä tarkoittaa, että konfigurointiedostoon tehdyt muutokset eivät ehkä ole siirrettävissä uudemman ohjelmaversion konfigurointitiedostoon, joten RPM säästi vanhan version ja asensi uuden tilalle. Käyttäjän pitää tarkistaa muutokset ohjelman konfiguroinnissa ja ratkaista, miten asetukset siirretään vanhasta versiosta uuteen, jotta ohjelma voi toimia halutulla tavalla.

Koska päivittäminen on yhdistelmä poistosta ja asentamisesta, voi molemmista vaiheista tulla virheilmoituksia sekä yksi vain päivitykseen liittyvä virheilmoitus. Jos RPM huomaa, että yritetään päivittää *vanhempaan* versioon, se ilmoittaa:

```
$ rpm -Uvh foo-1.0-1.i386.rpm
foo    package foo-2.0-1 (which is newer) is already installed
error: foo-1.0-1.i386.rpm cannot be installed
```

Jotta RPM ”päivittäisi” kuitenkin, voidaan käyttää parametriä `--force` komentorivillä.

```
# rpm -Uvh --force foo-1.0-1.i386.rpm
foo                         ########################################
#
```

52.3.4 Virkistys

Paketin virkistäminen muistuttaa päivittämistä:

```
# rpm -Fvh  foo-1.2-1.i386.rpm
foo                         ########################################
#
```

RPM:n virkistystoiminto vertaa parametrina olevaa pakettia jo asennettuun versioon. Jos parametrina oleva paketti on uudempi kuin aiemmin asennettu, tapahtuu päivitys. Jos järjestelmässä ei ole aiemmin asennettua versiota ohjelmasta, *ei* parametrina olevaa pakettia asenneta. Tässä on selkeä ero päivitystoimintoon. Päivitys *asentaa* paketin, vaikka vanhaa versiota ei löytyisikään järjestelmästä.

Virkistystoimintoa voidaan käyttää sekä yksittäisten että useiden pakettien yhteydessä. Erityisen kätevä se on, kun olet hakenut useita paketteja ja haluat päivittää vain ne paketit, joista on jo aiemmin asennetut versiot. Virkistystä käytettäessä sinun ei tarvitse pelätä, että outoja ja uusia ohjelmia asentuu vahingossa.

Tässä tapauksessa kirjoitat vain komennon:

```
# rpm -Fvh  *.rpm
```

RPM päivittää vain aiemmin asennetut paketit.

52.3.5 Kysely

Kysely asennettujen pakettien tietokannasta tehdään komennolla `rpm -q`. Yksinkertainen esimerkki on `rpm -q foo`, joka tulostaa paketin nimen, version ja levitysversion asennetusta paketista foo:

```
$ rpm -q foo
rpm-2.0-1
```

Paketin nimen sijasta voidaan käyttää seuraavia optioita option `-q` yhteydessä kertomaan, mitä paketteja halutaan kysellä. Näitä kutsutaan *pakettien valintaparametreiksi*.

- `-a` kyselee kaikkia asennettuja paketteja. Voit käyttää myös jokerimerkkejä kyselyssä, mutta ne on suojattava takakenoviivalla. Jos haluat esimerkiksi kysellä kaikkia foo-alkuisia paketteja, se onnistuu komennolla `rpm -qa foo\*`.
- `-f <tiedosto>` kysyy pakettia, jolle kuuluu <tiedosto>.
- `-p <pakettitiedosto>` tekee kyselyn pakettista <pakettitiedosto>.

Kysellyistä paketeista voidaan tulostaa useita tietoja. Seuraavia optioita voidaan käyttää määrittämään tulostettavat tiedot. Niitä kutsutaan *tietojen valintaparametreiksi*.

- `-i` näyttää paketista nimen, kuvauksen, levitysversion, koon, luontipäivän, asennuspäivän, tekijän ja muuta sekalaista tietoa.
- `-l` näyttää listan paketin ”omistamista” tiedostoista.
- `-s` näyttää paketin tiedostojen tilan.
- `-d` näyttää listan tiedostoista, jotka on merkitty dokumentaatioksi (man-sivut, info-sivut, README-tiedostot jne).
- `-c` näyttää listan tiedostoista, jotka on merkitty konfiguraatiotiedostoiksi. Näitä tiedostoja muokkaamalla mukautat paketin omaan järjestelmääsi (sendmail.cf, passwd, inittab jne).

Niihin optioihin, jotka listaavat tiedostoja, voi liittää option `-v`, joka tulostaa samassa formaatissa kuin `ls -l`.

52.3.6 Tarkistus

Paketin tarkistaminen vertaa tietoja asennetuista tiedostoista alkuperäisestä paketista löytyviin tietoihin. Tarkistus vertaa mm. jokaisen tiedoston kokoa, MD5-summaa, oikeuksia, tyyppiä, omistajaa ja ryhmää.

`rpm -V` tarkistaa paketin. Voit käyttää mitä tahansa *paketin valintaparametreistä*, jotka käsiteltiin kyselyjen yhteydessä, pakettien valintaan. Yksinkertainen esimerkki on `rpm -V foo`, joka tarkistaa, että kaikki tiedostot paketissa foo ovat kuten ne olivat alkuperäisen asennuksen jälkeen. Esimerkiksi:

- Tietyn tiedoston omistavan paketin tarkistus:

```
rpm -Vf /bin/vi
```

- Kaikkien asennettujen pakettien tarkistus:

```
rpm -Va
```

- Asennetun paketin tarkistusvertailu RPM-pakettiin:

```
rpm -Vp foo-1.0-1.i386.rpm
```

  Tämä voi olla hyödyllistä, jos epäilet RPM-tietokannan olevan vioittunut.

Jos tarkistus sujuu virheittä, ei ilmoituksia tule. Tulostuksessa on kahdeksan merkkiä, mahdollisesti “c”, joka tarkoittaa kyseessä olevan konfigurointitiedoston, ja viimeeksi tiedostonimi. Kukin kahdeksasta merkistä vastaa vertailutulosta tiedoston ja RPM-tietokannassa olevan merkinnän välillä. Yksinäinen “.” (piste) tarkoittaa, että testi läpäistiin. Seuraavat merkit tarkoittavat jonkin testin epäonnistumista:

5 MD5-tarkistussumma

S Tiedoston koko

L Symbolinen linkki

T Tiedoston muutosajankohta

D Laite

U Käyttäjä

G Ryhmä

M Moodi (sisältää oikeudet ja tiedoston tyypin)

? Lukukelvoton tiedosto

Jos ilmoituksia virheistä tulee, on vaihtoehtoina paketin poisto, uudelleenasennus tai muu korjaustoimenpide.

52.4 Käytännön esimerkkejä RPM:n käytöstä

RPM on käyttökelpoinen työkalu sekä järjestelmän hallintaan että ongelmien etsintään ja korjaamiseen. Paras tapa tutustua parametrien käyttöön on tarkastella esimerkkejä.

- Oletetaan, että olet poistanut joitakin tiedostoja vahingossa, mutta et ole varma, mitä poistit. Jos haluat tarkistaa koko järjestelmän ja nähdä, mitä tiedostoja puuttuu, olisi sopiva käsky:

```
rpm -Va
```

  Jos tiedostoja puuttuu tai ne ovat vahingoittuneet, on parasta joko asentaa uudelleen kyseiset paketit tai poistaa ja asentaa uudelleen ne.

- Oletetaan, että tapaat tiedoston, jota et tunnista. Saat selville, mistä paketista se on peräisin komennolla:

```
rpm -qf /usr/X11R6/bin/xjewel
```

  Tuloste olisi:

```
xjewel-1.6-1
```

- Voimme yhdistää edelliset esimerkit seuraavaan tapaukseen. Oletetaan, että sinulla on ongelmia ohjelman `/usr/bin/paste` käytössä. Haluat tarkistaa paketin, jonka osa se on, mutta et tiedä, minkä paketin osa se on. Sopiva komento on:

```
rpm -Vf /usr/bin/paste
```

  ja kyseinen paketti tarkistetaan.

- Jos käytät ohjelmaa ja haluat tietää siitä lisää, voit kirjoittaa seuraavan komennon, joka kertoo, mitä dokumentointia ohjelman paketissa tuli mukana:

```
rpm -qdf /usr/bin/ispell
```

  Tulostus olisi:

```
/usr/man/man4/ispell.4
/usr/man/man4/english.4
/usr/man/man1/unsq.1
/usr/man/man1/tryaffix.1
/usr/man/man1/sq.1
/usr/man/man1/munchlist.1
/usr/man/man1/ispell.1
/usr/man/man1/findaffix.1
```

```
/usr/man/man1/buildhash.1
/usr/info/ispell.info.gz
/usr/doc/ispell-3.1.18-1/README
```

- Löydät uuden koules-RPM:n, mutta et tiedä, mitä se sisältää. Saat lisää tietoa kirjoittamalla:

```
rpm -qip koules-1.2-2.i386.rpm
```

Tulostus olisi:

```
Name        : koules Distribution: Red Hat Linux Colgate
Version     : 1.2          Vendor: Red Hat Software
Release     : 2        Build Date: Mon Sep 02 11:59:12 1996
Install date: (none)   Build Host: porky.redhat.com
Group       : Games    Source RPM: koules-1.2-2.src.rpm
Size        : 614939
Summary     : SVGAlib action game; multiplayer, network
Description :
This arcade-style game is novel in conception and
excellent in execution.  No shooting, no blood, no guts,
no gore.  The play is simple, but you still must develop
skill to play.  This version uses SVGAlib to run on a
graphics console.
```

- Seuraavaksi haluat tietää, mitä tiedostoja paketista asentuisi. Käsky olisi:

```
rpm -qlp koules-1.2-2.i386.rpm
```

Tulostus:

```
/usr/man/man6/koules.6
/usr/lib/games/kouleslib/start.raw
/usr/lib/games/kouleslib/end.raw
/usr/lib/games/kouleslib/destroy2.raw
/usr/lib/games/kouleslib/destroy1.raw
/usr/lib/games/kouleslib/creator2.raw
/usr/lib/games/kouleslib/creator1.raw
/usr/lib/games/kouleslib/colize.raw
/usr/lib/games/kouleslib
/usr/games/koules
```

Tässä oli tavanomaisimpia esimerkkejä. Aikaa myöden löydät RPM:lle useita muita käyttötapoja.

52.5 Muita tietolähteitä RPM:stä

Saadaksesi lisää tietoja lue man-sivu, ohjelman ohjetuloste (`rpm --help`) ja RPM-dokumentit osoitteessa

`http://www.rpm.org/`

RPM:stä on kirjoitettu myös kirja nimeltä Maximum RPM, ja se on saatavana Red Hat Softwaren kautta ja hyvin varustetuista kirjakaupoista. Kirjassa on runsaasti tietoa RPM:n käytöstä sekä loppukäyttäjälle että RPM-pakettien kokoajille. Kirja on myös luettavissa osoitteen `http://www.rpm.org/` kautta.

Samasta paikasta löytyy linkki Fedora RPM Guideen, joka on tuoreempi ja laajempi kirja kuin edellinen.

Näiden lisäksi on saatavilla mm. ohjelmoijille ja pakettien kokoajille sunnattua dokumentointia.

52.6 Yum

`yum` on yksinkertaistaen riippuvuudet huomioon ottava versio rpm:stä. Itse asiassa se käyttää apunaan rpm:ää tai sen osia eli se on laajennos tai "kuori" rpm:lle. Sillä voidaan listata, asentaa, poistaa ja päivittää paketteja. Lisäksi sen avulla voidaan päivittää koko järjestelmä uudempaan levitysversioon.

Pelkkä komento `yum` kertoo tärkeimmät ohjeet:

```
# yum
Loading "fastestmirror" plugin
You need to give some command
Usage: yum [options] < grouplist, localinstall, groupinfo, localupdate, resolvedep, erase,
deplist, groupremove, makecache, upgrade, provides, shell, install, whatprovides,
groupinstall, update, repolist, groupupdate, info, search, check-update, list,
remove, clean, grouperase >

Options:
  -h, --help            show this help message and exit
  -t, --tolerant        be tolerant of errors
  -C                    run entirely from cache, don't update cache
  -c  [config file]     config file location
```

```
  -R  [minutes]         maximum command wait time
  -d  [debug level]     debugging output level
  -e  [error level]     error output level
  -q, --quiet           quiet operation
  -v, --verbose         verbose operation
  -y                    answer yes for all questions
  --version             show Yum version and exit
  --installroot=[path]  set install root
  --enablerepo=[repo]   enable one or more repositories (wildcards allowed)
  --disablerepo=[repo]  disable one or more repositories (wildcards allowed)
  -x [package], --exclude=[package]
                        exclude package(s) by name or glob
  --obsoletes           enable obsoletes processing during updates
  --noplugins           disable Yum plugins
  --nogpgcheck          disable gpg signature checking
  --disableplugin=[plugin]
                        disable plugins by name
```

Kuten huomaat, `yum` on varsin monipuolinen.

52.6.1 List-komento

Tutustutaan ensin `list`-komentoon.

```
# yum list  kernel
Loading "fastestmirror" plugin
Loading mirror speeds from cached hostfile
Installed Packages
kernel.x86_64                           2.6.23-0.164.rc5.fc8   installed
kernel.x86_64                           2.6.23.1-31.fc8        installed
Available Packages
kernel.x86_64                           2.6.23.1-49.fc8        updates
```

`yum` näytti asennetut ja saatavilla olevat paketit. Pelkkä `yum list` olisi tuottanut varsin pitkän listan eli kaikki asennetut ja saatavilla olevat paketit eli useita tuhansia rivejä. Sen takia rajoitimme listauksen vain kernel-paketteihin. Voimme rajoittaa edelleen tilan tai lähteen perusteella käyttämällä lisämääreitä available,installed,updates,extras jne. Installed tuottaa seuraavan listauksen:

```
# yum list installed kernel
Loading "fastestmirror" plugin
Loading mirror speeds from cached hostfile
Installed Packages
kernel.x86_64                           2.6.23-0.164.rc5.fc8   installed
kernel.x86_64                           2.6.23.1-31.fc8        installed
```

Jos haluamme listata kaikki `kernel`-alkuiset paketit, olisi komento `yum list kernel*`, jolloin esimerkiksi myös kernel-devel-paketit näkyvät.

52.6.2 Info-komento

Kuten `list`, mutta tulostaa kuvauksen paketeista (saman kuin `rpm -i`).

52.6.3 Search-komento

Search-komennolle annetaan merkkijono, jota etsitään pakettien kuvaus-, tiivistelmä- (summary), paketoija- ja nimikentistä. Tämä on kätevää, jos et tiedä paketin nimeä, mutta jonkin avainsanan, jota ei esiinny muissa yhteyksissä. Esimerkiksi digi-TV:hen liittyvä dvb on melko hyvä avainsana:

```
# yum search dvb
Loading "fastestmirror" plugin
Loading mirror speeds from cached hostfile
kplayer.x86_64 : A KDE media player based on MPlayer
dvb-apps.x86_64 : Utility, demo and test applications using the Linux DVB API
vdr.x86_64 : Video Disk Recorder
vdr-subtitles.x86_64 : DVB subtitles plugin for VDR
dvb-apps.x86_64 : Utility, demo and test applications using the Linux DVB API

klear.x86_64 : DVB TV application and harddisk-recorder for linux
libdvbpsi.x86_64 : Library for MPEG TS and DVB PSI tables decoding and generation
vdr-femon.x86_64 : DVB frontend status monitor plugin for VDR
vdr-subtitles.x86_64 : DVB subtitles plugin for VDR

libdvbpsi-devel.x86_64 : Development package for libdvbpsi
kplayer.x86_64 : A KDE media player based on MPlayer
```

52.6.4 Install-komento

Install-komennolle annetaan lista asennettavista paketeista ja `yum` tarkistaa ovatko ne jo asennettu ja onko niistä saatavilla uudempi versio. Jos pakettia ei ole asennettu tai siitä on uudempi versio, `yum` tarkistaa sen riippuvuudet ja asentaa myös riippuvuuksien vaatimat paketit.

52.6.5 Update-komento

Update-komennolle annetaan lista päivitettävistä paketeista ja `yum` tarkistaa onko niistä saatavilla uudempi versio. Jos paketista löytyy uudempi versio, `yum` tarkistaa sen riippuvuudet ja asentaa myös riippuvuuksien vaatimat paketit.

Jos `update`-komennolle ei anneta pakettilistaa, `yum` päivittää kaikki paketit, joista on saatavilla uudempi versio.

52.6.6 Remove-komento

Remove-komennolle annetaan lista poistettavista paketeista ja `yum` tarkistaa, mitkä paketit vaativat kyseisen paketin ja poistaa myös ne. Jos `remove`-komennon lisäksi käytetään `-y`-optiota, voidaan poistaminen pakottaa riippuvuuksista huolimatta, mutta tämä ei ole suositeltavaa.

52.6.7 Clean-komento

`yum clean` poistaa tiedostoja hakemistosta `/var/cache/yum/`, jonne yum tallettaa mm. kaikki rpm-paketit, joita se on hakenut. Asennettujakaan paketteja ei poisteta automaattisesti. Clean-komennolle voidaan antaa lisäparametrejä, jotka määrittelevät, mitä poistetaan. Lisäparametrit ovat: packages, headers, metadata, dbcache, all, joista packages vapauttaa yleensä eniten tilaa eikä hidasta yumin toimintaa, kuten muut lisäparametrit voivat tehdä.

52.6.8 Group-lisäkomento

Group-lisäkomentoa voidaan käyttää list-, install- ja update-komentojen yhteydessä. Ennen sen käyttöä on syytä tarkistaa järjestelmän määritellyt ryhmät, kuten alla:

```
# yum grouplist
Loading "fastestmirror" plugin
Setting up Group Process
Loading mirror speeds from cached hostfile
Fedora-8-comps.xml        100% |=========================| 1.2 MB    00:03
comps-f8.xml              100% |=========================| 1.2 MB    00:03
Installed Groups:
   Office/Productivity
   Engineering and Scientific
   Administration Tools
   Editors
   System Tools
   Fonts
   Text-based Internet
   Legacy Network Server
   Authoring and Publishing
   FTP Server
   Hardware Support
   Games and Entertainment
   XFCE
   Legacy Software Development
   Clustering
   Network Servers
   Web Development
   Graphics
   Web Server
```

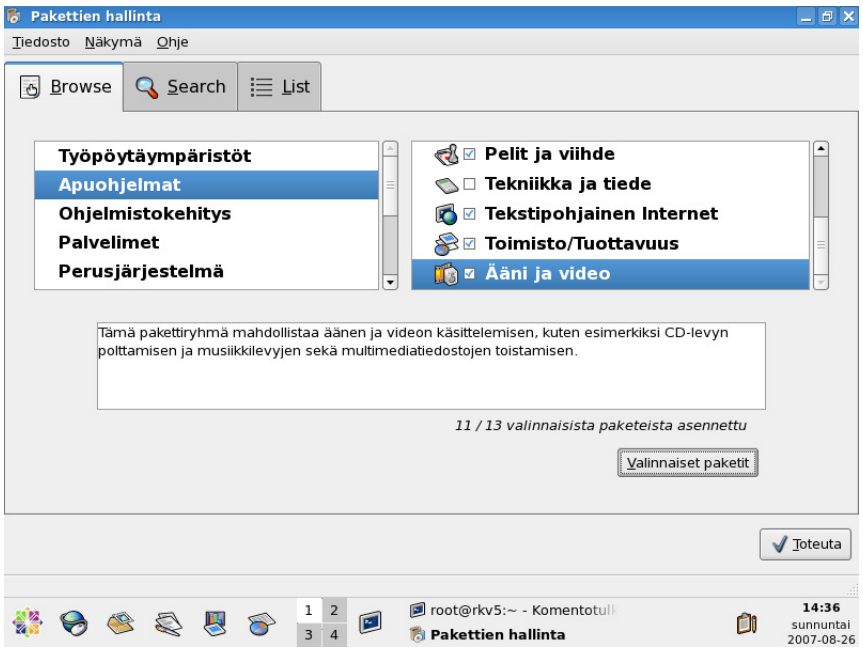
```
   Fedora Eclipse
   KDE (K Desktop Environment)
   Mail Server
   Educational Software
   Server Configuration Tools
   PostgreSQL Database
   Legacy Fonts
   Sound and Video
   Graphical Internet
Available Groups:
   MySQL Database
   Window Managers
   GNOME Software Development
   News Server
   XFCE Software Development
   X Software Development
   Virtualization
   DNS Name Server
   GNOME Desktop Environment
   Java Development
   X Window System
   Windows File Server
   Printing Support
   KDE Software Development
   Development Libraries
   Development Tools
Done
```

52.7 Muita tietolähteitä yum:n käytöstä

Edellä esiteltiin yum vain lyhyesti. Esimerkiksi yum:n pluginien käyttö ja merkitys ohitettiin täysin. Lisää tietoa tarjoavat man-sivut yum(8) ja yum.conf(5). Internetissä sivut `http://linux.duke.edu/yum/` ja `http://wiki.linux.duke.edu/YumFaq` ovat “virallisia” tietolähteitä.

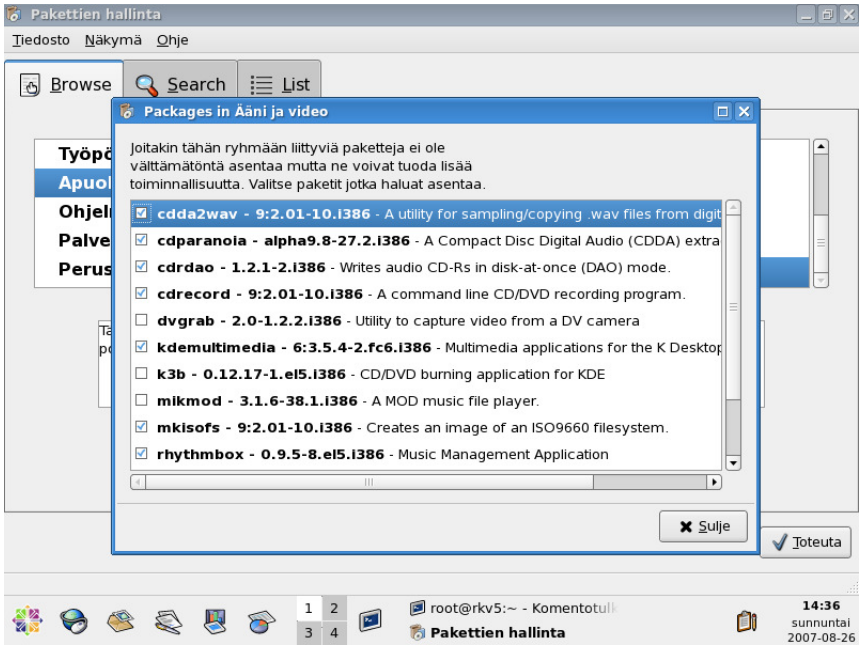
52.8 Pirut – Ohjelmien lisäys ja poisto

Pirut käynnistyy käynnistyspalkin Sovellukset-valikosta ⇒ Järjestelmä ⇒ Lisää tai poista ohjelmia tai komennoilla `pirut` ja `system-config-packages`.

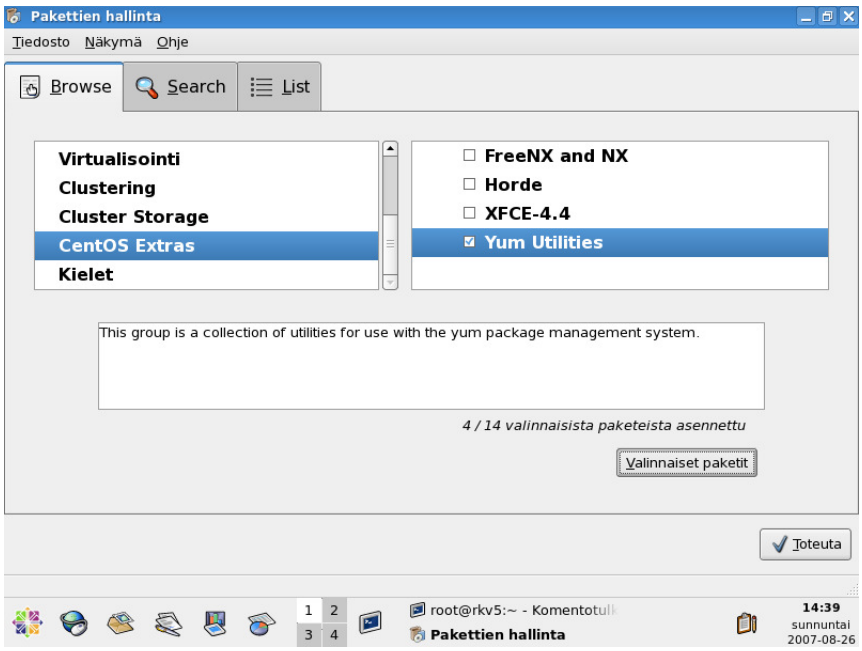


Kuva 52.1: Pakettien hallinnan oletusnäkymä

Oletusnäkymässä vasemmalla on ohjelmatyypit ja oikealla siihen kuuluvat ohjelmaryhmät. Ohjelmaryhmiin kuuluvia yksittäisiä paketteja pääsee valitsemaan
Valinnaiset paketit -painikkeella. Kussakin ohjelmaryhmässä osa ohjelmista on valittu oletuksena ja loput joutuu valitsemaan “käsin” yksitellen.



Kuva 52.2: Ohjelmaryhmän yksittäisten pakettien valinta



Kuva 52.3: Centosissa on enemmän paketteja kuin Red Hat Enterprise Linuxissa

52.9 PackageKit – Fedoran graafinen pakettien hallintatyökalu

Fedora 9:stä lähtien PackageKit on korvannut `system-config-packages`-ohjelman.

52.9.1 Taustaa

Red Hat on tehnyt itse, tukenut tekoa tai vain ottanut käyttöön useita erilaisia graafisia pakettienkäsittelyohjelmia. Niiden käyttö on ollut kuitenkin lyhytaikaista eikä niitä ole laajasti käytetty muissa levitysversioissa lukuunottamatta aivan suoria klooneja.

Komentorivillä tilanne on ollut vakaampi. Red Hat ei ole yrittänytkään tehdä mitään parempaa ohjelmaa kuin `rpm`. Red Hatille ja Red Hat -pohjaisille levitysversioille on ollut kuitenkin tarjolla kolmansien osapuolien tarjoamia pakettivarastoja ja niissä ensimmäisenä otettiin käyttöön kehittyneempiä pakettienhallintaohjelmia. Lyhyen aikaa käytettiin melko laajasti De-

bianista siirrettyä `apt-get`-ohjelmaa, joka osasi selvittää riippuvuudet automaattisesti. Sen eräänlaisena haittana oli sen tarkkuus ristiriitaisuuksien suhteen. Jos järjestelmässä oli asennettuna paketteja, joiden riippuvuudet eivät olleet tyydytetty tai riippuvuudet olivat keskenään ristiriitaisia, `apt-get` ei suostunut asentamaan uusia paketteja tai päivittämään vanhoja. Tämä tarkkuus oli toisaalta hyväkin asia, mutta tilanteen korjaaminen oli usein useille käyttäjille liian vaikeaa. Lopullinen isku oli kuitenkin ns. multi-lib-tuen viipyminen silloin, kun 64-bittiset prosessorit ja Linuxin levitysversiot olivat jo melko yleisiä. `apt-get` ei toiminut, jos 64-bittisessä Linuxissa oli myös 32-bittisiä paketteja asennettuna. `yum`:lla ei ollut kumpaakaan ongelmaa ja se onkin vakiinnuttanut asemansa jopa niin, että Red Hat itsekin tukee ja käyttää sitä. `yum`:kaan ei kuitenkaan ole välttämättä viimeinen sana...

PackageKit on varsin kunnianhimoinen hanke. Sen tekijöiden joukossa on Red Hatin työntekijöitä, mutta myös mm. Suse, Debian, Gentoo ja Mandrake ovat edustettuina. Sitä myös käytetään hyvin monissa levitysversioissa.

PackageKitin tavoitteena on yhtenäistää Linuxin levitysversioissa käytettävät graafiset pakettien hallintaohjelmat. Tekstimoodissa toimivia ohjelmia, kuten yum, apt, conary, smart jne. se ei pyri korvaamaan vaan käyttää niitä "moottorinaan". PackageKitissä on kuitenkin myös tekstimoodissa toimiva ohjelma ja silläkin on tavoitteena yhtenäistäminen.

52.9.2 PackageKitin käyttö

PackageKitin graafista käyttöliittymää ei ole tarkoitettu käynnistettäväksi komentorivilta eli se käynnistyy kuten aikaisemmatkin vastaavat työkalut käynnistyspalkin Sovellukset-valikosta ⇒ Järjestelmä ⇒ Lisää tai poista ohjelmia. Sen käyttö ei ole mitenkään yllättävää, mutta suotimien käyttö on siinä melko keskeisessä roolissa. Paketteja rastitaan sen jälkeen käsiteltäviksi eli poistettaviksi, asennettaviksi tai päivitettäviksi.

Komentorivillä tärkein ohjelma on `pkcon`, jonka toiminnallisuus vastaa `yum`ia. Lisäksi sarjaan kuuluu `pkmon`, jonka avulla voi seurata, mitä PackageKit tekee ja käyttötarkoitus on lähinnä virheiden jäljitys ja `pkgenpack`, jolla voidaan luoda paivityskokonaisuuksia (Service Pack), joissa on päivityspaketit riippuvuuksineen. Käytännössä siis on kuitenkin vain yksi tärkeä ohjelma. Se kertoo itsestään seuraavaa, jos sille ei anneta mitään parametrejä:

```
# pkcon
Käyttö:
  pkcon [VALITSIN...] PackageKit Console Program

PackageKitin konsolikäyttöliittymä

Alikomennot:
  get-actions
  get-groups
  get-filters
  get-transactions
```

```
  get-time
  search [name|details|group|file] [data]
  install [packages]
  install-local [files]
  download [directory] [packages]
  install-sig [type] [key_id] [package_id]
  remove [package]
  update <package>
  refresh
  resolve [package]
  get-updates
  get-depends [package]
  get-requires [package]
  get-details [package]
  get-distro-upgrades
  get-files [package]
  get-update-detail [package]
  get-packages
  repo-list
  repo-enable [repo_id]
  repo-disable [repo_id]
  repo-set-data [repo_id] [parameter] [value];
  what-provides [search]
  get-categories


Ohjevalitsimet:
  -h, --help              Näytä ohjevalitsimet

Sovelluksen valitsimet:
  -v, --verbose           Näytä ylimääräisiä virheenjäljitystietoja
  --version               Näytä ohjelman versio ja lopeta
  --filter                Aseta suodin, esim. asennettu
  -n, --nowait            Lopeta odottamatta toimintojen valmistumista
  -y, --noninteractive    Install the packages without asking for confirmation
  --background            Run the command using idle network bandwidth and also using less power
```

PackageKitin arkkitehtuurissa on se miellyttävä piirre, että se ei lukitse rpm-tietokantaa turhan takia. Jos olet avannut graafisen käyttöliittymän, voit silti samaan aikaan operoida tekstimoodiohjelman kanssa, jos graafinen osuus ei ole pyytänyt “moottoria” tekemään rpm-tietokantaan vaikuttavia muutoksia. Käyttäjän ohjelmat ovat siis vain käyttöliittymiä ja lukitukset tapahtuvat “moottorin” tasolla.

52.10 Kysymyksiä

1. **Asennus** Millä `rpm`-ohjelman parametrillä asennetaan paketteja?
2. **Pakotus** Millä `rpm`-ohjelman parametrillä ohitetaan riippuvuuksien tarkistus?

3. **Poisto** Millä `rpm`-ohjelman parametrillä poistetaan paketteja?
4. **Päivitys** Millä `rpm`-ohjelman parametrillä päivitetään paketteja?
5. **Kysely** Millä `rpm`-ohjelman parametreillä kysellään kaikkia asennettuja paketteja?
6. **Paketin tiedostot** Millä `rpm`-ohjelman parametrillä kysytään, mitä tiedostoja paketti sisältää?
7. **Asennus** Millä `yum`-ohjelman komennolla asennetaan paketteja?
8. **Poisto** Millä `yum`-ohjelman komennolla poistetaan paketteja?
9. **Päivitys** Millä `yum`-ohjelman komennolla päivitetään paketteja?
10. **Kysely** Millä `yum`-ohjelman komennolla ja lisämääreellä kysellään kaikkia asennettuja paketteja?

52.11 Tehtäviä

1. **KDE:n paketit** Etsi kaikki asennetut paketit, joissa on merkkijono kde (vihje: putkita `grep`-ohjelmalle). Kuuluuko KDE:hen muita paketteja?