Εισαγωγή στη διαδικασία παραγωγής πακέτων του Debian

Μανώλης Γαλάτουλας [1]
galas@tee.gr

[1] Ελληνική κοινότητα του Debian
debian-user-greek@lists.debian.org

Debian packaging workshop
3ο Συνέδριο ΕΛ/ΛΑΚ
Αθήνα, ΕΜΠ, 20 Ιουνίου 2009

Σε λίγες γραμμές

Ξεκινώντας με το “πακετάρισμα”

- Έχουμε υπόψιν ότι είναι καλό να τρέξουμε όλες τις σχετικές ρουτίνες σε ένα περιβάλλον *chroot*! Για το λόγο αυτό χρησιμοποιούμε πχ. τα **pbuilder** ή **sbuild**
- Βεβαιωνόμαστε ότι **ΔΕΝ** δουλεύει κάποιος άλλος ήδη στο πακέτο! (orphaned packages) ελέγχουμε στο **http://www.debian.org/devel/wnpp/being_packaged**
- Δοκιμάζουμε αρχικά τη *μεταγλώττιση-compilation* του προγράμματος **χωρίς να το εγκαταστήσουμε**

Απαραίτητα πακέτα:

- **dpkg-dev, file, gcc, g++, libc6-dev, make, patch, perl, autoconf, automake, dh-make, debhelper, devscripts, fakeroot, gnupg, g77, gpc, xutils**
- **lintian, pbuilder, sbuild**
- **debian-policy, developers-reference**

Ιδιαίτερα πρέπει να αναφέρουμε τα:

lintian είναι το “ελεγκτικό” μέσο που σας επιτρέπει να εντοπίσετε τα πιο συνηθισμένα σφάλματα μετά την παραγωγή του πακέτου και παρέχει την εξήγησή τους.

pbuilder (δηλ. personal builder) περιέχει προγράμματα που χρησιμοποιούνται για τη δημιουργία ενός περιβάλλοντος chroot. Η παραγωγή ενός πακέτου σε ένα τέτοιο περιβάλλον επαληθεύει τις κανονικές εξαρτήσεις του και βοηθά στην αποφυγή σφαλμάτων FTBFS.

dh-make είναι απαραίτητο για τη δημιουργία του σκελετό του παραδειγματικού πακέτου μας χρησιμοποιώντας μερικά από τα εργαλεία του **debhelper** για τη δημιουργία πακέτων. Αν και δεν είναι ουσιώδη για τη δημιουργία πακέτων συνιστώνται ιδιαίτερα για νέους συντηρητές.

Πώς να επιλέξετε πακέτο

Συμβουλευτείτε πρώτα από όλα το: **http://www.debian.org/devel/wnpp/**

- ελέγξτε ότι δεν υπάρχει κάποιος που να ασχολείται ήδη με το πακέτο. Αν υπάρχει επικοινωνήστε μαζί του(ς) αν πιστεύετε ότι πρέπει. Αν όχι βρείτε κάποιο άλλο ενδιαφέρον πακέτο.
- το πρόγραμμα πρέπει να έχει άδεια. Αν δεν είσαστε βέβαιοι για την ενότητα πακέτων στην οποία ταιριάζει το πακέτο, στείλτε το κείμενο της άδειας στη λίστα debian-legal@lists.debian.org και ζητήστε τη συμβουλή τους.

συγκεκριμένα για τις ενότητες πακέτων του Debian η άδεια πρέπει:

- Για την **κύρια-main** να είναι συμβατή με όλες τις απαιτήσεις του Debian Free Software Guidelines (**DFSG**) καθώς και να **μην** απαιτεί κάποιο άλλο πακέτο εκτός της main για τη μεταγλώττιση ή την εκτέλεσή του (όπως απαιτεί η Πολιτική του Debian).
- Για την **contrib** να είναι συμβατή με την **DSFG**, μπορεί όμως να απαιτεί πακέτα εκτός της main για την μεταγλώττιση ή την εκτέλεσή του.
- Για την **non-free**, μπορεί να μην είναι συμβατό με κάποιες από τις απαιτήσεις του DSFG αλλά θα πρέπει να είναι *distributable*.

Σημεία που πρέπει να προσεχθούν:

- το πρόγραμμά σας **δεν** θα πρέπει να εκτελείται ως *setuid root*, ακόμα καλλίτερα να μην χρειάζεται να εκτελεστεί ως setuid ή setgid από οποιονδήποτε χρήστη/ομάδα.
- το πρόγραμμα δεν θα πρέπει να είναι ένας δαίμονας ή κάτι που εγκαθίσταται στους κατάλογους */sbin ή να ανοίγει κάποια θύρα (port) ως χρήστης root.
- το πρόγραμμα θα πρέπει να δίνει ως αποτέλεσμα ένα εκτελέσιμο binary αρχείο. Η διαχείριση βιβλιοθηκών είναι πολύ δυσκολότερη.
- θα πρέπει να έχει επαρκή τεκμηρίωση και ο κώδικας να είναι κατανοητός.
- θα πρέπει να επικοινωνήσετε με τον συγγραφέα του προγράμματος και να ελέγξετε ότι συμφωνεί στο πακετάρισμα του προγράμματος. Μην προσπαθήσετε να πακετάρετε κομμάτια του προγράμματος που δεν συντηρούνται από τον δημιουργό τους.

Ως νέοι συντηρητές αποφεύγουμε λοιπόν:

τη δημιουργία πολύπλοκων πακέτων, όπως για παράδειγμα:

- πολλαπλά binary πακέτα
- πακέτα βιβλιοθηκών
- που ο τύπος αρχείου του πηγαίου κώδικά τους δεν είναι tar.gz ή tar.bz2
- που το συμπιεσμένο αρχείο (tarball) του πηγαίου κώδικα περιέχει μη-διανεμήσιμο (non-distributable) περιεχόμενο.

Γιατί chroot;

Συνιστάται ιδιαίτερα η διαδικασία της παραγωγής του πακέτου να γίνεται σε ένα περιβάλλον chroot για τους παρακάτω λόγους:

- Ασφάλεια
- Δυνατότητα δοκιμών
- Δεν υπάρχει κίνδυνος να καταστραφεί το σύστημα του developer.
- πακέτα που έχουν εγκατασταθεί λάθος μπορούν να εντοπιστούν εύκολα: με ένα απλό **ls -lR** στη βάση του chroot περιβάλλοντος πριν και μετά την παραγωγή του πακέτου και με **diff -u before.list after.list**.
- Ευκολότερος εντοπισμός εξαρτήσεων που λείπουν. Άρα και αποφυγή σφαλμάτων τύπου **FTBFS**, δηλ. **Fails To Build From Source**.
- Το περιβάλλον chroot είναι ένα αμιγές σύστημα Debian (πχ. Lenny).

Στο παράδειγμά μας (το βρίσκουμε στο New Maintainer's Guide) υποθέτουμε ότι θέλουμε να πακετάρουμε το πρόγραμμα *gentoo* του οποίου η τρέχουσα έκδοση είναι 0.9.12.

Εκτελούμε την εντολή **dh_make** από το πακέτο *dh-make*:
dh_make -e your.maintainer@address -f ../gentoo-0.9.12.tar.gz
ΠΡΟΣΟΧΗ! η εντολή *dh_make* πρέπει να εκτελεστεί μόνο **μια** φορά! Δεν πρόκειται να έχει σωστά αποτελέσματα αν εκτελεστεί ξανά στον ήδη debianised κατάλογο!

Ο κατάλογος debian/*

Εδώ είναι όλα τα κρίσιμα αρχεία που επιτρέπουν στο dpkg να ξέρει πώς θα παράξει το πακέτο. Πρώτα από όλα το αρχείο:

debian/control (τυπικά με περιεχόμενο όπως:)

```
Source: gentoo
Section: unknown
Priority: optional
Maintainer: Josip Rodin <joy-mg@debian.org>
Build-Depends: debhelper (>> 3.0.0)
Standards-Version: 3.6.2
Package: gentoo
Architecture: any
Depends: $shlibs:Depends
Description: <insert up to 60 chars description>
<insert long description, indented with spaces>
```

Τι σημαίνουν οι εξαρτήσεις:

Depends το πακέτο δεν εγκαθίσταται αν δεν εγκατασταθούν τα πακέτα από τα οποία εξαρτάται. Χρησιμοποιήστε αυτό το flag μόνο αν είστε απόλυτα σίγουροι ότι το πρόγραμμά σας δεν θα τρέξει (ή θα προκαλέσει σοβαρή δυσλειτουργία στο σύστημα) εκτός αν είναι παρόν ένα συγκεκριμένο πακέτο.

Recommends Προγράμματα όπως το dselect ή το aptitude θα σας παρακινήσουν να εγκαταστήσετε μαζί με το πακέτο σας και τα συνιστώμενα πακέτα. Το dselect θα επιμείνει. Το dpkg και το apt-get θα αγνοήσουν αυτό το πεδίο. Χρησιμοποιήστε το για πακέτα που δεν είναι αυστηρά απαραίτητα αλλά τυπικά εγκαθίστανται με το πρόγραμμά σας.

Suggests Χρησιμοποιήστε το για πακέτα που θα δουλέψουν αρμονικά με το πρόγραμμά σας αλλά δεν είναι απαραίτητα.

Εξαρτήσεων συνέχεια:

Pre-Depends Ισχυρότερο από το Depends. Το πακέτο δεν θα εγκατασταθεί αν δεν έχουν εγκατασταθεί και ρυθμιστεί σωστά τα πακέτα από τα οποία προ-εξαρτάται. Χρησιμοποιήστε το πολύ σπάνια και μόνο μετά από συζήτηση στη λίστα debian-devel. Με άλλα λόγια: μην το χρησιμοποιήσετε καθόλου!

Conflicts Το πακέτο δεν θα εγκατασταθεί αν δεν έχουν αφαιρεθεί όλα τα πακέτα με τα οποία “συγκρούεται”. Χρησιμοποιήστε το μόνο αν το πρόγραμμά σας δεν πρόκειται με βεβαιότητα να τρέξει ή θα δημιουργήσει πολύ σοβαρά προβλήματα δυσλειτουργίας στην παρουσία κάποιου πακέτου..

Provides Αφορά κάποιους τύπους πακέτων για τα οποία υπάρχουν διάφορες εναλλακτικές. Μπορείτε να δείτε την πλήρη λίστα τέτοιων πακέτων στο /usr/share/doc/debian-policy/virtual-package-names-list.txt.gz. Χρησιμοποιήστε το αν το πρόγραμμά σας παρέχει μια λειτουργία ενός υπάρχοντος “εικονικού” πακέτου

Replaces Χρησιμοποιήστε το αν το πρόγραμμά σας αντικαθιστά αρχεία από κάποιο άλλο πακέτο ή αντικαθιστά πλήρως κάποιο άλλο πακέτο (σε συνδυασμό με το Conflicts). Αρχεία από το αναφερόμενα πακέτα θα αντικατασταθούν με τα αρχεία από το δικό σας πακέτο.

debian/copyright :

πληροφορίες σχετικά με upstream πηγές, πνευματικά δικαιώματα και άδειες χρήσης

debian/changelog :

απαραίτητο αρχείο! Η πληροφορία που περιέχει χρησιμοποιείται από το dpkg και άλλα προγράμματα για τον προσδιορισμό της έκδοσης, της αναθεώρησης, της διανομής και του επείγοντος του πακέτου.

το αρχείο debian/rules

πολύ σημαντικό αρχείο αφού περιέχει τους κανόνες τους οποίους θα χρησιμοποιήσει το *dpkg-buildpackage* για την πραγματική παραγωγή του πακέτου!

Είναι στην ουσία ένα άλλο είδος Makefile διαφορετικού όμως από αυτό στην upstream έκδοση του προγράμματος. Απόσπασμα από ένα τέτοιο αρχείο πχ. στο *http://debian.org/doc/maint-guide/ch-dreq.en.html#s-rules*

άλλα αρχεία κάτω από το debian/*

- README.Debian: άλλες λεπτομέρειες ή διαφορές μεταξύ του αρχικού πακέτου και της debianised έκδοσης!
- conffiles.ex: η λύση του Debian στη διατήρηση των υπαρχόντων αρχείων ρυθμίσεων!
- cron.d.ex: αν χρησιμοποιείται το cron.d
- dirs: κατάλογοι που χρειάζονται αλλά για κάποιο λόγο η συνηθισμένη διαδικασία εγκατάστασης δεν δημιουργεί
- docs: τα ονόματα των αρχείων τεκμηρίωσης
- emacsen-*.ex: αν το πρόγραμμά σας παρέχει αρχεία Emacs

- init.d.ex: για δαίμονες που ξεκινούν με την εκκίνηση του συστήματος
- manpage.1.ex, manpage.sgml.ex: manual pages
- menu.ex: μενού για προγράμματα του X Window system
- watch.ex: ρύθμιση των προγραμμάτων uscan και uupdate (από το πακέτο devscripts)
- ex.package.doc-base
- postinst.ex, preinst.ex, postrm.ex, prerm.ex: maintainer's scripts

Τροποποίηση πηγαίου κώδικα

Συχνά upstream πακέτα εγκαθιστούν αρχεία κάτω από τον κατάλογο */usr/local*. **ΜΗΝ ΕΓΚΑΤΑΣΤΗΣΕΤΕ ΟΠΟΙΑΔΗΠΟΤΕ ΑΡΧΕΙΑ ΕΚΕΙ!!!**.
Αυτό συνεπάγεται αλλαγές στο original αρχείο Makefile όπως:

```
install: gentoo
install ./gentoo $(BIN)
install icons/* $(ICONS)
install gentoorc-example $(HOME)/.gentoorc
```

Μετά τις αλλαγές μας θα πρέπει να μοιάζει έτσι:

```
install: gentoo-target
install -d $(BIN) $(ICONS) $(DESTDIR)/etc
install ./gentoo $(BIN)
install -m644 icons/* $(ICONS)
install -m644 gentoorc-example $(DESTDIR)/etc/gentoorc
```

για complete rebuild

τρέχουμε **dpkg-buildpackage -rfakeroot** που κάνει τα εξής:

- καθαρισμός του δέντρου με τον πηγαίο κώδικα (debian/rules clean), χρησιμοποιώντας το fakeroot
- παραγωγή του πακέτου πηγαίου κώδικα (**dpkg-source -b**)
- παραγωγή του προγράμματος (**debian/rules build**)
- παραγωγή του binary πακέτου (**debian/rules binary**), με το fakeroot
- υπογραφή του αρχείου πηγαίου κώδικα .dsc, χρησιμοποιώντας το gnupg
- δημιουργία και υπογραφή του αρχείου .changes, χρησιμοποιώντας τα dpkg-genchanges και gnupg

Έλεγχοι και upload

έλεγχος μεταγλώττισης **licensecheck -r ***

έλεγχος με το lintian **lintian package-version.changes**

piuparts piuparts binpackage-version.deb

Uploading στην αρχειοθήκη του Debian :

dupload gentoo_0.9.12-1_i386.changes

Για καινούρια upstream release (βασική εκδοχή) :

uupdate -u gentoo-0.9.13.tar.gz

Σημεία ελέγχου!

- Υπάρχει μια σελίδα εγχειριδίου (man page) για κάθε εκτελέσιμο αρχείο στο πακέτο; Σε μερικές σπάνιες περιπτώσεις έχει νόημα να υπάρχει μι μοναδική σελίδα εγχειριδίου για πολλαπλά binaries σε ένα πακέτο.
- Έχει κάποιο αντίστοιχο στο Debian menu; (πχ. προγράμματα που είναι “δαίμονες” δεν χρειάζονται συνήθως κάτι τέτοιο). Έχουν κάποιο αρχείο .desktop;
- Αν υπάρχουν στοιχεία στο πακέτο που είναι ανεξάρτητα αρχιτεκτονικής και αν είναι αρκετά μεγάλα είναι πακεταρισμένα σε ένα ξεχωριστό πακέτο με αρχιτεκτονική **all**;
- Μεταγλωττίζεται το πακέτο χωρίς προβλήματα; Αυτό δεν σημαίνει ότι μπορείτε να ανακατευθύνετε τα σφάλματα στην έξοδο **/dev/null**!
- Γκρινιάζει το lintian όταν ελέγχει τα αρχεία .changes;
- Τι λέει το piuparts για τα αρχεία .debs;

Πιο προχωρημένα θέματα:

περαιτέρω αυτοματοποίηση με την εντολή **debuild**

dpatch και quilt διαχείριση patches για πακέτα Debian (το quilt θεωρείται ανώτερο ιδιαίτερα για πολύπλοκα πακέτα)

quick rebuild για μεγάλα πακέτα επιτρέπει την παραγωγή του .deb αρχείου χωρίς αναπαραγωγή των upstream πηγών με την εντολή
fakeroot debian/rules binary. Μετά το fine tuning θα πρέπει όμως να επαναληφθεί η διαδικασία για την πλήρη αναπαραγωγή του πακέτου.

pbuilder για την επαλήθευση των εξαρτήσεων build σε ένα “καθαρό” περιβάλλον chroot. Δείτε το http://buildd.debian.org/ για περισσότερα σχετικά με τον Debian package auto-builder.

piuparts είναι μια σουίτα προγραμμάτων που ελέγχει το αν γίνεται σωστά η εγκατάσταση, αναβάθμιση ή η αφαίρεση πακέτων Debian δημιουργώντας επίσης ένα minimal Debian σύστημα σε περιβάλλον chroot.

προχωρημένα θέματα συνέχεια:

καινούρια upstream release (realistic) :

- Επαληθεύστε τις αλλαγές στον πηγαίο κώδικα upstream
- Μετάπτωση του παλιού Debian πακέτου στην καινούρια έκδοση.
- Παραγωγή του πακέτου όπως περιγράφεται με τα debuild ή pbuilder.
- οι διάφοροι έλεγχοι (για ανοιχτά bugs, το αρχείο .changes κλπ.)

Ακρωνύμια:

για διεξοδική λίστα πχ. στο
http://www.infodrom.org/Debian/doc/acronyms.html

WNPP: Work-needing and Prospective Packages

FTBFS: Failed To Build From Source

ITP: Intend To Package

NMU: Non-Maintainer Upload

QA: Quality Assurance

RFP: Request For Packaging

BTS, PTS: Bug Tracking System, Package Tracking System

Απλό παράδειγμα:

$ mkdir -p /path/to (δημιουργία νέου κενού καταλόγου)
$ cd /path/to
$ tar -xvzf /path/from/gentoo-1.0.2.tar.gz (έχουμε πάρει τον πηγαίο κώδικα έτσι!)
$ cd gentoo-1.0.2
$ dh_make -e name@domain.dom -f /path/from/gentoo-1.0.2.tar.gz
... Απάντηση των προτρεπτικών ερωτήσεων.
... Διορθώσεις στο δέντρο του πηγαίου κώδικα
... Αν πρόκειται για ένα πακέτο σεναρίου (script), ορίστε το πεδίο Architecture στο debian/control σε ”Architecture: all”
... ΜΗΝ διαγράψετε το αρχείο ../gentoo_1.0.2.orig.tar.gz
$ debuild
... Βεβαιωθείτε ότι δεν προκύπτουν οποιεσδήποτε προειδοποιήσεις.
$ cd ..
$ dupload -t nm_target gentoo_1.0.2-1_i386.changes

Τα απολύτως προαπαιτούμενα

New Maintainer's Guide: *http://www.debian.org/doc/maint-guide/*

Debian Policy: *http://www.debian.org/doc/debian-policy/*

Απαραίτητοι Δικτυακοί τόποι

WNPP	http://www.debian.org/devel/wnpp/being_packaged
Μέντορες	http://mentors.debian.net
Debian Wiki	http://wiki.debian.org/HowToPackageForDebian

Τεκμηρίωση, άλλες παρουσιάσεις

- http://www.slideshare.net/miahfost/the-gory-details-of-debian-packages-pres
  entation
- http://www.slideshare.net/gnunify/how-to-build-debian-packages
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Deb ian-Bi
  nary-Package-Building-HOWTO.html
- https://wiki.ubuntu.com/MOTU/Packages/Packaging/Tips
- http://qref.sourceforge.net/Debian/reference/ch-package.en.html
- http://www.debian.org/doc/manuals/apt-howto/ and package apt-howto