Краткое руководство по созданию установочных пакетов ОС Ubuntu Linux и их интеграции в центральные репозитории Ubuntu

Голосов И.С., Ефремов И.Е., Фурсов М.Ю. (УНИПРО)

Назначение документа

Документ предназначен для специалистов, создающих установочные пакеты Ubuntu на основе оригинальных программных пакетов. Руководство содержит инструкцию по созданию установочных пакетов; в нем описываются требуемые для этого служебные файлы и скрипты. Также, в документе приведены инструкции по интеграции созданных пакетов в центральные репозитории проекта Ubuntu и их обновлению.

Содержание

Раздел 1. Начало работы

1.1 Подготовка рабочего окружения

Для начала работы интегратору потребуется:

1. Установить последнюю стабильную версию ОС Ubuntu Linux[1]
2. Установить пакеты:
   - build-essential
   - devscripts
   - ubuntu-dev-tools
   - debhelper
   - diff and patch
   - fakeroot
   - lintian
   - pbuilder

Пакет orig.tar.gz

Наиболее важной частью процесса создания установочного пакета ПО для Ubuntu является правильно структурированный пакет исходных кодов. В качестве примера рассмотрим пакет **GNU hello** [2].

Пакет исходных кодов должен быть представлен в виде архива: **hello-2.6.tar.gz.** Содержимое архива распаковывается в директорию **hello-2.6**.

Первый шаг к созданию установочного пакета – это переименование архива в соответствии со специальным форматом – после переименования архив должен именоваться **hello_2.6.orig.tar.gz.** По символу подчеркивания и ключевому слову **orig** система сборки выделяет предназначенные для нее пакеты.

Важным аспектом является то, что интеграция какого-либо ПО в центральные репозитории требует отсутствия в пакете файлов, не используемых для построения или работы. Например, ни один пакет в Ubuntu не может прямо включать в себя какую-либо библиотеку, в случае, если эта библиотека уже входит в репозитории Ubuntu. В этом случае, библиотека должна быть удалена из архива и в служебные файлы должна быть добавлена ссылка на требуемую зависимость. В таком случае, **архив orig.tar.gz** отличается по составу от исходного, и в его имя должен быть добавлен инфикс **«+repack»,** например, **hello_2.6+repack.orig.tar.gz**.

Ссылки к разделу 1:

[1] http://ubuntu.com

[2] http://www.gnu.org/software/hello/

Раздел 2. Создание служебных файлов

Основа для работы системы сборки установочного пакета – набор служебных файлов, которые содержат описание будущего пакета, инструкции и скрипты для его создания. Все служебные файлы для создания установочного пакета хранятся в директории **debian**, расположенной в корне распакованного архива исходных кодов.

2.1. Служебный файл control

Пример заполнения файла **control** (файл, описывающий структуру, зависимости и содержимое пакета):

Source: hello
Section: devel
Priority: optional
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
XSBC-Original-Maintainer: Captain Packager <packager@coolness.com>
Standards-Version: 3.7.3
Build-Depends: debhelper (>= 5)
Homepage: http://www.gnu.org/software/hello/

Package: hello
Architecture: any
Depends: ${shlibs:Depends}
Description: The classic greeting, and a good example
The GNU hello program produces a familiar, friendly greeting. It allows non-programmers to use a classic computer science tool which would otherwise be unavailable to them. Seriously, though: this is an example of how to do a Debian package. It is the Debian version of the GNU Project's `hello world' program (which is itself an example for the GNU Project).

Рассмотрим необходимые для заполнения поля:

1. Поле Source: после двоеточия необходимо указать имя пакета, в данном случае – **hello.**
2. Поле **Section:** определяет категорию ПО: инженерное, служебное, научное. Список возможных секций содержится по ссылке [1].
3. Поле **Priority**: для подавляющего большинства пользовательских пакетов должно быть **Optional.** Возможные значения поля: **[Important|Standard|Optional|Extra]. Important и Standard** определяют обязательные пакеты, из которых состоит

поставка операционной системы, **Extra** – проблемные пакеты, конфликтующие с какими-либо другими пакетами.

4. Поле **Maintainer**: должно содержать фиксированное значение Ubuntu Developers **<ubuntu-devel-discuss@lists.ubuntu.com>.**
5. Поле **XSBC-Original-Maintainer**: имя и электронная почта создателя установочного пакета.
6. Поле **Standards-Version**: версия стандарта, по которому создан пакет. Выводится командой **apt-cache show debian-policy | grep Version** .
7. Поле **Build-Depends**: перечисление пакетов, от которых зависит сборка пакета. Здесь должны быть перечислены компиляторы, библиотеки и прочие зависимости, при необходимости с версиями. Пакеты, входящие в **build-essentials** (например, компилятор **gcc)**, указывать не нужно. См. файл /**usr/share/doc/build-essential/list.**
8. Поле **Homepage:** домашняя страница проекта.

Перечисленные выше поля характеризуют, так называемый, установочный пакет исходных кодов. Из этого пакета может быть построено несколько бинарных пакетов, которые, непосредственно, устанавливаются на ПЭВМ пользователей. Каждый из бинарных пакетов должна описывать секция, по образцу следующей. В данном примере эта секция описывает единственный бинарный пакет:

1. Поле **Package: имя бинарного пакета**. В простом случае совпадает с полем Source. Наиболее яркий случай несовпадения – это пакет, содержащий библиотеку, в этом случае если имя библиотеки – **somelibrary**, то бинарный пакет будет именоваться **libsomelibrary1**, где 1 – мажорная версия библиотеки.
2. Поле **Architecture**: содержит список архитектур, на которых ПО работоспособно. Для кросс-платформенного ПО указывается значение **all** – в случае если ПО работает на всех платформах одинаково без перекомпиляции (скрипты), либо **any** – если для каждой платформы требуется отдельная сборка. Возможно указать список поддерживаемых архитектур: **i386, amd64, ppc** и так далее.
3. Поле **Depends:** должно содержать обязательные значения **${shlibs:Depends}, ${misc:Depends}** которые являются макроопределениям для значений, которые подставит система сборки. Система автоматически отслеживает зависимости построенных бинарных файлов. Если пакет зависит от других пакетов, описанные в текущем **control-файле**, они должны быть перечислены в этом поле.
4. Поле **Description**: содержит описание пакета. Первое предложение должно кратко формулировать суть пакета, оно начинается с большой буквы и не заканчивается

точкой – это так называемое «короткое описание». С новой строки, с отступом в один пробел содержится более детальное описание из пяти-семи предложений.

5. Возможны **опциональные поля**:
   a. **Recommends** – список пакетов, рекомендованных для установки вместе с текущим;
   b. **Suggests –** аналогичные текущему пакеты, которые также могут оказаться полезны;
   c. **Conflicts** – список пакетов, совместно с которыми текущий пакет не может быть установлен.
   d. **Section –** переопределяет секцию, указанную выше для всех бинарных пакетов.

Рассмотрим случаи нескольких бинарных пакетов, описанных в одном файле:

```
Package: libgala2
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: guaranteed accuracy in linear algebra
 GALA is a Fortran library for solving linear algebra problems
 with guaranteed accuracy.

 This package contains the shared library.

Package: libgala-dev
Section: libdevel
Architecture: any
Depends: libgala2 (= ${binary:Version}), ${misc:Depends}
Description: gala library mod files and static lib
 GALA is a Fortran library for solving linear algebra problems
 with guaranteed accuracy.
 .
 This package contains mod files and the static library.
```

Это типичный пример «библиотечного» пакета – первая секция описывает динамическую библиотеку, которая будет использоваться пользователями, вторая секция описывает пакет, который требуется для разработчика, использующего библиотеку. Второй пакет содержит статическую версию ПО, а также файлы необходимые для

разработки – для Фортрана-90 это *.mod-файлы, для Си и Си++ - заголовочные *.h и *.hpp-файлы. Необходимо обратить внимание на именование библиотечных пакетов (libgala2 – префикс «lib» и мажорная версия «2», libgala-dev: префикс lib и суффикс, показывающий что пакет предназначен для разработчиков). Также важно выставление зависимости пакета для разработчиков от пакета для пользователей – при установке пакета для разработчиков динамическая версия библиотеки будет установлена по зависимости.

Другой случай нескольких пакетов – создание основного бинарного пакета, и пакета со статичесскими данными:

Package: ugene
Architecture: any
Depends: ${shlibs:Depends}, libqt4-gui (>=4.5.0), libqt4-core (>=4.5.0), ugene-data
Description: integrated bioinformatics toolkit
Unipro UGENE is a cross-platform visual environment for DNA and protein sequence analysis. UGENE integrates the most important bioinformatics computational algorithms and provides an easy-to-use GUI for performing complex analysis of the genomic data. One of the main features of UGENE is a designer for custom bioinformatics workflows.

Package: ugene-data
Architecture: all
Description: required data for UGENE - integrated bioinformatics toolkit
Unipro UGENE is a cross-platform visual environment for DNA and protein sequence analysis.
.
This package contains various data and example files for UGENE.

В этом примере, второй пакет содержит статические данные, одинаковые для всех архитектур. Посколько объем данных является значительным, создатели пакета приняли решение выделить их, для того чтобы избежать их размножения в составе бинарных пакетов для разных архитектур (i386, amd64 и прочие). Основной пакет зависит от пакета с данными, таким образом, при установке приложения, данные будут установлены автоматически. Пакет данных не требует компиляции и одинаков для всех платформ, поэтому содержит значение all в секции Architecture.

2.2. Служебный файл rules

Этот файл является Make-файлом для работы системы сборки установочного пакета. Большинство современных пакетов содержат тем или иным способом сгенерированный файл **rules**, вместо написанного вручную. Одним из наиболее распространенных способов получения **rules** является адаптирование шаблонных заготовок. Шаблоны поставляются вместе с пакетом **debhelper** и располагаются в директории **/usr/share/doc/debhelper/examples**. Детальная информация об использовании **debhelper** располагается по ссылке [2]. Готовые шаблоны содержат в комментариях инструкции по адаптации и удобны для создания простых пакетов: **rules.arch** для приложений, **rules.multi** для построения двух бинарных пакетов – архитектурно-зависимого и архитектурно-независимого, **rules.multi2** – для построения нескольких архитектурно-зависимых пакетов.

При использовании **debhelper** для создания файла **rules** необходимо указать **debhelper** и его версию в поле **Build-depends** служебного файла **control.**

При создании этого файла наиболее важен факт наличия корректного **Makefile** в оригинальном пакете исходных кодов. Именно от его свойств, таких как наличие нужных операций **(install, uninstall, clean, all)** зависит удобство создания **rules-файла**.

2.3. Служебный файл copyright

Данный файл предназначен для агрегирования кратких сведений о правовой информации для всех файлов, входящих в состав пакета исходных кодов. Формат простого файла представлен ниже:

This package was debianized by {Your Name} <your email address>
{Date}
It was downloaded from: {URL of webpage}

Upstream Author(s): {Name(s) and email address(es) of author(s)}

Copyright:
Copyright (C) {Year(s)} by {Author(s)} {Email address(es)}

Licence:

{Add licence text here. For GNU licences add the licence header and a link to the appropriate file in /usr/share/common-licences.}

Packaging:
Copyright (C) {Year(s)} by {Your Name} <your email address>
released under {the licence you choose for your packaging}

В этом шаблоне необходимы для заполнения поля:

- **{Your Name}** – имя создателя установочного пакета;
- **<your email address>** – адрес электронной почты;
- **{Date}** – дата создания пакета. Формат даты должен соответствовать выводу команды date –R;
- **{URL of webpage}** – адрес веб-страницы проекта;
- **{Name(s) and email address(es) of author(s)}** – имена и адреса электронной почты авторов проекта;
- **{Year(s)} by {Author(s)} {Email address(es)}** – информация о копирайте по годам;
- **{Add licence text here…}** – заголовок лицензии (для распространенных лицензий, таких как GNU GPL или Apache License), либо текст лицензии полностью;
- **{Year(s)}...** – информация о копирайте для файлов установочного пакета;
- **{the licence you choose for your packaging}** – лицензия на файлы установочного пакета. В общем случае эта лицензия может не совпадать с лицензией на содержимое пакета.

Важно заметить, что помимо описанной информации, данный файл должен содержать лицензионную информацию о каждом файле, на который не распространяется указанная выше лицензия. Например, при переиспользовании сторонних исходных кодов, файл copyright должен содержать перечень этих файлов с указанием лицензий и правовой информации.

Помимо лицензионной информации, **файл copyright** должен содержать информацию о модификациях исходного пакета исходных кодов в процессе создания пакета ***.orig.tar.gz.**

Другое важное замечание касается обязательного указания версий лицензии: например, **GNU GPL v3 или Apache License v2**. Наиболее корректно указывать что файлы распространяются на условиях определенной версии, либо любых версий превышающих указанную (в сокращениях это отмечается знаком «+», например: GPLv3+ – лицензия версии 3, либо любая старшая версия).

2.4. Служебный файл changelog

Файл содержит список изменений, вносимых в установочный пакет в процессе его существования. Записи оформляются по образцу:

package (version) distribution; urgency=urgency

  * change details
    more change details
  * even more change details

 -- maintainer name <email address>[two spaces] date

Рассмотрим необходимые для заполнения поля:

- **package** – имя пакета;
- **version –** версия. Это поле должно содержать версию, в формате принятом в ОС Ubuntu Linux: 1.2.3-0ubuntu1, где 1.2.3 (либо 1.2) – версия самого пакета, 0ubuntu – ключевое слово, 1 – версия сборки установочного пакета. В случае если исходный пакет был модифицирован, в середину строки-версии добавляется ключевое слово +repack, в результате: 1.2.3+repack-0ubuntu1. Создание пакетов для специализированных репозиториев, таких как Ubuntu Personal Package Archive может потребовать дополнительных суффиксов, например: 1.2.3+repack-0ubuntu1~ppa1~karmic1;
- **distribution** – кодовое имя, указывающее целевую версию ОС Ubuntu Linux, например lucid, maverick, natty;
- **urgency** – для пользовательских пакетов указывается значение low;
- **{change details}** – описание внесенных изменений в установочных пакет;
- **maintainer name** – имя инженера, внесшего запись;
- **<email address>** – адрес электронной почты автора записи;
- **date** – дата, в формате команды date –R.

Пример правильно заполненного журнала изменений:

hello (2.4-0ubuntu1) jaunty; urgency=low

  * New upstream release with lots of bug fixes and feature improvements.

 -- Captain Packager <packager@coolness.com> Wed, 5 Jan 2009 22:38:49 -0700

При написании журнала изменений важно обращать внимание на строгое форматирование записей: наличие пустых строк, двухпробельный отступ у каждой

записи, двухпробельный интервал между информацией об авторе и датой внесения записи.

2.5. Прочие служебные файлы

- **compat:** файл должен содержать единственное число – мажорную версию пакета debhelper: в случае если файл rules основан на debhelper.
- **source/format**: файл должен содержать строку «3.0 (quilt)», указывающую формат установочного пакета исходных кодов принятый в репозиториях Ubuntu и Debian.
- **watch:** файл, содержащий скрипт, скачивающий пакет исходных кодов с сайта проекта. Инструкция по написанию файла watch находится по ссылке [3].
- Создание нескольких бинарных пакетов может требовать дополнительных файлов вида **binarypackage1.install, binarypackage2.install и** т. д. Эти файлы содержат списки файлов полученных в результате сборки, включаемых в пакет с соответствующим именем.

Рассмотрим пример с пакетом gala, control-файл для которого был показан выше:

Файл libgala-dev.install:

usr/lib/lib*.a

usr/lib/lib*.so

usr/share/libgala2/examples/*.f90

usr/include/*.mod

Файл libgala2.install:

usr/lib/lib*.so.?.?

usr/lib/lib*.so.?

Из примера видно, что бинарный пакет libgala2 содержит только разделяемую библиотеку и ссылку на нее, а пакет libgala-dev, предназаначенный для программистов, использующих библиотеку для разработки, содержит статическую версию, заголовочные файлы и примеры.

Ссылки к разделу 2:

[1] http://packages.ubuntu.com/natty/

[2] http://www.opennet.ru/docs/RUS/debian_pkg/node10.html

[3] https://wiki.ubuntu.com/PackagingGuide/Recipes/DebianWatch

Раздел 3. Создание установочных пакетов

Установочный пакет Ubuntu – это файл с расширением .deb, содержащий в себе бинарные файлы программного пакета и метаинформацию их описывающую. Отличают установочные пакеты исходных кодов – это пакеты, содержащие только исходный код и метаинформацию, созданные на основе оригинальных (upstream) пакетов исходных кодов.

Для того чтобы создать установочные пакеты, необходимо следующее:

1. **Создать архив orig.tar.gz.** В общем случае, этот архив может отличаться от оригинального – из него могут быть удалены части, несущественные для создания установочных пакетов. В случае модификации, в имя архива добавляется строка «+repack»: packagename_1.2.3+repack.orig.tar.gz.
2. **Распаковать этот архив**.
3. **Поместить в корень распакованной директории директорию debian**, содержащую описанные служебные файлы.
4. **Выполнить команду dpkg-buildpackage –S** для построения установочного пакета с исходными кодами. Рассмотрим результат работы команды на примере:

gala-2.0	Распакованный архив orig.tar.gz
gala_2.0-0ubuntu1.debian.tar.gz	Запакованная директория debian
gala_2.0-0ubuntu1.dsc	Метаинформация о пакете
gala_2.0-0ubuntu1_source.changes	Метаинформация о пакете
gala_2.0.orig.tar.gz	Оригинальный архив

5. **Выполнить команду dpkg-buildpackage** –B для построения бинарного установочого пакета. В результате будут созданы пакеты, описанные в файле control.

   Пример:

libgala2_2.0-0ubuntu1_i386.deb	Бинарный пакет libgala2
libgala-dev_2.0-0ubuntu1_i386.deb	Бинарный пакет libgala-dev

6. **Бинарные пакеты готовы к установке.**
7. **Для валидации построенных пакетов необходимо запустить программу lintian[2], указав ей нужные файлы**.

Ссылки к разделу 3:

Официальное руководство на английском:

[1] https://wiki.ubuntu.com/PackagingGuide

[2] http://lintian.debian.org

Раздел 4. Интеграция и поддержка пакетов в репозитории Ubuntu

Для интеграции пакета в центральные репозитории необходимо выполнение следующих условий:

1. Безошибочно должны строиться установочные пакеты: бинарные и пакет исходных кодов.
2. Пакеты должны быть подписаны цифровой подписью автора установочных пакетов, указанного в changelog. Для этого при построении пакетов с помощью dpkg-buildpackage необходимо указать ключ –k<key_id>, где key_id – идентификатор ключа в хранилище ключей gpg [1].
3. Пакеты должны безошибочно валидироваться последней версией lintian.
4. Пакеты должны соответствовать всем требованиям на оформление служебных файлов. Дополнительную информацию о требованиях можно получить по ссылке [2].
5. Пакет должен содержать только лицензионно чистый код. Относительно каждого файла должна быть четко указана его лицензия в служебном файле copyright, это требование распространяется не только на исходный код, но и на документацию, служебные данные, и прочие файлы.
6. Инженер, создающий пакеты должен зарегистрироваться в системе Launchpad [3] и зарегистрировать в системе свой OpenPGP ключ. Инструкция по регистрации ключа доступна после регистрации пользователя в системе.

4.1. Алгоритм регистрации пакета в очереди на верификацию менторами проекта Ubuntu:

1. Зарегистрировать нового пользователя в системе Launchpad и добавить в систему цифровой ключ.
2. Создать запись в баг-трекере Launchpad, с предложением добавления пакета в репозитории. Запись должна относиться к проекту Ubuntu, называться [needs-packaging] <packagename> и содержать краткую информацию о проекте и ссылку на его веб-сайт.
3. Внести в changelog строчку * Initial packaging (LP: #XXXXXX), где XXXXXX – номер записи в баг-трекере.
4. Построить установочный пакет исходных кодов, подписанный цифровой подписью, используя команду dpkg-buildpackage –S –sa –k<key_id>.
5. Выполнить команду dput revu <packagename>-<packagever>_source.changes.

6. Если все сделано верно, пакет будет загружен на веб-портал http://revu.ubuntuwire.net
7. После этого пакет будет ожидать ревью менторов проекта Ubuntu. Для ускорения процесса, менторов можно искать на IRC-канале #ubuntu-motu сервера freenode.net.
8. После исправления полученных замечаний, процесс повторяется. После того как ментор одобрит пакет, он будет включен в репозитории Ubuntu.

4.2. Алгоритм обновления пакета, содержащегося в репозиториях Ubuntu:

1. Требуется пересмотреть все служебные файлы: возможно потребуется обновление версий, например, содержащейся в файле compat, либо Standards-version в файле control.
2. Обновить лицензионную информацию в файле copyright, если потребуется.
3. Если пакет будет включен в другую версию Ubuntu, нужно обновить имя дистрибутива в файле changelog.
4. Внести новую запись в changelog, описывающую причину обновления и все внесенные изменения. При этом должна обновиться либо версия оригинального пакета, либо версия сборки -0ubuntu1 (увеличивается последнее число: -0ubuntu2 и т. д.)
5. Собрать установочный пакет исходных кодов, подписанный цифровой подписью.
6. Завести запись в баг-трекере системы Launchpad, содержащую запрос на обновление пакета. Приложить к записи построенный пакет.
7. Ожидать одобрения внесенных изменений менторами проекта.

Ссылки к разделу:

[1] http://www.gnupg.org/

[2] https://wiki.ubuntu.com/UbuntuDevelopment/CodeReviews

[3] http://launchpad.net

[4] https://wiki.ubuntu.com/UbuntuDevelopment/NewPackages