Справочник разработчика Debian

Авторы «Справочника разработчика», Андреас Барт,
Адам Ди Карло, Рафаэль Герцог, Лукас Нуссбаум,
Кристиан Шварц, Иен Джексон

3 ноября 2017 г.

Справочник разработчика Debian
by Авторы «Справочника разработчика», Андреас Барт, Адам Ди Карло, Рафаэль Герцог, Лукас Нуссбаум, Кристиан Шварц, Иен Джексон

Published 2017-10-28
Copyright © 2004, 2005, 2006, 2007 Андреас Барт
Copyright © 1998, 1999, 2000, 2001, 2002, 2003 Адам Ди Карло
Copyright © 2002, 2003, 2008, 2009 Рафаэль Герцог
Copyright © 2008, 2009 Лукас Нуссбаум
Copyright © 1997, 1998 Кристиан Шварц

Данное руководство является Свободным ПО; вы можете распространять его и/или изменять в соответствии с условиями Стандартной общественной лицензии GNU, опубликованной Free Software Foundation; либо версии 2, либо (по вашему усмотрению) любой более поздней версии.

Руководство распространяется в надежде на то, что оно будет кому-нибудь полезно, но *без каких-либо гарантий*; даже без гарантии товарного состояния или пригодности его использования в конкретных целях. Для получения более подробной информации ознакомьтесь со Стандартной общественной лицензией GNU.

Копия Стандартной общественной лицензии GNU доступна в `/usr/share/common-licenses/GPL-2` в дистрибутиве Debian, либо во Всемирной паутине на веб-сайте GNU. Также вы можете получить печатную версию, обратившись по адресу: Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Если вы хотите распечатать данное руководство, вам следует использовать pdf-версию. Данная страница также доступна на английском, французском, немецком, итальянском, русском и японском языках.

Оглавление

1 Границы данного документа **1**

2 Подача заявки на получение статуса сопровождающего **3**
2.1 Приступаем к работе . . . 3
2.2 Наставники и поручители Debian . . . 3
2.3 Регистрация в качестве разработчика Debian . . . 4

3 Обязанности разработчика Debian **7**
3.1 Обязанности сопровождающего пакетов . . . 7
3.1.1 Работа по подготовке следующего `стабильного` выпуска . . . 7
3.1.2 Сопровождение пакетов в `стабильном` выпуске . . . 7
3.1.3 Работа с критичными для выпуска ошибками . . . 7
3.1.4 Координация с разработчиками основной ветки . . . 8
3.2 Административные обязанности . . . 8
3.2.1 Сопровождение вашей связанной с Debian информации . . . 8
3.2.2 Сопровождение вашего открытого ключа . . . 8
3.2.3 Голосование . . . 9
3.2.4 Вежливый уход в отпуск . . . 9
3.2.5 Уход в отставку . . . 9
3.2.6 Возвращение после ухода . . . 10

4 Ресурсы для разработчиков и сопровождающих Debian **11**
4.1 Списки рассылки . . . 11
4.1.1 Базовые правила . . . 11
4.1.2 Базовые списки рассылки . . . 11
4.1.3 Специальные списки рассылки . . . 12
4.1.4 Запрос новых списков рассылки, связанных с разработкой . . . 12
4.2 Каналы IRC . . . 12
4.3 Документация . . . 13
4.4 Машины Debian . . . 13
4.4.1 Сервер bugs . . . 13
4.4.2 Сервер ftp-master . . . 13
4.4.3 Сервер www-master . . . 14
4.4.4 Веб-сервер people . . . 14
4.4.5 Серверы систем управления версиями . . . 14
4.4.6 chroot для различных выпусков . . . 14
4.5 База данных разработчиков . . . 14
4.6 Архив Debian . . . 15
4.6.1 Разделы . . . 16
4.6.2 Архитектуры . . . 17
4.6.3 Пакеты . . . 17
4.6.4 Выпуски . . . 17
4.6.4.1 Стабильный, тестируемые и нестабильный выпуски . . . 17
4.6.4.2 Дополнительная информация о тестируемом выпуске . . . 18
4.6.4.3 Экспериментальный выпуск . . . 18
4.6.5 Кодовые имена выпусков . . . 19
4.7 Зеркала Debian . . . 19
4.8 Система входящих пакетов . . . 19
4.9 Информация о пакете . . . 20
4.9.1 В веб . . . 20
4.9.2 Утилита **dak ls** . . . 20
4.10 Система отслеживания пакетов Debian . . . 21
4.10.1 Интерфейс электронной почты для системы отслеживания пакетов . . . 21
4.10.2 Фильтрация почты от системы отслеживания пакетов . . . 22

4.10.3 Пересылка коммитов системы управления версиями в систему отслеживания пакетов 22
4.10.4 Веб-интерфейс системы отслеживания пакетов . . . 23
4.11 Обзор пакетов разработчика . . . 23
4.12 Установка Debian FusionForge: Alioth . . . 23
4.13 Плюшки для разработчиков и сопровождающих Debian . . . 24

5 Управление пакетами 25
5.1 Новые пакеты . . . 25
5.2 Запись изменений в пакете . . . 26
5.3 Тестирование пакета . . . 26
5.4 Схема пакета с исходным кодом . . . 27
5.5 Выбор выпуска . . . 27
5.5.1 Специальный случай: загрузка в `стабильный` и `предыдущий стабильный` выпуски 27
5.5.2 Специальный случай: загрузка в `testing/testing-proposed-updates` . . . 28
5.6 Загрузка пакета . . . 28
5.6.1 Загрузка на `ftp-master` . . . 28
5.6.2 Задержанные загрузки . . . 29
5.6.3 Загрузки безопасности . . . 29
5.6.4 Другие очереди загрузки . . . 29
5.6.5 Notifications . . . 29
5.7 Определение раздела для пакета, подраздела и приоритета . . . 29
5.8 Работа с ошибками . . . 30
5.8.1 Мониторинг ошибок . . . 30
5.8.2 Ответ на ошибки . . . 30
5.8.3 Работа с ошибками . . . 31
5.8.4 Когда ошибки исправляются путём новых загрузок . . . 32
5.8.5 Работа с ошибками, связанными с безопасностью . . . 33
5.8.5.1 Система отслеживания безопасности . . . 34
5.8.5.2 Конфиденциальность . . . 34
5.8.5.3 Рекомендации по безопасности . . . 34
5.8.5.4 Подготовка пакетов для решения проблем безопасности . . . 35
5.8.5.5 Загрузка исправленного пакета . . . 36
5.9 Перемещение, удаление, переименование, придание статуса осиротевшего, усыновление и повторное введение пакетов . . . 36
5.9.1 Перемещение пакетов . . . 36
5.9.2 Удаление пакетов . . . 37
5.9.2.1 Удаление пакетов из каталога `Incoming` (каталога входящих пакетов) . . . 38
5.9.3 Замена или переименование пакетов . . . 38
5.9.4 Придание статуса осиротевшего пакета . . . 38
5.9.5 Усыновление пакета . . . 38
5.9.6 Повторное добавление пакетов . . . 39
5.10 Работа на переносом и перенос пакетов . . . 39
5.10.1 Будьте добры к тем, кто занимается переносом . . . 40
5.10.2 Руководство для загрузок теми, кто занимается переносом . . . 41
5.10.2.1 Повторная компиляция или только двоичные NMU . . . 41
5.10.2.2 Когда следует делать NMU, если вы занимаетесь переносом . . . 41
5.10.3 Инфраструктура переноса и автоматизация . . . 42
5.10.3.1 Списки рассылки и веб-страницы . . . 42
5.10.3.2 Инструменты переноса . . . 42
5.10.3.3 `wanna-build` . . . 42
5.10.4 Если ваш пакет *не* может быть перенесён . . . 43
5.10.5 Отмечаем несвободные пакеты как собираемые автоматически (auto-buildable) . . . 43
5.11 Загрузки не-сопровождающим (NMU) . . . 44
5.11.1 Когда и как делать NMU . . . 44
5.11.2 NMU и файл `debian/changelog` . . . 45
5.11.3 Использование очереди `DELAYED/` . . . 45
5.11.4 NMU с точки зрения сопровождающего . . . 46
5.11.5 NMU исходного кода и двоичные NMU (binNMU) . . . 46
5.11.6 NMU и загрузки командой контроля качества . . . 46

5.11.7 NMU и командные загрузки . . . 47
5.12 Совместное сопровождение . . . 47
5.13 Тестируемый выпуск . . . 48
5.13.1 Основы . . . 48
5.13.2 Обновления из нестабильного выпуска . . . 48
5.13.2.1 Устаревание . . . 49
5.13.2.2 Удаление из тестируемого выпуска . . . 49
5.13.2.3 Круговые зависимости . . . 49
5.13.2.4 Влияние пакета в тестируемом выпуске . . . 50
5.13.2.5 Подробности . . . 50
5.13.3 Прямые обновления тестируемого выпуска . . . 50
5.13.4 Часто задаваемые вопросы . . . 51
5.13.4.1 Что такое критичные для выпуска ошибки, как производится их подсчёт? . . . 51
5.13.4.2 Как установка какого-то пакета в `тестируемый` выпуск может сломать другие пакеты? . . . 51

6 Лучшие практики создания пакетов 53
6.1 Лучшие практики для `debian/rules` . . . 53
6.1.1 Сценарии-помощники . . . 53
6.1.2 Отделение ваших заплат в несколько файлов . . . 54
6.1.3 Множественные двоичные пакеты . . . 54
6.2 Лучшие практики для `debian/control` . . . 54
6.2.1 Общие принципы описания пакетов . . . 54
6.2.2 Резюме пакета или короткое описание . . . 55
6.2.3 Длинное описание . . . 55
6.2.4 Домашняя страница основной ветки разработки . . . 56
6.2.5 Размещение системы контроля версий . . . 56
6.2.5.1 Vcs-Browser . . . 56
6.2.5.2 Vcs-* . . . 56
6.3 Лучшие практики для `debian/changelog` . . . 57
6.3.1 Написание полезных пунктов файла изменений . . . 57
6.3.2 Распространённые неправильные представления о записях об изменениях . . . 57
6.3.3 Общие ошибки в записях об изменениях . . . 58
6.3.4 Дополнение журналов файлами `NEWS.Debian` . . . 58
6.4 Лучшие практики для сценариев сопровождающих . . . 59
6.5 Управление настройкой с помощью `debconf` . . . 60
6.5.1 Не злоупотребляйте debconf . . . 60
6.5.2 Общие рекомендации для авторов и переводчиков . . . 60
6.5.2.1 Пишите на правильном английском . . . 60
6.5.2.2 Будьте добры к переводчикам . . . 60
6.5.2.3 Отменяйте статус неясных строк когда исправляете опечатки или орфографию . . . 61
6.5.2.4 Не допускайте ничего по поводу интерфейсов . . . 62
6.5.2.5 Не используйте первое лицо . . . 62
6.5.2.6 Будьте нейтральны в гендерном отношении . . . 62
6.5.3 Определение полей шаблонов . . . 62
6.5.3.1 Тип . . . 62
6.5.3.1.1 string . . . 62
6.5.3.1.2 password . . . 62
6.5.3.1.3 boolean . . . 62
6.5.3.1.4 select . . . 62
6.5.3.1.5 multiselect . . . 63
6.5.3.1.6 note . . . 63
6.5.3.1.7 text . . . 63
6.5.3.1.8 error . . . 63
6.5.3.2 Описание: краткое и расширенное описания . . . 63
6.5.3.3 Choices . . . 64
6.5.3.4 Default . . . 64
6.5.4 Руководство по стилю в зависимости от полей шаблона . . . 64

6.5.4.1 Тип поля . . . 64
6.5.4.2 Поле Description . . . 64
6.5.4.2.1 Шаблоны string/password . . . 64
6.5.4.2.2 Шаблоны boolean . . . 64
6.5.4.2.3 select/multiselect . . . 64
6.5.4.2.4 Примечания . . . 64
6.5.4.3 Поле Choices . . . 65
6.5.4.4 Поле Default . . . 65
6.5.4.5 Поле Default . . . 65
6.6 Интернационализация . . . 65
6.6.1 Обработка переводов debconf . . . 66
6.6.2 Интернационализованная документация . . . 66
6.7 Общие ситуации при создании пакетов . . . 66
6.7.1 Пакеты, использующие **autoconf/automake** . . . 66
6.7.2 Библиотеки . . . 66
6.7.3 Документация . . . 67
6.7.4 Конкретные типы пакетов . . . 67
6.7.5 Независящие от архитектуры данные . . . 67
6.7.6 Необходимость в конкретной локали во время сборки . . . 68
6.7.7 Делаем так, чтобы deborphan определят переходные пакеты . . . 68
6.7.8 Лучшие практики для файлов `.orig.tar.{gz,bz2,xz}` . . . 68
6.7.8.1 Чистый исходный код . . . 68
6.7.8.2 Повторная упаковка исходного кода основной ветки . . . 69
6.7.8.3 Изменение двоичных файлов . . . 70
6.7.9 Лучшие практики для отладочных пакетов . . . 70
6.7.10 Лучшие практики для метапакетов . . . 70

7 Помимо создания пакетов **71**
7.1 Отправка отчётов об ошибках . . . 71
7.1.1 Отправка множества отчётов об ошибках за один раз (массовое заполнение отчётов об ошибках) . . . 71
7.1.1.1 Пользовательские метки . . . 72
7.2 Работа по контролю качества . . . 72
7.2.1 Ежедневная работа . . . 72
7.2.2 Вечеринки по исправлению ошибок . . . 72
7.3 Связь с другими сопровождающими . . . 73
7.4 Работа с неактивными и/или недоступными сопровождающими . . . 73
7.5 Взаимодействие с будущими разработчиками Debian . . . 74
7.5.1 Поручение пакетов . . . 74
7.5.1.1 Поручительство нового пакета . . . 75
7.5.1.2 Поручение обновления существующего пакета . . . 76
7.5.2 Поддержка новых разработчиков . . . 77
7.5.3 Обработка заявок новых сопровождающих . . . 77

8 Интернационализация и переводы **79**
8.1 Как переводы обрабатываются в Debian . . . 79
8.2 ЧаВО по I18N и L10N для сопровождающих . . . 80
8.2.1 Как перевести некоторый данный текст . . . 80
8.2.2 Как проверить перевод . . . 80
8.2.3 Как обновить перевод . . . 80
8.2.4 Как работать с отчётом об ошибке, касающемся перевода . . . 80
8.3 ЧаВО по I18N и L10N для переводчиков . . . 80
8.3.1 Как помочь с переводом . . . 81
8.3.2 Как предоставить перевод для добавления его в пакет . . . 81
8.4 Лучшие текущие практики, касающиеся локализации . . . 81

A Обзор инструментов Debian для сопровождающего **83**
A.1 Базовые инструменты . . . 83
A.1.1 `dpkg-dev` . . . 83
A.1.2 `debconf` . . . 83
A.1.3 `fakeroot` . . . 84
A.2 Инструменты для проверки пакетов на предмет ошибок и соответствия стандартам . . . 84
A.2.1 `lintian` . . . 84
A.2.2 **debdiff** . . . 84
A.3 Помощники для `debian/rules` . . . 84
A.3.1 `debhelper` . . . 84
A.3.2 `dh-make` . . . 85
A.3.3 `equivs` . . . 85
A.4 Сборщики пакетов . . . 85
A.4.1 `git-buildpackage` . . . 85
A.4.2 `debootstrap` . . . 85
A.4.3 `pbuilder` . . . 85
A.4.4 `sbuild` . . . 86
A.5 ПО для загрузки пакетов . . . 86
A.5.1 `dupload` . . . 86
A.5.2 `dput` . . . 86
A.5.3 **dcut** . . . 86
A.6 Автоматизация сопровождения пакетов . . . 86
A.6.1 `devscripts` . . . 86
A.6.2 `autotools-dev` . . . 86
A.6.3 `dpkg-repack` . . . 87
A.6.4 `alien` . . . 87
A.6.5 `dpkg-dev-el` . . . 87
A.6.6 **dpkg-depcheck** . . . 87
A.7 Инструменты для переноса . . . 87
A.7.1 `dpkg-cross` . . . 87
A.8 Документация и информацию . . . 87
A.8.1 `docbook-xml` . . . 87
A.8.2 `debiandoc-sgml` . . . 88
A.8.3 `debian-keyring` . . . 88
A.8.4 `debian-el` . . . 88

Глава 1

Границы данного документа

Целью данного документа является обзор рекомендованных процедур и доступных разработчикам Debian ресурсов.

Обсуждаемые здесь процедуры включают в себя то, как стать сопровождающим (раздел 2, «Подача заявки на получение статуса сопровождающего»); как создавать новые пакеты (раздел 5.1, «Новые пакеты») и как загружать пакеты (раздел 5.6, «Загрузка пакета»); как работать с сообщениями об ошибках (раздел 5.8, «Работа с ошибками»); как перемещать, удалять или придавать пакетам статус осиротевших пакетов (раздел 5.9, «Перемещение, удаление, переименование, придание статуса осиротевшего, усыновление и повторное введение пакетов»); как переносить пакеты на другие архитектуры (раздел 5.10, «Работа на переносом и перенос пакетов»); и то, как и когда выполнять временные выпуски пакетов других сопровождающих (раздел 5.11, «Загрузки не-сопровождающим (NMU)»).

Обсуждаемые ресурсы в данном руководстве включают в себя списки рассылки (раздел 4.1, «Списки рассылки») и серверы (раздел 4.4, «Машины Debian»); обсуждается структура архива Debian (раздел 4.6, «Архив Debian»); объясняются разные серверы, которые принимают загружаемые пакеты (раздел 5.6.1, «Загрузка на `ftp-master`»); также обсуждаются ресурсы, которые могут помочь сопровождающим контролировать качество своих пакетов (раздел A, «Обзор инструментов Debian для сопровождающего»).

Должно быть ясно, что данное руководство не содержит обсуждения технических деталей пактов Debian и технических деталей их создания. Данное руководство не содержит подробных сведений о стандартах, которым должно отвечать ПО в Debian. Вся эта информация может быть найдена в Руководстве по политике Debian.

Более того, данный документ *не выражает формальной политики*. Он содержит документацию по системе Debian и лучшие практики, принимаемые большинством. Таким образом, это не „нормативный“ документ.

Глава 2

Подача заявки на получение статуса сопровождающего

2.1 Приступаем к работе

So, you've read all the documentation, you've gone through the Debian New Maintainers' Guide (or its sucessor, Guide for Debian Maintainers), understand what everything in the `hello` example package is for, and you're about to Debianize your favorite piece of software. How do you actually become a Debian developer so that your work can be incorporated into the Project?

Во-первых, подпишитесь на debian-devel@lists.debian.org, если вы ещё не оформили подписку. Отправьте слово `subscribe` в поле `Тема` вашего сообщения электронной почты на адрес debian-devel-REQUEST@lists.debian.org. В случае возникновения проблем свяжитесь с администратором списка по адресу listmaster@lists.debian.org. Дополнительную информацию о доступных списках рассылки см. в разделе 4.1, «Списки рассылки». Ещё один обязательный список рассылки debian-devel-announce@lists.debian.org, для тех кто хочет следить за разработкой и развитием Debian.

Вам следует подписаться и ненадолго скрыться (то есть, читать сообщения и ничего не отправлять в список) до начала какой-либо работы над кодом, вам следует писать о ваших намерениях по реализации чего-либо для того, чтобы избежать повторов в работе.

Ещё одним хорошим списком рассылки, на который стоит подписаться, является debian-mentors@lists.debian.org. Подробности см. в разделе 2.2, «Наставники и поручители Debian». IRC-канал `#debian` также может быть полезен, см. раздел 4.2, «Каналы IRC».

Когда вы решите, как вы хотите участвовать в Debian, вам следует связаться с существующими сопровождающими Debian, которые работают над схожими задачами. Это позволит вам научиться у опытных разработчиков. Например, если вы заинтересованы в создании пакетов Debian для существующего ПО, вам следует попытаться найти поручителя. Поручитель будет работать с вами над вашим пакетом и загрузит его в архив Debian, когда он будет доволен проделанной вами работой по созданию пакета. Вы можете найти поручителя отправив сообщение в список рассылки debian-mentors@lists.debian.org с описанием вашего пакета и себя самого и с просьбой о поручительстве (для дополнительной информации о поручительстве см. раздел 7.5.1, «Поручение пакетов» и `https://wiki.debian.org/DebianMentorsFaq`). С другой стороны, если вы заинтересованы в переносе Debian на альтернативные архитектуры или ядра, вы можете подписаться на конкретные списки рассылки и спросить там о том, как начать работу. Наконец, если вы заинтересованы в работе над документацией или в контроле качества (QA), вы можете присоединиться к сопровождающим, которые уже работают над этими задачами и отправляют заплаты и улучшения.

Одной из проблем может быть то, что локальная часть вашего адреса электронной почты имеет слишком общий характер: следует избегать таких терминов как mail, admin, root, master, подробнее см. `https://www.debian.org/MailingLists/`.

2.2 Наставники и поручители Debian

Список рассылки debian-mentors@lists.debian.org создан для начинающих сопровождающих, которые ищут помощи в работе над созданием пакетов или в решении связанных с разработкой проблем. Каждому новому разработчику рекомендуется подписаться на этот список рассылки, подробнее см. раздел 4.1, «Списки рассылки».

Те, кто предпочитает помощь в режиме «один-на-один» (напр., через частную электронную почту), также должны написать в этот список рассылки и опытные разработчики помогут вам.

Кроме того, если у вас имеются пакеты, готовые для добавления в Debian, но вы ожидаете прохождения процедуры получения статуса нового сопровождающего, в этом списке рассылки вы можете найти поручителя для загрузки вашего пакета. Поручители — люди, которые являются официальными разработчиками Debian и которые желают проверить ваши пакеты и загрузить их. Пожалуйста, для начала прочтите ЧаВО списка рассылки debian-mentors по адресу `https://wiki.debian.org/DebianMentorsFaq`.

Если вы хотите выступить в качестве наставника и/или поручителя, см. дополнительную информацию в разделе 7.5, «Взаимодействие с будущими разработчиками Debian».

2.3 Регистрация в качестве разработчика Debian

До того как вы решите зарегистрировать в Debian, вам необходимо прочитать всю доступную информацию в Уголке новых членов Debian. Там описаны подробности подготовки, которую вам следует пройти до того, как вы сможете зарегистрироваться для получения статуса разработчика Debian. Например, до подачи заявки вам необходимо прочитать Общественный договор Debian. Ваша регистрация как разработчика предполагает, что вы согласны и ручаетесь придерживаться Общественного договора Debian; очень важно, что сопровождающие согласны с ключевыми идеями, стоящими за Debian. Чтение Манифеста GNU также может быть хорошей идеей.

Регистрация в качестве разработчика представляет собой процесс проверки вашей личности и ваших намерений, а также проверку ваших технических навыков. Поскольку число людей, работающих над Debian, переросло 1000, и поскольку наши системы используются в ряде очень важных мест, мы должны быть аккуратны. Следовательно, нам необходимо проверять новых сопровождающих до того, как мы предоставим их учётные записи на наших серверах и позволим им загружать пакеты.

До того как вы фактически зарегистрируетесь, вам следует показать, что вы можете выполнять компетентную работу и быть хорошим участником разработки. Вы можете показать это, отправляя заплаты через систему отслеживания ошибок и работая некоторое время над пакетом, имеющим поручителем существующего разработчика Debian. Кроме того, мы ожидаем, что участники заинтересованы во всём проекте целиком, а не просто в сопровождении своих собственных пакетов. Если мы можете помочь другим сопровождающим, предоставляя дополнительную информацию об ошибке или даже заплату, сделайте это!

Для регистрации необходимо, чтобы вы были знакомы с философией Debian и технической документацией. Более того, вам потребуется ключ GnuPG, которые должен быть подписан существующими сопровождающими Debian. Если ваш ключ GnuPG ещё не подписан, вам следует попытаться встретиться с разработчиком Debian лично и подписать у него свой ключ. Имеется страница для координации подписания ключа GnuPG, которая поможет вам найти разработчика Debian около вас. (Если около вас нет разработчиков Debian, может быть разрешён альтернативный путь прохождения проверки личности как некоторое абсолютное исключение в зависимости от конкретного случая. Дополнительную информацию см. на странице установления личности.)

Если у вас нет ключа OpenPGP, создайте его. Каждому разработчику необходим ключ OpenPGP для подписывания и проверки загрузок пакетов. Вам следует прочесть руководство для ПО, которое вы используете, поскольку оно содержит большое количество очень важной информации касательно безопасности. Большая часть проблем безопасности возникла из-за человеческих ошибок, а не из-за проблем с ПО или каких-то высокотехнологичных шпионских штучек. Дополнительную информацию о сопровождении вашего открытого ключа см. раздел 3.2.2, «Сопровождение вашего открытого ключа».

Debian использует `GNU Privacy Guard` (пакет `gnupg` версии 1 или новее) в качестве своего базового стандарта. Вы также можете использовать некоторые другие реализации OpenPGP. Заметьте, что OpenPGP является открытым стандартом на основе RFC 2440.

Для разработки Debian вам потребуется ключ версии 4. Длина вашего ключа должна быть больше 1024 бит. Оснований использовать меньший ключ нет и это намного менее безопасно.[1]

[1] Ключи версии 4 соответствуют стандарту OpenPGP, определённому в RFC 2440. Версия 4 представляет собой тот тип ключа, который всегда создаётся при использовании GnuPG. Версии PGP начиная с 5.x также могут создавать ключи версии 4, другой возможностью является PGP 2.6.x, который совместим с ключами версии v3 (в PGP они также называют наследуемыми RSA).

Ключи версии 4 (первичные) могут использовать либо алгоритм RSA, либо алгоритм DSA, поэтому это не имеет отношения к вопросу GnuPG о том, какой вид ключей вам нужен: (1) DSA и Elgamal, (2) DSA (только для подписи), (5) RSA (только для подписи). Если у вас нет специальных требований, выберите вариант по умолчанию.

Наиболее простым способом определить, является ли существующий ключ ключом версии 4 или 3 (или 2), состоит в том, чтобы проверить его отпечаток. Отпечатки ключей версии 4 являются SHA-1 хэшем некоторого материала ключа, поэтому они представляют собой 40 шестнадцатеричных чисел, обычно сгруппированных в блоки по 4. Отпечатки ключей более старых версий используют MD5 и обычно отображаются в виде блоков по 2 шестнадцатеричных числа. Например, если ваш отпечаток похож на `5B00 C96D 5D54 AEE1 206B  AF84 DE7A AF6E 94C0 9C7F`, то это ключ версии 4.

Если ваш открытый ключ не размещён на сервере открытых ключей, таком как `subkeys.pgp.net`, прочтите доступную по адресу Шаг 2: Установление личности информацию. Этот документ содержит инструкции о том, как поместить ваш ключ на серверы открытых ключей. Группа новых сопровождающих поместит ваш открытый ключ на серверы, если ключ ещё не был там размещён.

Некоторые страны ограничивают использование ПО для шифрования своими гражданами. Это не должно затруднить вашу деятельность как сопровождающего пакетов Debian, поскольку допускается использование продуктов шифрования для аутентификации, а не в целях зашифровать что-либо. Если вы живёте в стране, в которой запрещено использование шифрования даже для аутентификации, свяжитесь с нами, чтобы мы могли подготовить специальные договорённости.

Чтобы подать заявку на получение статуса нового сопровождающего, существующий разработчик Debian должен поддержать вашу заявку (`адвокат`). Если вы уже участвуете в Проекте Debian в течении некоторого времени и хотите подать заявку на получение статуса зарегистрированного разработчика, то существующий разработчик, с которым вы работали несколько последних месяцев, должен выразить свою убеждённость, что вы сможете успешно участвовать в развитии Проекта Debian.

Когда вы найдёте адвоката, ваш ключ GnuPG будет подписан, и у вас уже будет опыт участия в Проекте Debian, вы будете готовы подать заявку. Вы можете просто зарегистрироваться на нашей странице заявок. После того, как вы авторизуетесь, ваш адвокат должен подтвердить вашу заявку. Когда ваш адвокат завершит этот шаг, вам будет назначен менеджер заявки, который пройдёт вместе с вами все необходимые шаги процесса получения статуса нового сопровождающего. Вы всегда можете проверить свой статус в таблице статусов заявок.

Для получения подробных сведений, обратитесь в Уголок новых членов Debian на веб-сайте Debian. Убедитесь, что вам знакомы все необходимые шаги процесса получения статуса нового сопровождающего до того, как вы фактически подадите свою заявку. Если вы будете хорошо подготовлены, вы сможете сохранить большое количество времени.

Другой возможностью является передача ключа программе **pgpdump**, которая сообщит что-то вроде Public Key Packet - Ver 4.

Кроме того, заметьте, что ваш ключ должен быть подписан этим же ключом (т. е. должны быть подписаны все собственные идентификаторы пользователя; это предотвращает их подделку). Всё современное ПО OpenPGP делает это автоматически, но если у вас имеется более старый ключ, вам может потребоваться вручную добавить эти подписи.

Глава 3

Обязанности разработчика Debian

3.1 Обязанности сопровождающего пакетов

Как сопровождающий пакета вы должны предоставлять пакеты высокого качества, которые хорошо интегрируются в систему и соответствуют Политике Debian.

3.1.1 Работа по подготовке следующего `стабильного` выпуска

Предоставление пакетов высокого качества в `нестабильный выпуск` не достаточно, большинство пользователей извлекут пользу из ваших пакетов в том случае, если они будут выпущены в составе следующего `стабильного` выпуска. Поэтому вам следует взаимодействовать с командой, курирующей выпуск, чтобы подстраховаться, что ваши пакеты будут добавлены в выпуск.

Более конкретно, вам необходимо отслеживать, мигрировали ли ваши пакеты в `тестируемый` выпуск (см. раздел 5.13, «Тестируемый выпуск»). Если миграция не произошла после периода тестирования, вам следует выяснить, почему так получилось, и поработать над исправлением проблемы. Это может предполагать исправление пакета (в случае критичных для выпуска ошибок или проблем сборки на некоторых архитектурах), но также это может предполагать обновление (или исправление, или удаление из `тестируемого` выпуска) других пакетов, чтобы помочь в завершении перехода пакетов, в котором ваш пакет застрял из-за своих зависимостей. Команда по выпуску может предоставить вам некоторую информацию о факторах, блокирующих некоторый данный переход пакетов, в случае, если вы сами не можете их определить.

3.1.2 Сопровождение пакетов в `стабильном` выпуске

Большая часть работы сопровождающего пакетов сводится к загрузке обновлённых версий пакетов в `нестабильный` выпуск, но также предполагается отслеживание пакетов в текущем `стабильном` выпуске.

Хотя вносить изменения в `стабильный` выпуск не рекомендуется, они возможны. Когда сообщается о проблеме безопасности, вам следует совместно с командой безопасности предоставить исправленную версию (см. раздел 5.8.5, «Работа с ошибками, связанными с безопасностью»). Если сообщается об ошибках с важностью important (или более) в `стабильной` версии вашего пакета, вам следует подумать над предоставлением целевого исправления. Вы можете спросить команду `стабильного` выпуска о том, примут они такое обновление или нет, и затем уже подготовить загрузку в `стабильный` выпуск (см. раздел 5.5.1, «Специальный случай: загрузка в `стабильный` и `предыдущий стабильный` выпуски»).

3.1.3 Работа с критичными для выпуска ошибками

Обычно вам следует работать с сообщениями об ошибках в ваших пакетах так, как это описано в разделе 5.8, «Работа с ошибками». Тем не менее, имеется специальная категория ошибок, на которые следует обратить внимание — это так называемые критичные для выпуска ошибки (RC ошибки). Все такие сообщения об ошибках, имеющие важность `critical`, `grave` или `serious`, делают пакет неподходящим для включения в следующий `стабильный` выпуск. Таким образом, они могут задержать выпуск Debian (когда они затрагивают пакет из `тестируемого` выпуска), либо блокировать миграцию пакетов в `тестируемый` выпуск (когда они лишь затрагивают пакет из `нестабильного` выпуска). В худшем сценарии такие ошибки приведут к удалению пакета. Вот почему эти ошибки следует исправить как можно скорее.

Если по какой-либо причине вы не можете исправить критичную для выпуска ошибку в вашем пакете в течении двух недель (например, из-за нехватки времени, либо потому, что её очень сложно исправить), вам

следует явно указать это в сообщении об ошибке, также вам следует отметить ошибку тегом `help`, который используется для привлечения добровольцев. Учтите, что критичные для выпуска ошибки часто исправляются путём обновления и загрузки пакета теми, кто не является сопровождающим (см. раздел 5.11, «Загрузки не-сопровождающим (NMU)»), поскольку они блокируют переход многих пакетов в `тестируемый` выпуск.

Отсутствие внимания к критичным для выпуска ошибкам обычно интерпретируется командой контроля качества как знак того, что сопровождающий прекратил свою работу без корректного придания своему пакету статуса осиротевшего пакета. Команда по поиску пропавших (MIA) может подключиться к работе, что может привести к тому, что ваши пакеты получат статус осиротевших пакетов (см. раздел 7.4, «Работа с неактивными и/или недоступными сопровождающими»).

3.1.4 Координация с разработчиками основной ветки

Большая часть вашей работы в качестве сопровождающего Debian будет состоять во взаимодействии с разработчиками основной ветки. Пользователи Debian иногда будут сообщать об ошибках, которые не касаются непосредственно Debian. Вам следует пересылать эти сообщения об ошибках разработчикам основной ветки, чтоб они могли исправить эти ошибки в последующем выпуске основной ветки разработки.

Хотя вашей задачей не является исправление ошибок, которые не касаются непосредственно Debian, вы вполне можете это делать, если имеете необходимые знания и навыки. Когда вы вносите такие исправления, обязательно передайте их и сопровождающим основной ветки разработки. Пользователи и разработчики Debian иногда присылают заплаты, исправляющие ошибки основной ветки разработки — вам следует оценить их и переслать в основную ветку разработки.

Если вам необходимо изменить исходный код из основной ветки разработки для того, чтобы собрать пакет, соответствующий правилам, то вам следует предложить своё исправление разработчикам основной ветки, которые может быть включено ими в основную ветку разработки, чтобы вам не пришлось изменять исходный код последующих выпусков основной ветки. Какие бы изменения вы не производили, попытайтесь не создавать ответвлений от основной ветки.

Если вы обнаружите, что разработчики основной ветки враждебны по отношению к Debian или сообществу Свободного ПО, ещё раз подумайте о том, так ли необходимо включать данное ПО в Debian. Иногда социальная стоимость, которую платит сообщество Debian, не покрывается прибылью, которую может принести какое-либо ПО.

3.2 Административные обязанности

Проект, имеющий размеры, сопоставимые с Debian, полагается на некоторую административную инфраструктуру для того, чтобы всё можно было отследить. Как у члена проекта у вас будут некоторые обязанности, исполнение которых гарантирует, что всё идёт гладко.

3.2.1 Сопровождение вашей связанной с Debian информации

База данных LDAP, содержащая информацию о разработчиках Debian, расположена по адресу `https://db.debian.org/`. Вам следует ввести информацию о себе в эту базу данных и обновлять её по мере изменения. В первую очередь, убедитесь, что адрес электронной почты, на который пересылается ваша почта с ящика на debian.org, актуален, также проверьте ваш адрес, на который оформлена подписка на рассылку debian-private, если вы подписаны на неё.

Дополнительную информацию о базе данных см. в разделе 4.5, «База данных разработчиков».

3.2.2 Сопровождение вашего открытого ключа

Будьте аккуратны с вашими закрытыми ключами. Не размещайте их на публичных серверах или многопользовательских машинах, таких как серверы Debian (см. раздел 4.4, «Машины Debian»). Сделайте резервную копию ваших ключей; храните копию автономно. Прочтите документацию, поставляемую с вашим ПО; прочтите ЧаВО по PGP.

Вам следует убедиться в том, что ваш ключ не только защищён от кражи, но также и в том, что он не может быть потерян. Создайте и сделайте копию (лучше всего сделать также и бумажную копию) вашего сертификата отзыва ключа; он понадобится в случае потери ключа.

If you add signatures to your public key, or add user identities, you can update the Debian key ring by sending your key to the key server at `keyring.debian.org`. Updates are processed at least once a month by the `debian-keyring` package maintainers.

Если вам необходимо добавить абсолютно новый ключ или удалить старый, вам следует получить подпись на новом ключе от другого разработчика. Если старый ключ скомпрометирован, либо недействителен, вам также следует добавить сертификат отзыва ключа. Если реальных причин для создания нового ключа нет, сопровождающие брелока могут отклонить новый ключ. Подробности могут найдены по адресу `http://keyring.debian.org/replacing_keys.html`.

Применимы процедуры разворачивания ключа, обсуждаемые в разделе 2.3, «Регистрация в качестве разработчика Debian».

You can find a more in-depth discussion of Debian key maintenance in the documentation of the `debian-keyring` package and the http://keyring.debian.org/ site.

3.2.3 Голосование

Даже несмотря на то, что Проект Debian в действительности не представляет собой демократию, мы используем демократический процесс для выбора лидеров Проекта и утверждения общих решений. Эти процедуры определяются Конституцией Проекта Debian.

Помимо ежегодных выборов лидера Проекта, голосования не проводятся регулярно, и к ним не относятся легкомысленно. Каждое предложение сначала обсуждается в списке рассылки debian-vote@lists.debian.org, требуется некоторое одобрение любого предложения до того, как секретарь Проекта инициирует процедуру голосования.

Вам не нужно отслеживать предваряющие голосование обсуждения, поскольку секретарь опубликует несколько требований голосования в списке рассылки debian-devel-announce@lists.debian.org (все разработчики должны быть подписаны на этот список рассылки). Демократия не работает, если люди не принимают участия в голосовании, поэтому мы просим разработчиков голосовать. Голосование происходит при помощи GPG-подписанных/зашифрованных сообщений электронной почты.

Список всех предложений (прошлых и текущих) доступен на странице Информации о голосованиях Debian, там же доступна информация о том, как выдвинуть, поддержать и проголосовать по поводу предложения.

3.2.4 Вежливый уход в отпуск

Разработчики могут отсутствовать, это нормально, это может быть запланированный отпуск, либо они могут быть загружены другой работой. Важно, чтобы остальные разработчики знали о том, что вы находитесь в отпуске, чтобы они могли сделать то, что нужно, если возникнут проблемы в ваших пакетах, либо от вас требуется выполнение каких-либо других обязанностей в Проекте.

Обычно это означает, что другие разработчики могут осуществлять загрузки, не будучи сопровождающими (см. раздел 5.11, «Загрузки не-сопровождающим (NMU)»), в случае если во время вашего отсутствия возникнет какая-либо большая проблема (критичная для выпуска ошибка, обновление безопасности и т. д.). Иногда ничего критичного не происходит, но всё равно лучше предупредить остальных о том, что вы будете недоступны.

Для того, чтобы проинформировать других разработчиков, вам следует сделать две вещи. Во-первых, отправьте сообщение по адресу debian-private@lists.debian.org с [VAC] в начале темы сообщения[1], в сообщении укажите промежуток времени, когда вы будете находиться в отпуске. Вы также можете указать какие-либо специальные инструкции по поводу того, что нужно предпринять в случае, если возникнет проблема.

Далее, следует отметить себя как находящегося в отпуске в базе данных LDAP разработчиков Debian (эта информация доступна только разработчикам Debian). Не забудьте удалить флаг статуса «в отпуске» по своему возвращению!

В идеале вам следует зайти на координационные страницы GPG во время планирования своего отпуска и выяснить, хочет ли кто-либо подписать свой ключ. Это особенно важно, когда люди едут в экзотические места, где у нас пока нет разработчиков, но где есть заинтересованные в участии в Проекте люди.

3.2.5 Уход в отставку

Если вы решили покинуть Проект Debian, вам следует убедиться, что вы выполнили следующие шаги:

1. Придайте всем ваши пакетам статус осиротевших пакетов как это описано в разделе 5.9.4, «Придание статуса осиротевшего пакета».

[1] Это нужно для того, чтобы те, кто не хочет читать сообщения об отпусках, могли отфильтровать такие сообщения.

2. Вышлите подписанное с помощью gpg электронное письмо о том, что вы хотите покинуть Проект, по адресу debian-private@lists.debian.org.

3. Сообщите сопровождающим брелока Debian, что вы выходите из Проекта, открыв запрос в Debian RT путём отправления сообщения на адрес keyring@rt.debian.org со словами «Debian RT» в теме сообщения (регистр не имеет значения).

4. Если вы получали сообщения через алиас @debian.org (напр. press@debian.org) и хотите удалить свой адрес из рассылки, откройте билет RT для системных администраторов Debian. Просто отправьте сообщение на адрес admin@rt.debian.org со словами ”Debian RT” где-нибудь в теме сообщения, в сообщении укажите, от каких алиасов вы более не хотите получать сообщения.

Важно, чтобы весь приведённый выше процесс был соблюдён, поскольку поиск неактивных разработчиком и пометка их пакетов как осиротевших занимает значительное количество времени и требует усилий.

3.2.6 Возвращение после ухода

Учётная запись разработчика получает отметку «emeritus» в случае, если выход в отставку выполнен в соответствии с разделом 3.2.5, «Уход в отставку», и «disabled» в противном случае. Вышедшие в отставку разработчики, имеющие учётную запись с отметкой «emeritus», могут заново активировать свою учётную запись следующим образом:

- Свяжитесь с da-manager@debian.org.
- Пройдите через укороченную процедуру получения статуса нового члена (это необходимо для того, чтобы убедиться, что возвращающийся разработчик всё ещё помнит важные части P&P и T&S).
- Докажите, что вы всё ещё контролируете ключ GPG, ассоциированный с учётной записью, либо предоставьте доказательство того, что новый ключ GPG принадлежит именно вам, для этого необходимы как минимум две подписи других разработчиков.

Прошлые разработчики, учётные записи которых были «отключены», должны заново проходить через процесс NM.

Глава 4

Ресурсы для разработчиков и сопровождающих Debian

В данной главе вы найдёте очень краткую дорожную карту списков рассылки Debian, машин Debian, которые доступны вам как разработчику, а также всех других доступных вам ресурсов, которые могут помочь в вашей работе в качестве сопровождающего.

4.1 Списки рассылки

Большая часть общения между разработчиками Debian (а также пользователями) происходит в списках рассылки, которые расположены по адресу `lists.debian.org`. Чтобы узнать о том, как подписаться или отписаться, как отправлять сообщения и как не отправлять их, где искать старые сообщения и как осуществлять поиск по ним, как связаться с сопровождающими данного списка рассылки и как просмотреть другую информацию о данном списке рассылки, прочтите `https://www.debian.org/MailingLists/`. В данном разделе приводятся лишь некоторые аспекты списков рассылки, которые особенно интересны разработчикам.

4.1.1 Базовые правила

При ответе на сообщения в списке рассылки не отправляйте копию сообщения (`CC`) автору сообщения, на которое вы отвечаете, если он об этом не попросил явным образом. Всякий, кто отправляет сообщения в список рассылки, должен быть подписан на него, чтобы видеть ответы.

Отправление нескольких одинаковых сообщений в разные списки рассылки не желательно. Как это обычно делается при работе в сети, сокращайте цитаты из сообщений, на которые вы отвечаете. В общем придерживайтесь обычных договорённостей по отправке сообщений.

Для получения дополнительной информации ознакомьтесь с нормами поведения. Также стоит прочитать Руководство по взаимодействию в сообществе Debian.

4.1.2 Базовые списки рассылки

Базовые списки рассылки Debian, которыми следует пользоваться разработчикам:

- debian-devel-announce@lists.debian.org используется для анонсирования важной для разработчиков информации. Ожидается, что все разработчики подписаны на этот список рассылки.
- debian-devel@lists.debian.org используется для обсуждения различных вопросов разработки, связанных с техническими проблемами.
- debian-policy@lists.debian.org используется для обсуждения Политики Debian, а также для голосования по её изменению.
- debian-project@lists.debian.org используется для обсуждения различных не технических проблем, связанных с Проектом.

Существуют и другие списки рассылки для обсуждения специальных вопросов; см. список по адресу `https://lists.debian.org/`.

4.1.3 Специальные списки рассылки

debian-private@lists.debian.org представляет собой специальный список рассылки для частных дискуссий между разработчиками Debian. Предполагается, что он будет использоваться для сообщений, которые по какой-либо причине не должны быть доступны широкой публике. Как таковой это небольшой список рассылки, пользователям рекомендуется не использовать debian-private@lists.debian.org, если только это не является действительно необходимым. Более того, *не* пересылайте сообщения из этого списка кому бы то ни было. Архивы данного списка не доступны в сети по очевидным причинам, но вы можете просмотреть их, используя вашу учётную запись для командной оболочки на `master.debian.org`, архив находится в каталоге `~debian/archive/debian-private/`.

debian-email@lists.debian.org представляет собой специальный список рассылки, используемый для сбора связанной с Debian корреспонденции, такой как взаимодействие с авторами основной ветки разработки по поводу лицензий, ошибок и т. д., либо для обсуждения Проекта с другими пользователями и разработчиками, если сохранение этого обсуждения может оказаться полезным.

4.1.4 Запрос новых списков рассылки, связанных с разработкой

До отправки запроса по поводу списка рассылки, связанного с разработкой пакета (или небольшой группы связанных пакетов), решите подходит для этого использование псевдонима (через файл .forward-aliasname на master.debian.org, который переводит адрес в довольно хороший адрес *you-aliasname@debian.org*) или самостоятельно управляемый список рассылки на Alioth.

Если вы решили, что обычный список рассылки на lists.debian.org — это то, что вы хотите, то заполните запрос, следуя данному руководству.

4.2 Каналы IRC

Несколько каналов IRC специально посвящены разработке Debian. В основном они размещены в сети сообщества открытых и свободных технологий (OFTC). Запись DNS `irc.debian.org` является псевдонимом для `irc.oftc.net`.

Основным общим каналом для Debian является `#debian`. Это большой канал общего назначения, где пользователи могут найти свежие новости о Проекте, этот канал обслуживается роботами. `#debian` предназначен для говорящих на английском языке; также имеются `#debian.de`, `#debian-fr`, `#debian-br` и другие каналы со сходными названиями для тех, кто говорит на других языках.

Основным каналом по вопросам разработки Debian является `#debian-devel`. Это довольно активный канал; обычно на нём присутствует не менее 150 человек в любое время дня. Это канал для тех, кто хочет работать над Debian, это не канал поддержки (это этого используется канал `#debian`). Тем не менее, этот канал открыт для всех, кто хочет затаиться (и научиться). Его тематика состоит в обсуждении любой информации, которая интересна разработчикам.

Поскольку `#debian-devel` является открытым каналом, вам не следует обсуждать в нём то, что обсуждается в debian-private@lists.debian.org. Для этого цели имеется другой канал, который имеет имя `#debian-private`, он защищён ключом. Ключ доступен по адресу `master.debian.org:~debian/misc/irc-password`.

Имеются и другие дополнительные каналы, специально предназначенные для конкретных тем. `#debian-bugs` используется для координации вечеринок по исправлению ошибок. `#debian-boot` используется для координации работы над программой установки Debian. `#debian-doc` время от времени используется для разговоров по поводу документации, например, для обсуждения документации, которую вы читаете в настоящий момент. Другие каналы выделены для архитектур или наборов пакетов: `#debian-kde`, `#debian-dpkg`, `#debian-jr`, `#debian-edu`, `#debian-oo` (пакет OpenOffice.org)...

Существуют также некоторые каналы для разработчиков, родным языком которых не является английский, например `#debian-devel-fr` для людей, говорящих на французском языке и заинтересованных в разработке Debian.

В других сетях IRC также существуют каналы, посвящённые Debian, в частности в IRC сети freenode, на который до 4 июня 2006 года работало перенаправление с псевдонима `irc.debian.org`.

Для получения скрытия (cloak) в сети freenode отправьте Йоргу Ясперту (Jörg Jaspert) <joerg@debian.org> подписанное вашим ключом сообщение электронной почты с указанием вашего псевдонима. Добавьте где-нибудь слово cloak в заголовке Subject:. Ваш псевдоним должен быть зарегистрирован, см. страницу настройки псевдонима. Сообщение должно быть подписано ключом, который входит в брелок Debian. Дополнительную информацию о скрытии см. в документации Freenode.

4.3 Документация

Данный документ содержит большое количество полезной разработчикам Debian информации, но он не может содержать всё. Ссылки на большинство других интересных документов приведены в Уголке разработчика. Потратьте некоторое время на то, чтобы ознакомиться с ними, и вы обязательно узнаете много нового.

4.4 Машины Debian

У Debian имеется несколько компьютеров, которые работают в качестве серверов, большинство из них обслуживают критически важные функции в Проекте Debian. Большая часть машин используется для переноса пакетов, все эти машины имеют постоянное подключение к сети Интернет.

Некоторые машины доступны для использования отдельными разработчиками до тех пор, пока эти разработчики соблюдают правила, приведённые в Политике использования машин Debian.

Вообще говоря, вы можете использовать эти машины для связанных с Debian целей, если это кажется вам подходящим. Пожалуйста, будьте добры с системными администраторами, не используйте большое количество места на диске, пропускной способности сети, либо процессорного времени без предварительного разрешения системных администраторов. Обычно эти машины обслуживаются добровольцами.

Позаботьтесь о защите ваших паролей Debian и ключей SSH, установленных на машинах Debian. Избегайте входа или методов загрузки, которые пересылают пароли по сети Интернет в открытом виде, такие как Telnet, FTP, POP и т. д.

Пожалуйста, не размещайте какой-либо материал, не связанный с Debian, на серверах Debian, если у вас нет на это соответствующего разрешения.

Текущий список машин Debian доступен по адресу: `https://db.debian.org/machines.cgi`. Эта веб-страница содержит имена машин, контактную информацию, информацию о том, кто может входить на данную машину, ключи SSH и т. д.

Если у вас имеются проблемы с работой сервера Debian, и вы считаете, что системные операторы должны быть извещены об этой проблеме, вы можете обратиться к списку открытых проблем в очереди DSA в нашей системе отслеживания билетов `https://rt.debian.org/` (вы можете войти под пользователем «debian», пароль же доступен по адресу: `master.debian.org:~debian/misc/rt-password`). Чтобы сообщить о новой проблеме, вам нужно лишь отправить сообщение по адресу admin@rt.debian.org и убедиться, что в теме вы указали «Debian RT». Чтобы связаться с командой DSA по электронной почте, для частной или привилегированной информации, которая не должна быть публичной, используйте адрес dsa@debian.org, а для всего остального адрес debian-admin@lists.debian.org. Команда DSA также может быть найдена на IRC-канале `#debian-admin` в сети OFTC.

Если у вас имеются проблемы с конкретной службой, которые не связаны с администрированием системы (такие как пакеты, которые следует удалить из архива, предложения для веб-сайта и т. д.), вам следует отправить сообщение об ошибке в псевдопакете „pseudo-package“. Информацию о том, как отправлять такие сообщения об ошибках, см. раздел 7.1, «Отправка отчётов об ошибках».

Некоторые корневые серверы ограничены, но информация с них зеркалируется на других серверах.

4.4.1 Сервер bugs

`bugs.debian.org` является каноническим размещением системы отслеживания ошибок.

Если вы планируете проводить какой-либо статистической анализ, либо обработку сообщений об ошибках Debian, то это можно сделать здесь. Тем не менее, пожалуйста, опишите ваши планы в сообщений в debian-devel@lists.debian.org до реализации чего бы то ни было, чтобы уменьшить объём ненужной работы или время на обработку запросов.

4.4.2 Сервер ftp-master

Сервер `ftp-master.debian.org` хранит каноническую копию архива Debian. Вообще говоря, пакет, загружаемый на ftp.upload.debian.org, в конце концов попадает на этот сервер, см. раздел 5.6, «Загрузка пакета».

Доступ у нему ограничен; зеркало доступно по адресу `mirror.ftp-master.debian.org`.

О проблемах с архивом Debian FTP следует сообщать как об ошибках в псевдопакете `ftp.debian.org`, либо по электронной почте на адрес ftpmaster@debian.org, также см. описание процедур в разделе 5.9, «Перемещение, удаление, переименование, придание статуса осиротевшего, усыновление и повторное введение пакетов».

4.4.3 Сервер www-master

Основной веб-сервер расположен по адресу `www-master.debian.org`. Он содержит официальные веб-страницы и представляет собой лицо Debian для большинства новичков.

Если вы обнаружите проблему с веб-сервером Debian, вам следует отправить сообщение об ошибке в псевдопакете `www.debian.org`. Не забудьте проверить, сообщил ли кто-либо уже об этой проблеме в систему отслеживания ошибок.

4.4.4 Веб-сервер people

`people.debian.org` представляет собой сервер, используемый для размещения собственных веб-страниц разработчиков обо всём, что связано с Debian.

Если у вас имеется какая-то касающаяся Debian информация, которую вы хотели бы разместить в веб, вы можете сделать это, поместив ваш материал в каталог `public_html` своего домашнего каталога на `people.debian.org`. Материал будет доступен по URL: `https://people.debian.org/~`*`ваш-идентификатор-пользо`*

Вам следует использовать именно это место потому, что мы создаём его резервную копию, на других узлах резервные копии не создаются.

Обычно единственная причина использовать другой узел заключается в том, что вы хотите опубликовать какие-то материалы, которые подпадают под ограничения экспорта с территории США, в это случае вы можете использовать один из тех серверов, которые размещены за пределами США.

Если у вас имеются какие-либо вопросы, отправьте сообщения по адресу debian-devel@lists.debian.org.

4.4.5 Серверы систем управления версиями

Если вам нужно использовать систему управления версиями для какой-либо из ваших задач в Debian, вы можете использовать один из существующих репозиториев, размещённых на Alioth, либо вы можете запросить создание нового проекта и выделения для него репозитория системы управления версиями по вашему выбору. Alioth поддерживает CVS (cvs.alioth.debian.org/cvs.debian.org), Subversion (svn.debian.org), Arch (tla/baz, оба на arch.debian.org), Bazaar (bzr.debian.org), Darcs (darcs.debian.org), Mercurial (hg.debian.org) и Git (git.debian.org). Если вы планируете сопровождать пакеты через репозиторий системы управления версиями, см. `https://wiki.debian.org/Alioth/PackagingProject`. Информацию о предоставляемых Alioth сервисах см. в разделе 4.12, «Установка Debian FusionForge: Alioth».

4.4.6 chroot для различных выпусков

На некоторых машинах доступны chroot для различных выпусков. Вы можете использовать их следующим образом:

```
vore$ dchroot unstable
Executing shell in chroot: /org/vore.debian.org/chroots/user/unstable
```

Во всех chroot доступны обычные домашние каталоги пользователей. Вы можете узнать то, какие chroot доступны, через `https://db.debian.org/machines.cgi`.

4.5 База данных разработчиков

База данных разработчиков, расположенная на `https://db.debian.org/`, представляет собой каталог LDAP для управления атрибутами разработчиков Debian. Вы можете использовать этот ресурс для поиска по списку разработчиков Debian. Часть этой информации также доступна через службу finger на серверах Debian, попробуйте выполнить **finger ваша-учётная-запись@db.debian.org** и посмотрите вывод этой команды.

Разработчики могут входить в базу данных для изменения различной информации о самих себе, как то:

- адрес для пересылки вашей почты debian.org
- подписка на debian-private
- нахождение в отпуске

- персональная информация, например, адрес, страна, широта и долгота места, где вы проживаете, и которые будут использованы на мировой карте разработчиков Debian, номера телефонов и факсов, псевдоним в IRC и адрес веб-страницы
- пароль и предпочтительная оболочка на машинах Проекта Debian

Естественно, большая часть информации не доступна публично. Для получения дополнительной информации, прочтите документацию, которая может быть найдена по адресу https://db.debian.org/doc-general.html.

Разработчики могут добавлять свои SSH-ключи, которые будут использовать для авторизации на официальных машинах Debian, и даже добавлять новые записи DNS вида *.debian.net. Эти возможности описаны по адресу https://db.debian.org/doc-mail.html.

4.6 Архив Debian

Дистрибутив Debian состоит из огромного числа пакетов (в настоящее время около 15000 пакетов с исходным кодом) и нескольких дополнительных файлов (таких как документация и образы установочных дисков).

Ниже приведён пример дерева каталогов полного архива Debian:

```
dists/stable/main/
dists/stable/main/binary-amd64/
dists/stable/main/binary-armel/
dists/stable/main/binary-i386/
     ...
dists/stable/main/source/
     ...
dists/stable/main/disks-amd64/
dists/stable/main/disks-armel/
dists/stable/main/disks-i386/
     ...

dists/stable/contrib/
dists/stable/contrib/binary-amd64/
dists/stable/contrib/binary-armel/
dists/stable/contrib/binary-i386/
     ...
dists/stable/contrib/source/

dists/stable/non-free/
dists/stable/non-free/binary-amd64/
dists/stable/non-free/binary-armel/
dists/stable/non-free/binary-i386/
     ...
dists/stable/non-free/source/

dists/testing/
dists/testing/main/
     ...
dists/testing/contrib/
     ...
dists/testing/non-free/
     ...

dists/unstable
dists/unstable/main/
     ...
dists/unstable/contrib/
     ...
dists/unstable/non-free/
     ...

pool/
pool/main/a/
```

```
pool/main/a/apt/
     ...
pool/main/b/
pool/main/b/bash/
     ...
pool/main/liba/
pool/main/liba/libalias-perl/
     ...
pool/main/m/
pool/main/m/mailx/
     ...
pool/non-free/f/
pool/non-free/f/firmware-nonfree/
     ...
```

Как вы можете видеть, каталог верхнего уровня содержит два каталога, `dists/` и `pool/`. Последний представляет собой “пул”, в котором фактически находятся пакеты, и который обрабатывается базой данных для сопровождения архива и сопутствующими программами. Первый же содержит выпуски, `стабильный`, `тестируемый` и `нестабильный`. Файлы `Packages` и `Sources` в подкаталогах выпусков могут указывать на файлы в каталоге `pool/`. Деревья каталогов каждого выпуска организованы одинаковым образом. То, что мы описываем ниже для `стабильного` выпуска применимо и для `нестабильного`, и для `тестируемого` выпусков.

`dists/stable` содержит три каталога, а именно `main`, `contrib` и `non-free`.

В каждом из них имеется каталог для пакетов с исходным кодом (`source`) и каталог для каждой поддерживаемой архитектуры (`binary-i386`, `binary-amd64` и т. д.).

Каталог `main` содержит дополнительные каталоги, в которых хранятся образы дисков и некоторые важные части документации, требующиеся для установки дистрибутива Debian на конкретную архитектуру (`disks-i386`, `disks-amd64` и т. д.).

4.6.1 Разделы

Раздел `main` архива Debian — то, что является **официальным дистрибутивом Debian**. Раздел `main` является официальным потому, что он полностью соответствует всем нашим рекомендациям. Другие два раздела не соответствуют им в той или иной степени; как таковые они **не** являются официальной частью Debian.

Каждый пакет из основного раздела должен полностью соответствовать Критериям Debian по определению Свободного ПО (DFSG), а также всем другим правилам, описываемым в Руководстве по политике Debian. DFSG представляет собой наше определение понятия “свободное ПО.” Для получения дополнительной информации ознакомьтесь с Руководством по политике Debian.

Пакеты из раздела `contrib` должны соответствовать DFSG, но могут не отвечать другим требованиями. Например, они могут зависеть от несвободных пакетов.

Пакеты, которые не соответствуют DFSG, помещаются в раздел `non-free`. Эти пакеты не считаются частью дистрибутива Debian, хотя мы даём возможность их использовать, а также предоставляем нашу инфраструктуру (например, нашу систему отслеживания ошибок, а также списки рассылки) под пакеты с несвободным ПО.

Руководство по политике Debian содержит более точное определение этих трёх разделов. Приведённое выше определение является лишь вводным.

Разделение этих трёх разделов на верхнем уровне архива является важным для всех людей, которые хотят распространять Debian, через FTP серверы в Интернет или на CD-ROM: распространяя только разделы `main` и `contrib`, можно избежать правовых рисков. Например, некоторые пакеты в разделе `non-free` не разрешают их коммерческое распространение.

С другой стороны, производитель CD-ROM легко может проверить лицензию отдельно взятого пакета в разделе `non-free` и добавить столько пакетов из этого раздела на CD-ROM, сколько это разрешено их лицензиями. (Поскольку это отличается от производителя к производителю, данная работа не может быть проделана разработчиками Debian.)

Заметьте, понятие раздела также используется для указания категорий, которые упрощают организацию и просмотр доступных пакетов, например, `admin`, `net`, `utils` и т. д. Когда-то давно эти разделы (скорее подразделы) существовали в виде подкаталогов внутри архива Debian. Сегодня же они существуют только в поле Section пакета.

4.6.2 Архитектуры

В начале ядро Linux было доступно только для платформ Intel i386 (или выше), поэтому и Debian был доступен только для этой платформы. Но когда Linux стал всё более и более популярным, ядро было перенесено на другие архитектуры, и Debian начал их поддерживать. И если этой поддержки оборудования было недостаточно, Debian решал собрать некоторые переносы на основе ядер Unix, например `hurd` и `kfreebsd`.

Debian GNU/Linux 1.3 был доступен только для `i386`. Debian 2.0 поставлялся для архитектур `i386` и `m68k`. Debian 2.1 поставлялся для архитектур `i386`, `m68k`, `alpha` и `sparc`. Затем Debian очень сильно вырос. Debian 6 поддерживает девять архитектур Linux (`amd64`, `armel`, `i386`, `ia64`, `mips`, `mipsel`, `powerpc`, `s390`, `sparc`) и две архитектуры kFreeBSD (`kfreebsd-i386` и `kfreebsd-amd64`).

Информация для разработчиков и пользователей о конкретных переносах доступна на веб-страницах переносов Debian.

4.6.3 Пакеты

Имеется два типа пакетов Debian, а именно пакеты `с исходным кодом` и `двоичные` пакеты.

В зависимости от формата пакета с исходным кодом, он состоит из одного или нескольких файлов в дополнение к обязательному файлу `.dsc`:

- формат “1.0” содержит либо файл `.tar.gz`, либо файл `.orig.tar.gz` и файл `.diff.gz`;
- формат “3.0 (quilt)” содержит обязательный tar-архив из основной ветки разработки `.orig.tar.{gz,bz2,xz}`, множество необязательных дополнительных tar-архивов из основной ветки разработки `.orig-`*`компонент`*`.tar.{gz,bz2,xz}` и обязательный tar-архив debian doc `debian.tar.{gz,bz2,xz}`;
- формат “3.0 (native)” содержит только один tar-архив `.tar.{gz,bz2,xz}`.

Если пакет разработан специально для Debian и не распространяется за пределами Debian, то в нём содержится только один файл `.tar.{gz,bz2,xz}`, содержащий исходный код программы, такой пакет называется “родным” пакетом с исходным кодом. Если пакет распространяется где-либо ещё, файл `.orig.tar.{gz,bz2,xz}` содержит так называемый `исходный код из основной ветки разработки`, то есть исходный код, распространяемый `сопровождающим основной ветки` (часто это автор ПО). В этом случае файл `.diff.gz` или `debian.tar.{gz,bz2,xz}` содержит изменения, сделанные сопровождающим Debian.

Файл `.dsc` содержит список всех файлов в пакете с исходным кодом, а также их контрольные суммы (**md5sums**, **sha1sums**, **sha256sums**) и некоторую дополнительную информацию о пакете (сопровождающий, версия и т. д.).

4.6.4 Выпуски

Система каталогов, описанная в предыдущей главе, сама по себе содержится в `каталогах выпусков`. Каждый выпуск фактически содержит каталог `pool` на верхнем уровне архива Debian.

Подводя итог, скажем, что корневой каталог архива Debian расположен на сервере FTP. Например, на зеркальном сайте, `ftp.us.debian.org`, сам архив Debian содержится в /debian, что является его обычным расположением (другое расположение — `/pub/debian`).

Выпуск включает в себя исходный код Debian и двоичные пакеты, а также соответствующие индексные файлы `Sources` и `Packages`, содержащие заголовочную информацию всех пакетов. Первый файл хранится в каталоге `pool/`, а последний — в каталоге `dists/` архива (для обратной совместимости).

4.6.4.1 Стабильный, тестируемые и нестабильный выпуски

Всегда имеются выпуски, называемые `стабильным` выпуск (он расположен в `dists/stable`), `тестируемый` выпуск (он расположен в `dists/testing`), а также `нестабильный` выпуск (он расположен в `dists/unstable`). Эта иерархия отражает процесс разработки Проекта Debian.

Активная разработка выполняется в `нестабильном` выпуске (вот почему этот выпуск иногда называется `разрабатываемым выпуском`). Каждый разработчик Debian может загружать свои пакеты в этот выпуск в любое время. Таким образом, содержимое этого выпуска меняется день ото дня. Поскольку для того, чтобы убедится, что этот выпуск работает нормально, иногда он в буквальном смысле бывает нестабильным.

Тестируемый выпуск создаётся автоматически путём принятия пакетов из нестабильного выпуска, если они удовлетворяют определённым критериям. Эти критерии должны гарантировать хорошее качестве пакетов в тестируемом выпуске. Обновление тестируемого выпуска запускается дважды в день, сразу после установки новых пакетов. См. раздел 5.13, «Тестируемый выпуск».

После периода разработки, когда управляющий выпуском решает, что время подходящее, тестируемый выпуск замораживается, что означает, что правила, управляющие тем, как пакеты переходят из нестабильного выпуска в тестируемый ужесточаются. Удаляются пакеты с большим количеством ошибок. Запрещается изменять что-либо в тестируемом выпуске за исключением исправления ошибок. Через некоторое время, в зависимости от хода всего процесса, тестируемый выпуск переходит на вторую стадию заморозки. Подробности работы с тестируемым выпуском публикуются выпускающей командой в списке рассылки debian-devel-announce. Когда открытые проблемы будут решены так, что выпускающая команда будет удовлетворена, осуществляется выпуск дистрибутива. Это предполагает, что тестируемый выпуск переименовывается в стабильный выпуск, создаётся его копия для нового тестируемого выпуска, а предыдущий стабильный переименовывается в предыдущий стабильный выпуск и остаётся им до тех пор пока наконец не будет архивирован. Во время архивирования его содержимое перемещается на archive.debian.org.

Этот цикл разработки основывается на допущении, что нестабильный выпуск становится стабильным после периода тестирования. Даже если выпуск считается стабильным, в нём всё равно присутствуют ошибки — вот почему стабильный выпуск продолжает обновляться. Тем не менее, эти обновления довольно тщательно тестируются до их включения в архив и добавляются по одному для того, чтобы снизить риск добавления новых ошибок. Вы можете найти предлагаемые дополнения к стабильному выпуску в каталоге proposed-updates. Те пакеты из proposed-updates, которые прошли проверку, периодически перемещаются в составе группы других пакетов в стабильный выпуск, а номер редакции стабильного выпуска увеличивается (напр., ‘6.0’ становится ‘6.0.1’, ‘5.0.7’ становится ‘5.0.8’ и т. д.). За подробностями обратитесь к разделу загрузки в стабильный выпуск.

Заметьте, что разработка нестабильного выпуска продолжается во время периода заморозки, поскольку нестабильный выпуск остаётся параллельным тестируемому выпуску.

4.6.4.2 Дополнительная информация о тестируемом выпуске

Обычно пакеты устанавливаются в тестируемый выпуск после того, как они пройдут некоторое тестирование в нестабильном выпуске.

Для получения дополнительных сведений обратитесь к информации о тестируемом выпуске.

4.6.4.3 Экспериментальный выпуск

Экспериментальный выпуск является специальным выпуском. Это не полный выпуск в том же смысле как являются полными стабильный, тестируемый и нестабильный выпуски. Наоборот, это лишь временное место для экспериментального ПО, которое вполне может сломать ваше систему, либо для ПО, которое пока ещё недостаточно стабильно для того, чтобы помещать его в нестабильный выпуск (но всё равно имеются причины для создания такого пакета). Ожидается, что пользователи, которые скачивают и устанавливают пакеты из экспериментального выпуска, предупреждены о возможных проблемах. Короче, для экспериментального выпуска мы не даём никаких гарантий, вся ответственность исключительно на вас.

Вот строки sources.list(5) для экспериментального выпуска:

```
deb http://ftp.xy.debian.org/debian/ experimental main
deb-src http://ftp.xy.debian.org/debian/ experimental main
```

Если имеется возможность того, что ПО может нанести непоправимый вред системе, лучше всего поместить его в экспериментальный выпуск. Например, поддержка экспериментальной файловой системы с сжатием должна, вероятно, войти в экспериментальный выпуск.

Если имеется новая версия пакета из основной ветки разработки, которая добавляет новые возможности, но приводит к поломке старых возможностей, она либо не должна быть загружена, либо должна быть загружена в экспериментальный выпуск. Новая бета версия некоторого ПО, использующая совершенно другую настройку, может войти в экспериментальный выпуск решению сопровождающего. Если вы работаете над несовместимой или сложной ситуацией по обновлению пакета, вы также можете использовать экспериментальный выпуск для тестирования, так чтобы те, кто будут тестировать ваш пакет, могли раньше получить к нему доступ.

Некоторое экспериментальное ПО может войти и в нестабильный выпуск, если в описание вы добавите предупреждение, но это не рекомендуется, поскольку пакеты предполагается, что пакеты из нестабильного выпуска будут перемещены в тестируемый выпуск, а затем в стабильный. Вам не следует бояться использовать экспериментальный выпуск, поскольку он не доставляет проблем сопровождающим ftp, экспериментальные пакеты периодически удаляются как только вы загрузите в нестабильный выпуск пакет, имеющий более высокий номер версии.

Новое ПО, которое скорее всего не повредит систему, может быть сразу добавлено в нестабильный выпуск.

Альтернативой экспериментальному выпуску является ваше личное пространство на people.debian.org.

4.6.5 Кодовые имена выпусков

Каждый выпущенный дистрибутив Debian имеет кодовое имя: Debian 6.0 называется squeeze; Debian 7 — wheezy; Debian 8 — jessie; следующий выпуск Debian 9 будет называться stretch а Debian 10 будет называться buster. Кроме того, имеется «псевдодистрибутив», называемый sid, который является текущим нестабильный выпуском; поскольку пакеты перемещаются из нестабильного выпуска в тестируемый по мере их достижения стабильного состояния, сам sid никогда не будет выпущен. Помимо обычного содержимого дистрибутива Debian sid содержит пакеты для тех архитектур, которые пока официально не поддерживаются и не были выпущены Debian. Эти архитектуры планируется интегрировать в будущем в основную ветку дистрибутива. Кодовые имена и версии предыдущих выпусков указаны на вебсайте.

Поскольку Debian следует открытой модели разработки (т. е., всякий может участвовать и следить за разработкой), даже нестабильный и тестируемый выпуски распространяются в Интернет через сеть FTP и HTTP серверов Debian. Таким образом, если мы назвали каталог, содержащий версию, рассматриваемую в качестве кандидата на выпуск, testing, то мы переименуем его в stable, когда эта версия будет выпущена, что приведёт к тому, что все FTP зеркала заново загрузят весь дистрибутив (который довольно велик).

С другой стороны, если мы с самого начала назовём каталоги выпусков Debian-x.y, люди будут полагать, что доступен выпуск Debian версии x.y. (Такое было в прошлом, когда производитель CD-ROM собрал CD-ROM Debian 1.0 на основе разрабатываемой версии pre-1.0. Вот почему первым официальным выпуском Debian был 1.1, а не 1.0.)

Таким образом, имена каталогов для выпусков в архиве соответствуют кодовым именам выпусков, а не статусу выпуска (напр., jessie'). Эти имена остаются одними и теми же в период разработки в после выпуска; символические ссылки, которые легко могут быть изменены, обозначают текущий стабильный выпуск. Вот почему фактические каталоги используют кодовые имена, а символические ссылки для стабильного, тестируемого и нестабильного выпусков указывают на соответствующие каталоги выпусков.

4.7 Зеркала Debian

Разные архивы для скачивания и веб-сайт имеют несколько зеркал для того, чтобы освободить наши каноничные серверы от тяжёлой нагрузки. Фактически, некоторые каноничные серверы не доступны публично — вместо этого первый ряд зеркал занимается балансировкой нагрузки. Так, пользователи всегда получают доступ к зеркалу и привыкают использовать их, что позволяет Debian лучше разделять пропускную способность между несколькими серверами и сетями и вообще позволяет пользователям избежать обращения к основному серверу. Заметьте, что первый ряд серверов является наиболее актуальными, поскольку они обновляются по запросу с внутренних сайтов (мы называем это проталкивающим зеркалированием).

Все информация о зеркалах Debian, включая список доступных публично FTP/HTTP серверов, может быть найдена на https://www.debian.org/mirror/. Эта полезная страница содержит информацию и инструменты, которые могут быть вам полезны, если вы заинтересованы в настройке собственного зеркала, как для внутренних нужд, так и с публичным доступом.

Заметьте, что зеркала обычно обслуживаются третьими лицами, которые заинтересованы в оказании помощи Debian. Как таковые, разработчики обычно не имеют учётных записей на этих машинах.

4.8 Система входящих пакетов

Система входящих пакетов ответственна за сбор обновлённых пакетов и их установку в архив Debian. Она состоит из набора каталогов и сценариев, которые установлены на ftp-master.debian.org.

Пакеты загружаются сопровождающими в каталог с названием `UploadQueue`. Этот каталог сканируется каждые несколько минут службой, названой **queued**, выполняются файлы `*.command`, а оставшиеся и корректно подписанные файлы `*.changes` перемещаются вместе с соответствующими их файлами в каталог `unchecked`. Этот каталог невидим для большинства разработчиков, поскольку доступ у главному ftp ограничен; он сканируется каждые 15 минут сценарием **dak process-upload**, который проверяет целостность загруженных пакетов и их криптографические подписи. Если пакет считается готовым к установке, он перемещается в каталог `done`. Если это первая загрузка данного пакета (либо в ней содержаться новые двоичные пакеты), пакет перемещается в каталог `new`, где он должен ожидать подтверждения от сопровождающих ftp. Если пакет содержит файлы, которые должны быть установлены вручную, он перемещается в каталог `byhand`, где он ожидает ручной установки, выполняемой сопровождающими ftp. В противном случае, если была обнаружена какая-либо ошибка, пакет получает отказ и перемещается в каталог `reject`.

Когда пакет будет принят, система отправляет сообщение с подтверждением этого сопровождающему и закрывает все ошибки, которые были отмечены как исправленные в данной загрузке, тогда ПО для автоматической сборки может начать их повторную компиляцию. Теперь пакет доступен публично в `http://incoming.debian.org/`, она остаётся там до фактической установки в архив Debian. Это происходит четыре раза в день (и также по историческим причинам называется dinstall run'); затем пакет удаляется из входящих и устанавливается в пул вместе со всеми другими пакетами. Когда все другие обновления (создающие, например, индексные файлы `Packages` и `Sources`) будут произведены, вызывается специальный сценарий, которые просит все первичные зеркала запустить обновление.

ПО для сопровождения архива также отправит подписанный с помощью OpenPGP/GnuPG файл `.changes`, который вы загрузили, в соответствующий список рассылки. Если у пакета поле `Distribution` имеет значение `stable`, то анонс отправляется в debian-changes@lists.debian.org. Если у пакета поле `Distribution` имеет значение `unstable` или `experimental`, то анонс будет отправлен в debian-devel-changes@lists.debian.org или debian-experimental-changes@lists.debian.org.

Хотя доступ к главному ftp ограничен, копия установки доступна всем разработчикам по адресу `mirror.ftp-master.deb`

4.9 Информация о пакете

4.9.1 В веб

Каждый пакет имеет несколько выделенных для него веб-страниц. `https://packages.debian.org/`*`имя-пакета`* отображает каждую версию пакета, доступную в различных выпусках. Каждая версия представляет собой ссылку на страницу, предоставляющую информацию, включая описание пакета, зависимости и ссылки для скачивания пакета.

Система отслеживания ошибок отслеживает ошибки каждого пакета. Вы можете просмотреть ошибки любого данного пакета по адресу `https://bugs.debian.org/`*`имя-пакета`*.

4.9.2 Утилита dak ls

dak ls является частью набора инструментов dak, она выводит список доступных версий пакета для всех известных выпусков и архитектур. Инструмент **dak** доступен на `ftp-master.debian.org`, а также на зеркале `mirror.ftp-master.debian.org`. Он используется единственный аргумент, соответствующий имени пакета. Пример объяснит это лучше:

```
$ dak ls evince
evince | 0.1.5-2sarge1 |     oldstable | source, alpha, arm, hppa, i386, ia64,  ↩
    m68k, mips, mipsel, powerpc, s390, sparc
evince |    0.4.0-5 |     etch-m68k | source, m68k
evince |    0.4.0-5 |        stable | source, alpha, amd64, arm, hppa, i386, ia64 ↩
    , mips, mipsel, powerpc, s390, sparc
evince |   2.20.2-1 |       testing | source
evince | 2.20.2-1+b1 |       testing | alpha, amd64, arm, armel, hppa, i386, ia64 ↩
    , mips, mipsel, powerpc, s390, sparc
evince |   2.22.2-1 |      unstable | source, alpha, amd64, arm, armel, hppa,  ↩
    i386, ia64, m68k, mips, mipsel, powerpc, s390, sparc
```

В этом примере вы можете видеть, что версия в `нестабильном` выпуске отличается от версии в `тестируемом` выпуске, и что была сделана binNMU этого пакета для всех архитектур. Каждая версия пакета была заново скомпилирована на всех архитектурах.

4.10 Система отслеживания пакетов Debian

Система отслеживания пакетов представляет собой построенный на основе электронной почты и веб инструмент для отслеживания активности пакета с исходным кодом. Вы можете получать ту же электронную почту, которую получает сопровождающий, просто подписавшись на пакет в системе отслеживания пакетов Debian.

Каждое сообщение электронной почты, отправляемое через систему отслеживания пакетов классифицируется по одному из ключевых слов, приведённых ниже. Это позволит вам выбирать сообщения, которые вы хотите получить.

По умолчанию вы будете получить следующие сообщения:

bts Все сообщения об ошибках и последующие их обсуждения.

bts-control Сообщения от control@bugs.debian.org об изменении статуса сообщения об ошибке.

upload-source Сообщения от **dak** о том, что загруженный пакет с исходным кодом был принят.

archive Другие сообщения о предупреждениях или ошибках от **dak** (например, отмена несоответствия раздела и/или приоритета). Также уведомления об удалении пакетов.

build Сообщения об ошибках сборки, отправляемые сетью сборочных служб, они содержат указатели на журналы сборки, которые следует изучить.

default Любые неопределённые сообщения электронной почты, отправляемые в систему отслеживания пакетов через `dispatch+`*`пакет_с_исходным_кодом`*`@tracker.debian.org`. Для того, чтобы предотвратить отправку спама, все сообщения, отправляющиеся по этим адресам, должны содержать заголовок `X-Distro-Tracker-Approved` с непустым значением.

contact Почта, отправляемая сопровождающему через псевдонимы *@packages.debian.org.

summary Регулярные обобщенные сообщения о состоянии пакета, включая переход в `тестируемый` выпуск. В идеале должны содержать уведомления о новых версиях основной ветки разработки, а также уведомления об осиротевших пакетах (но пока это не так).

Также вы можете получить дополнительную информацию:

upload-binary Уведомления от **dak**, отправляемые в случае принятия загруженного двоичного пакета. Другими словами, когда служба сборки или тот, кто занимается переносом, загружает ваш пакет для другой архитектуры, вы можете получить сообщений для отслеживания того, как ваш пакет компилируется для всех архитектур.

vcs Сообщения о коммитах в системе управления версиями, если пакет имеет репозиторий системы управления версиями и сопровождающий настроил пересылку сообщений о коммитах в систему отслеживания пакетов.

translation Переводы описаний или шаблонов debconf, отправленные в Проект по переводу описаний Debian.

derivatives Информацию об изменения, внесённых в пакет, в производных дистрибутивах (например, Ubuntu).

derivatives-bugs Сообщения об ошибках и комментарии из производных дистрибутивов (например, Ubuntu).

4.10.1 Интерфейс электронной почты для системы отслеживания пакетов

Вы можете управлять своими подписками на систему отслеживания пакетов, отправляя различные команды по адресу control@tracker.debian.org.

subscribe <пакет-с-исходным-кодом> [<адрес-электронной-почты>] Подписывает *адрес-электрон* на сообщения, связанные с пакетом с исходным кодом *пакет-с-исходным-кодом*. Адрес отправителя используется в том случае, если второй аргумент отсутствует. Если *пакет-с-исходным-кодом* не является корректным пакетом с исходным кодом, вы получите предупреждение. Тем не менее, если это корректный двоичный пакет, систем отслеживания пакетов подпишет вас на соответствующий пакет с исходным кодом.

unsubscribe <пакет-с-исходным-кодом> [<адрес-электронной-почты>] Удаляет предыдущую подписку на пакет с исходным кодом *пакет-с-исходным-кодом*, используя указанный адрес электронной почты, либо адрес отправителя в случае, если второй аргумент опущен.

unsubscribeall [<адрес-электронной-почты>] Удаляет все подписки указанного адреса электронной почты, либо адреса отравителя, в случае если второй аргумент опущен.

which [<адрес-электронной-почты>] Выводит список всех подписок отправителя, либо указанного адреса электронной почты.

keyword [<адрес-электронной-почты>] Сообщает вам то, какие ключевые слова вы принимаете. Объяснение ключевых слов см. выше.

keyword <пакет-с-исходным-кодом> [<адрес-электронной-почты>] То же, что и предыдущий пункт, но для данного пакета с исходным кодом, поскольку вы можете выбрать другой набор ключевых слов для каждого пакета с исходным кодом.

keyword [<адрес-электронной-почты>] {+|-|=} <список-ключевых-слов> Принимает (+) или отклоняет (-) сообщения, классифицируемые по данным ключевым словам. Определяет список (=) принимаемых ключевых слов. Изменяет набор ключевых слов, по умолчанию принимаемых пользователем.

keywordall [<адрес-электронной-почты>] {+|-|=} <список-ключевых-слов> Принимает (+) или отклоняет (-) сообщения, классифицируемые по данным ключевым словам. Определяет список (=) принимаемых ключевых слов. Изменяет набор принимаемых ключевых слов всех активных в настоящий момент подписок пользователя.

keyword <пакет-с-исходным-кодом> [<адрес-электронной-почты>] {+|-|=} <список-ключевых-слов> То же, что и предыдущий пункт, но отменяет список ключевых слов для обозначенного пакета с исходным кодом.

quit | thanks | -- Останавливает обработку команд. Все последующие строки будут игнорироваться роботом.

Утилита командной строки **pts-subscribe** (из пакета `devscripts`) может быть полезна для оформления временной подписки на какие-то пакеты, например, после осуществления NMU.

4.10.2 Фильтрация почты от системы отслеживания пакетов

После оформления подписки на пакет вы будете получить почту, пересылаемую системой отслеживания пакетов. Эти сообщения имеют специальные заголовки для того, чтобы вы могли отфильтровать их в специальный почтовый ящик (напр. , с помощью **procmail**). Добавлены следующие заголовки: `X-Loop`, `X-Distro-Tracker-Package`, `X-Distro-Tracker-Keyword`, `X-Debian-Package`, `X-Debian`, `List-Id` и `List-Unsubscribe`.

Ниже приведён пример добавленных заголовков для оповещения о загрузке исходного кода для пакета `dpkg`:

```
X-Loop: dispatch@tracker.debian.org
X-Distro-Tracker-Package: dpkg
X-Distro-Tracker-Keyword: upload-source
X-Debian-Package: dpkg
X-Debian: tracker.debian.org
List-Id: <dpkg.tracker.debian.org>
List-Unsubscribe: <mailto:control@tracker.debian.org?body=unsubscribe%20dpkg>
```

4.10.3 Пересылка коммитов системы управления версиями в систему отслеживания пакетов

Если вы используете публично доступный репозиторий системы управления версиями для сопровождения вашего пакета Debian, вы возможно пересылать оповещения о коммитах в систему отслеживания пакетов, чтобы подписчики (и возможные помощники) могли более внимательно следить за эволюцией пакета.

Когда вы настроите свои репозиторий для отправки сообщений о коммитах, вам следует убедиться, что копия сообщения отправляется в систему отслеживания пакетов по адресам `dispatch@tracker.debian.org`

или dispatch+*пакет_с_исходным_кодом*_vcs@tracker.debian.org. В первом случае вы должны убедиться, что система отслеживания пакетов может определить пакет с исходным кодом и соответствующее ключевое слово... либо добавляя заголовки X-Distro-Tracker-Package: *пакет_с_исходным_кодом* и X-Distro-Tracker-Keyword: vcs, либо полагаясь на тот факт, что система отслеживания пакетов обнаружит заголовок X-Git-Repo и будет считать, что имя git-репозитория совпадает с именем пакета с исходным кодом.

При использовании репозиториев Subversion рекомендуется использовать svnmailer. Пример того, как это делать, см. в https://wiki.debian.org/Alioth/PackagingProject.

4.10.4 Веб-интерфейс системы отслеживания пакетов

Система отслеживания пакетов имеет веб-интерфейс по адресу https://tracker.debian.org/, в нём собирается множество информации о каждом пакете с исходным кодом. Там имеется множество полезных ссылок (система отслеживания ошибок, статистика QA, контактная информация, статус перевода DDTP, журналы сборки) и собирается большое количество информации из разные мест (30 последних записей журнала изменений, статус в тестируемом выпуске и т. д.). Это очень полезный инструмент, если вы хотите знать, что происходит с конкретным пакетом с исходным кодом. Более того, после аутентификации вы сможете за один клик подписаться или отписаться от информации о любом пакете.

Вы можете напрямую перейти на веб-страницу конкретного пакета с исходным кодом при помощи URL вида https://tracker.debian.org/*пакет-с-исходным-кодом*.

Веб-интерфейс легко расширять, вы можете интегрировать в него дополнительные полезные данные. Если вы хотите принять участие в разработке, обратитесь к https://tracker.debian.org/docs/contributing.html.

4.11 Обзор пакетов разработчика

Веб-портал QA (гарантия качества) доступен по адресу https://qa.debian.org/developer.php, на нём отображается таблица с пакетами одного разработчика (включая те пакеты, у которых в качестве помощников указана группа). Таблица даёт обзор пакетов конкретного разработчика: число ошибок по их важности, список доступных версий в каждом выпуске, статус в тестируемом выпуске и множество ссылок на другую полезную информацию.

Рекомендуется регулярно просматривать эти данные для своих пакетов, так вы не забудете об открытых сообщениях об ошибках и не забудете то, за какие пакеты вы ответственны.

4.12 Установка Debian FusionForge: Alioth

Alioth является службой Debian, основанной на слегка изменённой версии ПО FusionForge (которое эволюционировало из SourceForge и GForge). Это ПО предлагает разработчикам доступ к простым в использовании инструментам, таким как системы отслеживания ошибок, ПО для управления заплатами, ПО для управления проектами/задачами, службы хостинга файлов, списки рассылки, репозитории систем управления версиями и т. д. Управление всеми этими инструментами осуществляется через веб-интерфейс.

Это ПО предназначено для того, чтобы предоставить свободным проектам, которые поддерживаются или ведутся Debian, облегчения участия внешних разработчиков в проектах, начатых Debian, а также для помощи проектам, чьими целями являются продвижение Debian или производных от него дистрибутивов. Оно активно используется многими командами Debian и предоставляет хостинг для всех видов репозиториев систем управления версиями.

Все разработчики Debian автоматически получают учётную запись на Alioth. Они могут активировать ее, используя инструмент восстановления пароля. Внешние разработчики могут запросить гостевые учётные записи на Alioth.

Для получения дополнительной информации перейдите по следующим ссылкам:

- https://wiki.debian.org/Alioth
- https://wiki.debian.org/Alioth/FAQ
- https://wiki.debian.org/Alioth/PackagingProject
- https://alioth.debian.org/

4.13 Плюшки для разработчиков и сопровождающих Debian

О плюшках, доступных разработчикам и сопровождающим Debian можно прочитать тут: https://wiki.debian.org/MemberBenefits.

Глава 5

Управление пакетами

Данная глава содержит информацию, связанную с созданием, загрузкой, сопровождением и переносом пакетов.

5.1 Новые пакеты

Если вы хотите создать новый пакет для дистрибутива Debian, вам следует вначале проверить список требующих доработки и будущих пакетов (WNPP). Проверка списка WNPP гарантирует, что в настоящий момент никто не работает над созданием пакетов для данного ПО, и что не будет проделана повторная работа. Прочтите страницу WNPP для получения дополнительной информации.

Допустим, что более никто не работает над вашим будущим пакетом. Далее, вам следует отправить сообщение об ошибке (раздел 7.1, «Отправка отчётов об ошибках») в псевдопакете `wnpp` с объяснением вашего плана по созданию нового пакета, ваше сообщение должно в себя включать описание пакета, лицензию будущего пакета и текущий адрес, по которому этот пакет может быть загружен.

Тема сообщения об ошибке должна иметь вид `ITP: foo -- краткое описание`, подставьте имя нового пакета вместо *foo*. Важность сообщения об ошибке должна быть установлена в значение `wishlist`. Отправьте копию по адресу debian-devel@lists.debian.org, используя заголовок X-Debbugs-CC (не используйте CC:, так как в этом случае в теме сообщения не будет указан номер ошибки). Если вы создаёте много новых пакетов (>10), то помните, что отправка в список рассылки большого количества отдельных сообщений будет слишком мешать другим, отправьте сообщение с обзором вашей работы в список рассылки debian-devel после заполнения всех сообщений об ошибках. Это позволит сообщить вам другим разработчикам о готовящихся пакетах и позволит вам проверить ваши описания и имена пакетов.

Добавьте запись вида `Closes: #`*nnnnn* в журнал изменений нового пакета для того, чтобы ваше сообщение об ошибке было автоматически закрыто сразу же как только пакет будет установлен в архиве (см. раздел 5.8.4, «Когда ошибки исправляются путём новых загрузок»).

Если вы считаете, что по поводу вашего пакета необходимо дать дополнительные объяснения администраторам очереди новых (NEW) пакетов, добавьте их в журнал изменений, отправьте по адресу ftpmaster@debian.org ваш ответ на сообщение электронной почты, которое вы получите как сопровождающий после вашей загрузки пакета, либо ответьте на сообщение об отказе, если вы осуществляете загрузку повторно.

Закрывая сообщения об ошибках безопасности, добавляйте номера CVE, а также `Closes: #`*nnnnn*. Это помогает команде безопасности отслеживать уязвимости. Если загрузка осуществляется для того, чтобы исправить ошибку, и если идентификационный номер рекомендации по безопасности ещё не известен, советуем вам во время следующей загрузки изменить запись в журнале изменений и добавить номер рекомендации. Даже в этом случае добавляйте все доступные указатели на общую информацию из записи оригинального журнала изменений.

Имеется ряд причин, почему мы просим сопровождающих анонсировать их намерения, они приведены ниже:

- Это помогает (потенциальному) сопровождающему получить советы опытных людей, участвующих в списке рассылки, и позволяет этим людям узнать, работает ли уже кто-нибудь над созданием пакета для данного ПО.

- Это позволяет другим людям, которые думают о работе над созданием пакета, знать, что кто-то уже начал этим заниматься, и поэтому можно объединить усилия.

- Это позволяет остальным сопровождающим знать о данном пакете больше, чем то, что содержится в одной строке описания и обычной записи из журнала изменений вида „Initial release“, которые публикуются в debian-devel-changes@lists.debian.org.
- Это помогает людям, которые постоянно пользуются `нестабильным` выпуском (и являются теми, кто первый широко тестирует пакеты). Нам следует поощрять этих людей.
- Анонсы дают сопровождающим и другим заинтересованным сторонам лучшее понимание того, что происходит и что является новым в Проекте.

Наиболее частые причины для отказа в добавлении нового пакета см. по адресу `https://ftp-master.debian.org/REJECT-FAQ.html`.

5.2 Запись изменений в пакете

Изменения, которые вы произвели в пакете, должны быть записаны в файл `debian/changelog`. Эти изменения должны содержать точное описание того, что было изменено, почему это было изменено (в случае если имеются какие-либо сомнения), а также номера закрываемых сообщений об ошибках. Кроме того, в журнале должно быть указано, когда была завершена работа над пакетом. Этот файл будет установлен как `/usr/share/doc/`*`пакет`*`/changelog.Debian.gz`, либо как `/usr/share/doc/`*`пакет`*`/changelog.gz` в случае, если пакет является родным.

Файл `debian/changelog` соответствует определённой структуре, имеющей ряд различных полей. Одно из этих полей, `выпуск`, описывается в разделе 5.5, «Выбор выпуска». Дополнительная информация о структуре этого файла может быть найдена в Политике Debian, в разделе с названием `debian/changelog`.

Записи журнала изменений могут использоваться для автоматического закрытия сообщений об ошибках Debian, когда пакет устанавливается в архив. См. раздел 5.8.4, «Когда ошибки исправляются путём новых загрузок».

Обычная запись в журнале изменений какого-либо пакета, содержащего новую версию из основной ветки разработки ПО, выглядит следующим образом:

```
  * New upstream release.
```

Имеются специальные инструменты, которые помогают вам создавать записи журнала изменений и готовить файл `changelog` к выпуску — см. раздел A.6.1, «`devscripts`» и раздел A.6.5, «`dpkg-dev-el`».

Также см. раздел 6.3, «Лучшие практики для `debian/changelog`».

5.3 Тестирование пакета

До того как вы загрузите ваш пакет, вам следует провести его простое тестирование. Как минимум вам следует попытаться выполнить следующие действия (вам будет нужна более старая версия того же пакета Debian):

- Установите пакет и убедитесь, что ПО работает, либо обновите пакет с более старой версии до вашей новой версии, если пакет Debian для данного ПО уже существует.
- Выполните **lintian**, в качестве цели укажите ваш пакет. Вы можете запустить **lintian** следующим образом: `lintian -v `*`версия-пакета`*`.changes`. Эта команда проверит пакет с исходным кодом, наряду с двоичным пакетом. Если вы не понимаете вывод, предоставляемый вам **lintian**, попробуйте добавить опцию `-i`, это приведёт к тому, что **lintian** будет выводить довольно подробное описание проблему.

  Обычно пакет *не* следует загружать в случае, если **lintian** сообщает об ошибках (они начинаются с `E`).

  Для получения дополнительной информации о команде **lintian**, см. раздел A.2.1, «`lintian`».
- Также вы можете выполнить **debdiff** (см. раздел A.2.2, «**debdiff**»), чтобы проанализировать изменения по сравнению с более старой версией (если таковая существует).
- Установите предыдущую версию пакета (если она существует) поверх новой — это позволит проверить работу сценариев `postrm` и `prerm`.
- Удалите пакет, затем заново установите его.

- Скопируйте пакет с исходным кодом в другой каталог и попытайтесь распаковать и собрать его заново. Это позволит проверить то, зависит ли пакет от существующих файлов, которые не были добавлены в сам пакет, а также проверить то, зависит ли он от того, сохраняются ли права доступа в файлах из файла `.diff.gz`.

5.4 Схема пакета с исходным кодом

Имеется два типа пакетов Debian с исходным кодом:

- так называемые `родные` пакеты, для которых нет различия между оригинальным исходным кодом и заплатами Debian
- (более частные) пакеты, для которых имеется оригинальный исходный файл с tarball-архивом и другой файл, содержащий изменения, внесённые Проектом Debian

У родных пакетов пакеты с исходным кодом включают в себя исходный файл контроля (`.dsc`) и исходный архив в виде tarball (`.tar.{gz,bz2,xz}`). Пакет с исходным кодом неродного пакета включает в себя исходный файл контроля, оригинальный исходный файл в виде tarball (`.orig.tar.{gz,bz2,xz}`) и специфичные изменения Debian (`.diff.gz` для формата пакета с исходным кодом “1.0” или `.debian.tar.{gz,bz2,xz}` для формата пакета с исходным кодом “3.0 (quilt)”).

Для пакетов в исходном формате “1.0” то, является пакет родным или нет, определялось командой **dpkg-source** во время сборки. Сегодня же рекомендуется явным образом указывать желаемый исходный формат, помещая строку “3.0 (quilt)”, либо “3.0 (native)” в файл `debian/source/format`. Остаток настоящего раздела посвящён исключительно неродным пакетам.

Во время первой загрузки пакета, версия пакета должна соответствовать определённой версии основной ветки разработки, должен быть загружен оригинальный исходный tar-файл, также он должен быть добавлен в файле `.changes`. В последующем этот самый tar-файл должен использоваться для сборки новых различий (diffs) и файлов `.dsc`, его не нужно будет загружать заново.

По умолчанию команды **dpkg-genchanges** и **dpkg-buildpackage** включат оригинальный исходный tar-файл тогда и только тогда, когда текущая запись в журнале изменений содержит версию из основной ветки разработки, которая отличается от предыдущей записи. Это поведение может быть изменено при помощи использования `-sa`, что позволяет всегда включать оригинальный исходный tar-файл, либо `-sd`, что позволяет всегда игнорировать его.

Если оригинальный исходный код не включён в загрузку, оригинальный tar-файлс исходным кодом, используемый **dpkg-source** при создании файла `.dsc` и diff для загрузки *должен* побайтно совпадать с тем файлом, который уже добавлен в архив.

Заметьте, что в неродных пакетах права доступа к файлам, которые не входят в `*.orig.tar.{gz,bz2,xz}` не будут сохранены, поскольку diff не сохраняет права доступа к файлам в заплате. Тем не менее, если используется формат исходного кода “3.0 (quilt)”, права доступа к файлам внутри каталог `debian` сохраняются, так как они сохранены в архиве tar.

5.5 Выбор выпуска

Каждая загрузка должна содержать явное указание того, для какого выпуска предназначается данный пакет. Процесс сборки пакета извлекает эту информацию из первой строки файла `debian/changelog` и помещает её в поле `Distribution` файла `.changes`.

Обычно пакеты загружаются в `нестабильный` выпуск. Загрузки в `нестабильный` (`unstable`) или `экспериментальный` (`experimental`) выпуски должны использовать эти названия выпуска в записях журнала изменений; uploads for other supported suites should use the suite codenames, as they avoid any ambiguity.

Фактически, существуют и другие выпуски: *`кодовое-имя`*`-security`, прочтите раздел 5.8.5, «Работа с ошибками, связанными с безопасностью» для получения дополнительной информации.

Нельзя загрузить пакет в несколько выпусков одновременно.

5.5.1 Специальный случай: загрузка в `стабильный` и `предыдущий стабильный` выпуски

Загрузка в `stable` означает, что пакет будет передан в очередь `proposed-updates-new` для проверки управляющими стабильного выпуска, и если пакеты будут одобрены, то они будут установлены в ка-

талог stable-proposed-updates архива Debian. Отсюда, в свою очередь, пакеты будут перенесены в stable во время формирования следующей редакции выпуска.

To ensure that your upload will be accepted, you should discuss the changes with the stable release team before you upload. For that, file a bug against the release.debian.org pseudo-package using **reportbug**, including the patch you want to apply to the package version currently in stable. The patch should be a source **debdiff** (see Раздел A.2.2) against the current version in stable. The changelog entry should be against the stable distribution (e.g. jessie') and should be detailed, including Closes statements for bugs that are fixed by the upload.

Необходимо с особым вниманием относиться к загрузке пакета в стабильный выпуск. По сути, пакет следует загружать в стабильный выпуск исключительно в случае, если произошло что-либо из ниже следующего:

- действительно критическая проблема функциональности
- пакет больше нельзя установить
- пакет отсутствует в выпущенной архитектуре

В прошлом загрузки в stable использовались и для решения проблема безопасности. Тем не менее, в настоящее время эта практика считается устаревшей, поскольку загрузки, используемые для рекомендаций Debian по безопасности, автоматически копируются в соответствующий архив proposed-updates в момент выпуска рекомендации. См. раздел 5.8.5, «Работа с ошибками, связанными с безопасностью» для получения подробной информации о работе с проблемами безопасности. Если команда безопасности считает, что проблема не настолько велика, чтобы исправлять её через механизм DSA, обычно управляющие стабильным выпуском добавляют ваше исправление в обычную загрузку в стабильный выпуск.

Изменение в пакете чего-либо ещё, что не является действительно важным, крайне не рекомендуется, поскольку даже тривиальные исправления могут вызвать появление новых ошибок.

Пакеты, загружаемые в стабильный выпуск, должны быть скомпилированы на системах, работающих под управлением стабильного выпуска, чтобы их зависимости были ограничены библиотеками (а также другими пакетами), доступными в стабильном выпуске; например, пакет, загруженный в стабильный выпуск, будет отклонён в том случае, если она зависит от пакета с библиотекой, который доступен только в нестабильном выпуске. Крайне не рекомендуется производить изменения зависимостей других пакетов (путём создания путаницы с Provides или файлами shlibs), которые могут привести к тому, что эти другие пакеты нельзя будет установить.

Загрузки в предыдущие стабильные выпуски возможны до тех пор, пока эти выпуски не будут добавлены в архив. Те же правила применяются и к стабильным выпускам.

5.5.2 Специальный случай: загрузка в testing/testing-proposed-updates

Подробности см. в разделе тестируемом выпуске.

5.6 Загрузка пакета

5.6.1 Загрузка на ftp-master

Чтобы загрузить пакет, вам следует загрузить файлы (включая подписанные файлы changes и dsc-файл) с помощью анонимного доступа по ftp на ftp.upload.debian.org в каталог /pub/UploadQueue/. Для того, чтобы файлы были обработаны, они должны быть подписаны ключом, входящим либо в брелок ключей разработчиков Debian, либо в брелок ключей сопровождающих Debian (см. https://wiki.debian.org/DebianMaintainer).

Заметьте, что вам следует переслать файл changes самым последним. В противном случае ваша загрузка может быть отклонены потому, что ПО для сопровождения архива, осуществляющее грамматический разбор файла changes, посчитает, что не все файлы были загружены.

Для загрузки пакетов Вам могут пригодиться пакеты dupload или dput.Эти удобные программы помогают автоматизировать процесс загрузки пакетов в Debian.

Для удаления пакетов см. ftp://ftp.upload.debian.org/pub/UploadQueue/README и пакет dcut.

5.6.2 Задержанные загрузки

Иногда полезно загрузить пакет сразу же, но так, чтобы этот пакет поступил в архив лишь несколько дней спустя. Например, при подготовке загрузки тем, кто не является сопровождающим, вы возможно захотите дать сопровождающему несколько дней для того, чтобы тот как-то отреагировал.

Загрузка в каталог задержанных пакетов приводит к сохранению пакета в очереди отложенной загрузки. Когда указанное время ожидания истечёт, пакет будет перемещён в обычный каталог входящих пакетов для его обработки. Это производится путём автоматической загрузки на `ftp.upload.debian.org` в каталог загрузки `DELAYED/`*`X`*`-day` (*`X`* может быть в интервале от 0 до 15). 0-дней загружается несколько раз в день на `ftp.upload.debian.org`.

Работая с dput, вы можете использовать параметр `--delayed` *`ЗАДЕРЖКА`*, чтобы поместить пакет в одну из очередей.

5.6.3 Загрузки безопасности

НЕ загружайте пакет в очередь загрузки безопасности (на `security-master.debian.org`) без предварительного разрешения от команды безопасности. Если пакет не соответствует требованиям команды безопасности, он может вызвать множество проблем и задержек во время рассмотрения этой нежелательной загрузки. Подробности см. в разделе 5.8.5, «Работа с ошибками, связанными с безопасностью».

5.6.4 Другие очереди загрузки

В Европе имеется альтернативная очередь загрузки по адресу `ftp://ftp.eu.upload.debian.org/pub/UploadQueue/`. Она работает точно так же как и `ftp.upload.debian.org`, но для разработчиков из Европы она может быть более удобной из-за более быстрого доступа.

Кроме того, пакеты можно загружать через ssh на `ssh.upload.debian.org`; файлы должны быть помещены в `/srv/upload.debian.org/UploadQueue`. Эта очередь не поддерживает отложенные загрузки.

5.6.5 Notifications

Сопровождающие архива Debian ответственны за обработку загрузок пакетов. По большей части загрузки обрабатываются автоматически и ежедневно при помощи специальных инструментов для сопровождения архива, **dak process-upload**. Говоря более конкретно, обновление существующих пакетов, предназначенные для `нестабильного` выпуска, обрабатываются автоматически. В других случаях (особенно это касается новых пакетов) помещение загруженного пакета в выпуск осуществляется вручную. Если загрузки обрабатываются вручную, изменение архива может потребовать некоторого времени. Пожалуйста, будьте терпеливы.

В любом случае вы получите уведомление по электронной почте о том, что пакет был добавлен в архив, в этом уведомлении также указаны сообщения об ошибках, которые будут закрыты благодаря данной загрузке. Внимательно изучите это уведомление, проверьте, все ли сообщения об ошибках, которые должны быть закрыты, в нём указаны.

Уведомление об установке также включает в себя информацию о том, в какой раздел был добавлен ваш пакет. Если имеет место несоответствие, то вы получите об этом отдельное сообщение. Читайте ниже.

Заметьте, что если вы загрузили пакет через очереди, служба очередей также отправит вам уведомление по электронной почте.

Also note that new uploads are announced on the IRC channel `#debian-devel-changes`. If your upload fails silently, it could be that your package is improperly signed, in which case you can find more explanations on `ssh.debian.org:/srv/upload.debian.org/queued/run/log`.

5.7 Определение раздела для пакета, подраздела и приоритета

Поля `Section` и `Priority` файла `debian/control` фактически не определяют то, куда в архиве будет помещён ваш пакет, также они не определяют приоритет. Для того, чтобы сохранить общую целостность архива, сопровождающие архива осуществляют контроль над данными полями. Значения в файле `debian/control` в действительности являются лишь подсказками.

Сопровождающие архива отслеживаю канонические разделы и приоритеты пакетов с помощью специального `файла замещения`. Если между `файлом замещения` и полями пакета, определёнными в файле `debian/control`, имеет место несоответствие, то вы получите сообщение о расхождении в момент, когда

пакет будет установлен в архив. Вы можете либо исправить ваш файл `debian/control` во время следующей загрузки, либо вы можете захотеть изменить `файл замещения`.

Чтобы изменить раздел, в который был помещён пакет, вам для начала нужно убедиться, что файл `debian/control` в вашем пакете правилен. Далее, отправьте сообщение об ошибке в `ftp.debian.org` с запросом изменения раздела или приоритета для вашего пакета. Используйте тему сообщения на подобие `override: ПАКЕТ1:раздел/приоритет, [...], ПАКЕТX:раздел/приоритет` и добавьте обоснование данного изменения в теле сообщения об ошибке.

Дополнительную информацию о `файлах замещения` см. в dpkg-scanpackages(1) и https://www.debian.org/Bugs/Developer#maintincorrect.

Заметьте, что поле `Section` описывает и раздел, и подраздел, которые описываются в разделе 4.6.1, «Разделы». Если раздел имеет значение main (основной), то его указание должно быть опущено. Список разрешённых подразделов может быть найден в https://www.debian.org/doc/debian-policy/ch-archive.html#s-subsections.

5.8 Работа с ошибками

Каждый разработчик должен быть способен работать с системой отслеживания ошибок Debian. Это предполагает знание того, как следует правильно отправлять сообщения об ошибках (см. раздел 7.1, «Отправка отчётов об ошибках»), как обновлять и упорядочивать их, а также то, как с ними работать и как их закрывать.

Возможности системы отслеживания ошибок описаны в документации по системе отслеживания ошибок для разработчиков. Она включает в себя описание того, как закрывать сообщения об ошибках, отправлять ответные сообщения, назначать важность и теги, отмечать ошибки как перенаправленные, а также многих других вопросов.

Такие операции как переназначение сообщений об ошибках другим пакетам, объединение отдельных сообщений об ошибках в одну проблему, а также повторное открытие ошибок, если они были закрыты прежде времени, осуществляются при помощи так называемого сервера управляющей почты. Все доступные на этом сервере команды описываются в документации по управляющему серверу системы отслеживания ошибок.

5.8.1 Мониторинг ошибок

Если вы хотите быть хорошим сопровождающим, вам следует периодически проверять в системе отслеживания ошибок Debian (BTS) состояние ваших пакетов. Система отслеживания ошибок содержит все открытые сообщения об ошибках в ваших пакетах. Вы можете проверить их путём просмотра следующей страницы: `https://bugs.debian.org/`*`ваша-учётная-запись`*`@debian.org`.

Сопровождающие взаимодействуют с системой отслеживания ошибок через адреса электронной почты `bugs.debian.org`. Документация по доступным командам может быть найдена в https://www.debian.org/Bugs/, либо, если вы установили пакет `doc-debian`, вы можете посмотреть локальные файлы `/usr/share/doc/debian/bug-*`.

Некоторые разработчики находят полезным получение периодических отчётов об открытых ошибках. Вы можете добавить работу cron подобно следующей, если вы хотите получить еженедельные сообщения с обзором всех открытых сообщений об ошибках в ваших пакетах:

```
# запрашивает еженедельные отчёты об ошибках в моих пакетах
0 17 * * fri   echo "index maint address" | mail request@bugs.debian.org
```

Замените *`адрес`* вашим официальным адресом сопровождающего Debian.

5.8.2 Ответ на ошибки

Отвечая на сообщения об ошибках, убедитесь, что любое обсуждение сообщения об ошибке отправляется и изначальному автору сообщения об ошибке, и записи в системе отслеживания ошибок (напр., *`123`*@bugs.debian.org). Если вы пишете новое сообщение и не помните адрес электронной почты изначального автора сообщения об ошибке, вы можете использовать адрес *`123`*-submitter@bugs.debian.org для того, чтобы связаться с ним *и* записать вашу почту в журнал сообщения об ошибке (что означает, что вам не нужно будет отправлять копию письма на адрес *`123`*@bugs.debian.org).

Если вы получили сообщение об ошибке, в котором упоминается FTBFS, то это означает Fails to build from source (не удалось собрать из исходного кода). Те, кто занимаются переносами, часто используют данный акроним.

Как только вы закончили работать с сообщением об ошибке (напр., исправили ошибку) отметьте его как `завершённое` (закройте его), отправив сообщение с объяснением по адресу *123*-done@bugs.debian.org. Если вы исправили ошибку путём изменения и загрузки пакета, вы можете автоматизировать закрытие сообщения об ошибке как это описано в разделе 5.8.4, «Когда ошибки исправляются путём новых загрузок».

Вам *никогда* не следует закрывать ошибки через отправку команды `close` для сервера ошибок по адресу control@bugs.debian.org. Если вы так сделаете, то изначальный автор сообщения об ошибке не получит какой-либо информации о том, почему сообщение об ошибке было закрыта.

5.8.3 Работа с ошибками

Как сопровождающий пакетов вы часто будете находить ошибки в других пакетах, иногда вы будете получать сообщения об ошибках в ваших пакетах, которые в действительности касаются ошибок в других пакетах. Возможности системы отслеживания ошибок описываются в документации по системе отслеживания ошибок для разработчиков Debian. Такие операции как переназначение, объединение и отметка тегами сообщений об ошибках описываются в документации по управляющему серверу системы отслеживания ошибок. Данный раздел содержит некоторые рекомендации по управлению сообщениями об ошибках в ваших собственных пакетах, эти рекомендации основываются на коллективном опыте разработчиков Debian.

Отправка сообщений об ошибках по поводу проблем, которые вы найдёте в других пакетах, является одной из гражданских обязанностей сопровождающего, см. раздел 7.1, «Отправка отчётов об ошибках» для получения дополнительной информации. Тем не менее, работа с ошибками в ваших собственных пакетах ещё более важна.

Ниже приведена последовательность шагов, которой можно следовать при работе с сообщением об ошибке:

1. Решите, соответствует присланный отчёт реальной ошибке или нет. Иногда пользователи всего лишь неправильно запускают программу, поскольку они не прочли документацию. Если вы это обнаружите, то просто закройте сообщение об ошибке, предоставив достаточное количество информации, которая позволит пользователю исправить свою проблему (вышлите ссылки на хорошую документацию и так далее). Если вы снова получите то же сообщение об ошибке, задайте себе вопрос, достаточно ли хороша документация, или может быть программа не может обнаружить случаи неправильного использования и выдать информативное сообщение об ошибке. Об этой проблеме следует сообщить автору основной ветки разработки.

   Если тот, кто отправил сообщение об ошибке, не согласен с вашим решением о закрытии сообщения об ошибке, то он может заново открывать его до тех пор, пока вы не придёте к согласию о том, что делать. Если вы не сможете найти общего решения, тогда вы можете отметить ошибку тегом `wontfix`, чтобы люди знали, что ошибка существует, но что она не будет исправлена. Если эта ситуация не приемлема, вы (или тот, кто отправил сообщение об ошибке) можете потребовать решение проблемы от технического комитета, переназначив сообщение об ошибке `tech-ctte` (вы можете использовать команду clone, если вы хотите, чтобы сообщение об ошибке осталось сообщением об ошибке в вашем пакете). До того, как сделать это, прочтите информацию о рекомендуемой процедуре.

2. Если ошибка действительно имеет место, но она вызвана другим пакетом, просто переназначьте сообщение об ошибке тому пакету. Если вы не знаете, какому пакету следует переназначить сообщение об ошибке, вам следует попросить помощи в IRC или в debian-devel@lists.debian.org. Пожалуйста, проинформируйте сопровождающего того пакета, которому вы переназначаете сообщение об ошибке, например, отправив копию сообщения для системы отслеживания ошибок, содержащего команды для переназначения, по адресу *имя-пакета*@packages.debian.org, объясните ему причины в своём сообщении. Заметьте, что простое переназначение *не* пересылается сопровождающим пакета, которому вы переназначаете сообщение об ошибке, поэтому они не будет знать об этом до тех пор, пока они не посмотрят сводную информацию об ошибках в своих пакетах.

   Если ошибка затрагивает работу вашего пакета, вы можете клонировать сообщение об ошибке и переназначить полученную копию тому пакету, который фактически вызывает нежелательное поведение в вашем пакете. В противном случае, ошибка не будет показываться в списке ошибок вашего пакета, что, вероятно, приведёт к тому, что пользователи будут снова и снова сообщать об одной и той же ошибке. Вам следует заблокировать "вашу" ошибку переназначенной, клонированной ошибкой, чтобы засвидетельствовать отношение между ними.

3. Иногда вам также необходимо изменить важность ошибки так, чтобы она соответствовала вашему определению её важности. Люди обычно склонны преувеличивать важность ошибок для того, чтобы

их ошибки были быстрее исправлены. Важность некоторых ошибок может понижена до wishlist, если запрашиваемое изменение является скорее косметическим.

4. Если ошибка действительно имеет место, но об этой же проблеме было сообщено кем-то ещё, то два релевантных сообщения должны быть объединены в одно с помощью команды merge. В этом случае если ошибка будет исправлена, все те, кто сообщил о ней, будут уведомлены об этом. (Тем не менее, заметьте, что сообщения, отправленные одному из тех, кто сообщил об ошибке, не будут автоматически перенаправлены остальным пользователям, которые тоже сообщили об ошибке.) Подробности о технической стороне команды merge и родственной ей команды unmerge, см. в документации по управляющему серверу системы отслеживания ошибок.

5. Пользователь, отправивший сообщение об ошибке, мог забыть предоставить какую-то информацию, в этом случае вам следует попросить их предоставить необходимую информацию. Вы можете использовать тег `moreinfo`, чтобы отметить сообщение об ошибке как требующее предоставления дополнительной информации. Более того, если вы не можете воспроизвести ошибку, вы можете пометить сообщение об ошибке тегом `unreproducible`. Это будет означать, что если кто-то может воспроизвести ошибку, то вы просите его предоставить вам дополнительную информацию о том, как воспроизвести эту ошибку. Через несколько месяцев, если эта дополнительная информация не была никем отправлена, сообщение об ошибке может быть закрыто.

6. Если ошибка касается создания пакета, вам следует её исправить. Если вы не можете сами исправить её, то отметьте сообщение об ошибке тегом `help`. Кроме того, вы можете попросить о помощи в debian-devel@lists.debian.org или debian-qa@lists.debian.org. Если это проблема основной ветки разработки, вам следует переслать сообщение об ошибке автору основной ветки. Простой пересылки сообщения об ошибке не достаточно, вам следует проверять каждый выпуск основной ветки на предмет исправления этой ошибки. Если ошибка была исправлена, вам нужно просто закрыть сообщение об ошибке, в противном случае вам следует напомнить автору об ошибке. Если у вас имеются требуемые навыки, вы можете подготовить заплату, исправляющую ошибку и отправить её автору основной ветки. Убедитесь, что вы отправили заплату в систему отслеживания ошибок и пометили сообщение об ошибке тегом `patch`.

7. Если вы исправили ошибку в своей локальной копии пакета, либо если исправление было загружено в репозиторий системы управления версиями, вы можете отметить сообщение об ошибке тегом `pending`, что будет означать, что ошибка исправлена, и что сообщение об ошибке будет закрыто при следующей загрузке (добавьте пункт `closes:` в ваш файл `changelog`). Это особенно полезно, когда над одним и тем же пакетом работает несколько разработчиков.

8. Как только исправленный пакет будет доступен в архиве, сообщение об ошибке должно быть закрыто с указанием версии, в которой ошибка была исправлена. Это может быть сделано автоматически, см. раздел 5.8.4, «Когда ошибки исправляются путём новых загрузок».

5.8.4 Когда ошибки исправляются путём новых загрузок

Когда ошибки и проблемы в ваших пакетах будут исправлены, вам как сопровождающему следует закрыть сообщения об этих ошибках. Тем не менее, вам не следует закрывать сообщение об ошибке до того момента, как пакет, содержащий исправление ошибки, будет принят в архив Debian. Следовательно, как только вы получите уведомление о том, что загруженный вами пакет был установлен в архив, вы можете и должны закрыть сообщение об ошибке в системе отслеживания ошибок. Кроме того, сообщение об ошибке должно быть закрыто с указанием правильно версии, содержащей исправление ошибки.

Тем не менее, можно избежать необходимости ручного закрытия сообщений об ошибках после загрузки — просто перечислите ошибки в вашем файле `debian/changelog`, следуя определённым синтаксическим правилам, и ПО по сопровождению архива закроет соответствующие ошибки. Например:

```
acme-cannon (3.1415) unstable; urgency=low

  * Frobbed with options (closes: Bug#98339)
  * Added safety to prevent operator dismemberment, closes: bug#98765,
    bug#98713, #98714.
  * Added man page. Closes: #98725.
```

Технически говоря, следующее регулярное выражение Perl описывает то, как определяются журналы с закрытиями ошибок:

```
/closes:\s*(?:bug)?\#\s*\d+(?:,\s*(?:bug)?\#\s*\d+)*/ig
```

Мы предпочитаем синтаксис `closes: #XXX`, поскольку он является наиболее точным, также он проще в плане интеграции с текстом файла `changelog`. Если не указано обратное с помощью параметра `-v` для **dpkg-buildpackage**, будут закрыты только те сообщения об ошибках, которые указаны как закрытые в самой последней записи журнала изменений (по сути, будут закрыты именно те сообщения об ошибках, которые указаны в файле в разделе об изменениях в файле `.changes`).

Исторически загрузки определяются как загрузки теми, кто не является сопровождающим (NMU) отмечались тегом `fixed`, но не закрывались, но эта практика была прекращена с приходом отслеживания версий. То же самое справедливо и по отношению к тегу `fixed-in-experimental`.

Если вы неправильно введёте номер сообщения об ошибке или забудете указать этот номер в записи журнала изменений, не стесняйтесь исправить эту ошибку и её последствия. Чтобы заново открыть ошибочно закрытое сообщение об ошибке, отправьте команду `reopen XXX` на адрес системы управления системы отслеживания ошибок, control@bugs.debian.org. Чтобы закрыть любое сообщение об ошибках, указанная в котором ошибка была исправлена в вашей загрузке, перешлите файл `.changes` на адрес *XXX*-done@bugs.debian.org, вместо *XXX* укажите номер ошибки, также добавьте Version: *YYY* и пустую строку в качестве первых двух строк тела вашего сообщения, замените *YYY* на номер версии, в которой ошибка была исправлена.

Помните, что закрывать ошибки с помощью журнала изменений, как это описано выше, не является чем-то обязательным. Если вы просто хотите закрыть сообщения об ошибках, которые никак не касаются сделанной вами загрузки, закройте ошибку путём отправки вашего объяснения на адрес *XXX*-done@bugs.debian.org. **Не** закрывайте ошибки в записи журнала изменений какой-либо версии пакета, если изменения в этой версии пакета не имеют ничего общего с исправление этой ошибки.

Общую информацию о том, как писать журнал изменений, см. в разделе 6.3, «Лучшие практики для `debian/changelog`».

5.8.5 Работа с ошибками, связанными с безопасностью

Из-за того, что ошибки, связанные с безопасностью, обладают скорее некоторой чувствительной природой, с ними следует работать очень внимательно. Команда безопасности Debian существует для того, чтобы координировать эту работу, следить за серьёзными проблемами безопасности, помогать сопровождающим в их работе с проблемами безопасности или исправлять эти проблемы, отправлять рекомендации по безопасности и сопровождать `security.debian.org`.

Если вам становится известно о какой-либо ошибке в пакете Debian, связанной с безопасностью, вне зависимости от того, являетесь вы сопровождающим этого пакета или нет, соберите информацию о проблеме, и сразу же свяжитесь с командой безопасности по электронной почте: team@security.debian.org. Если хотите, то можете зашифровать ваше сообщение ключом Debian Security Contact, см. `https://www.debian.org/security/faq#contact`. **НЕ ЗАГРУЖАЙТЕ** какие-либо пакеты в `стабильный` выпуск, не связавшись с командой безопасности. В качестве полезной информации понимается следующее:

- Известно ли об этой ошибке широкой публике.
- Какие версии пакета подвержены данной ошибке. Проверьте каждую версию, которая в настоящее время поддерживается выпуском Debian, а также `тестируемый` и `нестабильный` выпуски.
- Суть исправления, если оно доступно (заплаты особенно полезны)
- Любые исправленные пакеты, которые вы подготовили самостоятельно (отправьте только debdiff или только файлы `.diff.gz` и `.dsc` и прочтите раздел 5.8.5.4, «Подготовка пакетов для решения проблем безопасности»)
- Любая помощь, которую вы можете предоставить в плане тестирования (уязвимость, тестирование регрессий и т. д.)
- Любая информация, необходимая для рекомендации по безопасности (см. раздел 5.8.5.3, «Рекомендации по безопасности»)

Как сопровождающий пакетов вы ответственны за сопровождение своих пакетов даже в стабильном выпуске. Вы имеете лучшие возможности для оценки заплат и тестирования обновлённых пакетов, поэтому, пожалуйста, внимательно изучите приводимую ниже информацию о том, как подготовить пакеты для работы команды безопасности.

5.8.5.1 Система отслеживания безопасности

Команда безопасности сопровождает центральную базу данных, систему отслеживания безопасности Debian. Она содержит всю публично доступную информацию о том, что известно о проблемах безопасности: какие пакеты и версии подвержены проблемами, либо были исправлены, а также какие выпуски, стабильный, тестируемый и/или нестабильный, уязвимы. Конфиденциальная информация не добавляется в эту систему.

Вы можете производить поиск по конкретной проблеме или имени пакета. Посмотрите ваш пакет, чтобы узнать, какие проблемы всё ещё открыты. Если вы можете, предоставьте дополнительную информацию об этих проблемах, либо помогите решить их в вашем пакете. Все инструкции доступны на страницах системы отслеживания проблем.

5.8.5.2 Конфиденциальность

В отличии от большей части другой деятельности, которая происходит в Debian, информация о проблемах безопасности иногда держится в секрете в течении некоторого времени. Это позволяет поставщикам ПО координировать раскрытие этой информации, чтобы минимизировать опасность для своих пользователей. Временное закрытие информации об уязвимости зависит от природы проблемы и соответствующего исправления, а также от того, находится ли уже информация об уязвимости в открытом доступе.

Разработчики могут узнать о проблемах безопасности из следующих источников:

- они могут найти упоминание проблемы на публичном форуме (списке рассылки, веб-сайте и т. д.)
- кто-то отправляет отчёт об ошибке
- кто-то информирует через частную почту

В первых двух случаях информация публична, важно подготовить исправление ошибки как можно раньше. Тем не менее, в последнем случае информация об ошибке может не быть публичной. Тогда при работе с проблемой возможны несколько вариантов действий:

- Если проблема безопасности не значительна, иногда нет необходимости скрывать информацию об этой проблеме, следует подготовить исправление и выпустить его.
- Если проблема относится к серьёзным проблемам, желательно сообщить о ней другим поставщикам и скоординировать выпуск. Команда безопасности постоянно находится на связи с различными организациями и индивидами, и заботится о распространении информации и координации.

Если тот, кто сообщает о проблеме, просит не раскрывать её, то в любом случае такой запрос следует уважать, конечно, это не касается информирования команды безопасности с целью подготовки исправления для стабильного выпуска Debian. При отправка конфиденциальной информации команде безопасности, обязательно упомяните о полученной просьбе.

Заметьте, что если требуется сохранить тайну, то вы не можете загрузить исправление в `нестабильный` выпуск (или куда-либо ещё, например, в публичный репозиторий системы управления версиями). Запутывания информации о подробностях изменения не достаточно, поскольку сам код доступен общественности и может быть (и будет) просмотрен остальными людьми.

Однако имеются две причины выпускать информацию даже в том случае, если было запрошено этого не делать: проблема уже была известна в течении некоторого времени, либо проблема или эксплоит доступны публично.

Команда безопасности для общения по поводу чувствительных вопросов использует закодированную при помощи ключа PGP переписку. Подробности см. в ЧаВО команды безопасности.

5.8.5.3 Рекомендации по безопасности

Рекомендации по безопасности выпускаются только для текущего стабильного выпуска, а *не* для `тестируемого` или `нестабильного` выпусков. Рекомендации при их выпуске отправляются в список рассылки debian-security-announce@lists.debian.org и размещаются на веб-странице о безопасности. Рекомендации по безопасности пишутся и публикуются командной безопасности. Тем не менее, они вовсе не против того, чтобы сопровождающий предоставил им какую-либо информацию или написал часть текста. В рекомендации по безопасности должна содержаться следующая информация:

- Описание проблемы и её масштаба, включая следующее:

  - Тип проблемы (повышение привилегий, отказ в обслуживании и т. д.)
  - Какие привилегии могут быть получены и кем (если это имеет место)
  - Как эта уязвимость может использоваться
  - Может ли она использоваться удалённо, либо локально
  - Как проблема была исправлена

  Эта информация позволяет пользователям оценить угрозу их системам

- Номера версий подверженных проблеме пакетов
- Номера версий исправленных пакетов
- Информация о том, где можно получить обновлённые пакеты (обычно из архива безопасности Debian)
- Ссылки на рекомендации по безопасности основной ветки разработки, идентификационные номера CVE и любую другую информацию, полезную для перекрёстного указания уязвимости

5.8.5.4 Подготовка пакетов для решения проблем безопасности

Вы можете помочь команде безопасности, если предоставите им исправления пакетов, подходящие для рекомендации по безопасности для стабильного выпуска Debian.

Когда вы производите обновление стабильного выпуска, следует быть осторожным в том, чтобы не изменить поведение системы и не внести новые ошибки. Для того, чтобы сделать это, для исправления ошибки производите как можно более мелкие изменения. Пользователи и администраторы полагаются на строго определённое поведение ПО, поэтому любое изменение может сломать работу чьей-то системы. Это в особенности касается библиотек: убедитесь, что вы не изменили API или ABI, причём не важно, насколько малым было изменение.

Это означает, что переход на новую версию из основной ветки разработке не является хорошим решением. Вместо этого вам следует осуществить обратный перенос релевантных изменений в версию ПО, которая входит в стабильный выпуск Debian. Как правило, сопровождающие основной ветки разработки помогают сделать это, если это необходимо. Если же нет, то вам может помочь команда безопасности Debian.

В некоторых случаях обратный перенос исправления безопасности невозможен, например, когда должно быть изменено или переписано большое количество исходного кода. Если это имеет место, может потребоваться переход на новую версию основной ветки разработки. Тем не менее, это осуществляется только в крайних случаях, вам всегда следует заранее координировать такой переход с командой безопасности.

С этим де связан и другой важный совет: всегда тестируйте ваши изменения. Если у вас имеется эксплоит, попробуйте использовать его и выяснить, работает ли он на изначальном пакете и действительно ли он не работает на исправленном пакете. Также протестируйте другие, обычные действия, поскольку исправление безопасности может поломать даже кажущиеся несвязанными возможности.

Никогда **НЕ** добавляйте какие-либо изменения в ваш пакет, которые не связаны напрямую с исправлением уязвимости. Это лишь потребует вернуть изменения, это лишь зря потратит время разработчика. Если в вашем пакете имеются другие ошибки, которые вам хотелось бы исправить, подготовьте загрузку в proposed-updates обычным образом уже после того, как будет выпущена рекомендация по безопасности. Механизм обновления безопасности не служит для внесения в ваши пакеты таких изменений, которые в противном случае были бы отвернуты при их загрузке в стабильный выпуск, поэтому, пожалуйста, не пытайтесь этого делать.

Проверьте и протестируйте ваши изменения столько раз, сколько это возможно. Проверьте отличия от предыдущей версии (для этого очень полезны **interdiff** из пакета `patchutils` и **debdiff** из пакета `devscripts`, см. раздел A.2.2, «**debdiff**»).

Убедитесь, что вы проверили следующее:

- **Укажите верный выпуск** в вашем файле `debian/changelog`: *`кодовое-имя`*`-security` (напр., `jessie-security`). Не указывайте *`выпуск`*`-proposed-updates` или `stable`!
- Загрузка должна иметь **urgency=high**.
- Ваши записи в журнале изменений должны быть информативны и осмысленны. Другие пользователи будут полагаться не них при определении того, была исправлена определённая ошибка или нет. Добавляйте записи `closes:` о любых **ошибках Debian**. Всегда добавляйте внешнюю ссылку, желательно **идентификатор CVE**. Тем не менее, если идентификатор CVE ещё не был назначен, не ждите его, продолжайте процесс. Указать идентификатор можно позже.

- Убедитесь, что **номер версии** верен. Он должен быть больше, чем номер версии текущего пакета, но меньше, чем версии пакетов в более поздних выпусках. Если вы сомневаетесь относительно версии, проверьте её с помощью `dpkg --compare-versions`. Будьте внимательны, не используйте повторно номер версии, который вы использовали для предыдущей загрузки, либо номер, конфликтующий с binNMU. Принято добавлять к номеру версии `+debXu1` (где *X* представляет собой главный номер выпуска), напр., `1:2.4.3-4+deb8u1`, для последующей загрузки версию, конечно же, следует увеличить на 1.
- Если исходный код из основной ветки разработки не был ранее загружен на `security.debian.org` (во время предыдущего обновления безопасности), соберите свою загрузку **с полным исходным кодом из основной ветки разработки** (`dpkg-buildpackage -sa`). Если же ранее была произведена загрузка на `security.debian.org`, содержащая ту же самую версию из основной ветки разработки, вы можете осуществить загрузку без добавления исходного кода из основной ветки (`dpkg-buildpackage -sd`).
- Убедитесь, что используется **в точности тот же файл `*.orig.tar.{gz,bz2,xz}`**, который использовался в обычном архиве, в противном случае переместить исправление безопасности в основной архив будет нельзя.
- Соберите пакет в **чистой системе**, в которой установлен пакеты только из того выпуска, для которого вы собираете свой пакет. Если у вам нет такой системы, вы можете использовать машину debian.org (см. раздел 4.4, «Машины Debian»), либо настроить chroot (см. раздел A.4.3, «`pbuilder`» и раздел A.4.2, «`debootstrap`»).

5.8.5.5 Загрузка исправленного пакета

Никогда **НЕ** загружайте пакет в очередь обновлений безопасности (на `security-master.debian.org`) без соответствующего разрешения от команды безопасности. Если пакет не соответствует требованиями команды безопасности, работа с нежелательной загрузкой приведёт к возникновению множества проблем и задержек.

Никогда **НЕ** загружайте ваше исправление в `proposed-updates`, не связавшись с командой безопасности. Пакеты из `security.debian.org` будут скопированы в каталог `proposed-updates` автоматически. Если какой-то пакет с тем же самым или более высоким номером версии уже установлен в архиве, обновление безопасности будет отклонено системой, управляющей архивом. В этом случае стабильный выпуск останется без обновления безопасности.

Как только вы создадите и протестируете новый пакет, когда он будет утверждён командой безопасности, этот пакет следует загрузить, чтобы он был установлен в архивы. Для обновлений безопасности пакеты следует загружать на `ftp://security-master.debian.org/pub/SecurityUploadQueue/`.

Как только будет принята ваша загрузка в очередь обновления безопасности, пакет будет автоматически собран для всех архитектур и сохранён для проверки командой безопасности.

Загрузки, ожидающие принятия или проверки, доступны только членам команды безопасности. Это необходимо потому, что они могут содержать исправления таких проблем безопасности, информация о которых пока не может быть раскрыта.

Если член команды безопасности принимает пакет, этот пакет будет установлен в `security.debian.org`, а также предложен для соответствующего каталога *`выпуск`*`-proposed-updates` на `ftp-master.debian.org`.

5.9 Перемещение, удаление, переименование, придание статуса осиротевшего, усыновление и повторное введение пакетов

Некоторые действия по управлению архивом в Debian не автоматизированы во время загрузки. Эти процедуры должны быть вручную выполнены сопровождающими. В данной главе содержатся советы о том, что следует делать в этих случаях.

5.9.1 Перемещение пакетов

Иногда для какого-то пакета раздел может быть изменён. Например, пакет из раздела `non-free` (раздел несвободных пакетов) может быть перелицензирован под GPL, в этом случае этот пакет следует переместить в main' (основной раздел) или contrib' (раздел ПО, зависящего от несвободного ПО).[1]

[1] См. Руководство по политике Debian для получения советов о том, к какому разделу относится тот или иной пакет.

Если вам нужно изменить раздел у одного из ваших пакетов, измените управляющую информацию о пакете, затем заново загрузите ваш пакет (подробности см. в Руководстве по политике Debian). Вам следует чётко убедиться, что вы добавили `.orig.tar.{gz,bz2,xz}` в вашу загрузку (даже если вы не загружаете новую версию из основной ветки разработки), либо что этот файл не будет встречаться в новом разделе с остальной частью пакета. Если ваш новый раздел верен, пакет будет перемещён автоматически. Если же пакет не будет перемещён, тогда свяжитесь с управляющими ftp для того, чтобы понять, что же произошло.

Если, с другой стороны, вам необходимо изменить `подраздел` одного из ваших пакетов (напр., „devel“, „admin“), то это уже немного другая процедура. Исправьте подраздел в управляющем файле вашего пакета, затем заново загрузите пакет. Кроме того, вам следует обновить файл отклонений, это описано в разделе 5.7, «Определение раздела для пакета, подраздела и приоритета».

5.9.2 Удаление пакетов

Если по какой-то причине вам необходимо полностью удалить пакет (например, если он представляет собой устаревшую библиотеку для совместимости, которая более не требуется), вам необходимо отправить отчёт об ошибке в псевдопакете `ftp.debian.org` с просьбой об удалении вашего пакета; как и все сообщения об ошибках, это сообщение об ошибках обычно должно иметь важность normal. Заголовок сообщения об ошибке должен иметь вид `RM: пакет [список архитектур] -- причина`, где *пакет* — это пакет, который следует удалить, а *причина* — краткое описание причины, по которой вы отправляете запрос на удаление. *[список архитектур]* опционален и и требуется только в том случае, если пакет необходимо удалить только для некоторых архитектур, а не для всех. Заметьте, что утилита **reportbug** создаст заголовок, соответствующий указанным правилам, если вы будете использовать её для отправки сообщения об ошибке в псевдопакете `ftp.debian.org`.

Если вы хотите удалить сопровождаемый вами пакет, вам следует указать это в заголовке вашего сообщения об ошибке, добавив в начале `ROM` (Request Of Maintainer, Запрос сопровождающего). Имеется некоторое количество других стандартных сокращений, используемых при удалении пакетов, полный список см. на странице `https://ftp-master.debian.org/removals.html`. Кроме того, на этой странице представлен удобный обзор обработки запросов об удалении.

Заметьте, что удаления могут быть выполнены только для `нестабильного`, `экспериментального` и `стабильного` выпусков. Пакеты из `тестируемого` выпуска напрямую не удаляются. Они будут удалены автоматически после удаления пакетов из `нестабильного` выпуска, и если в `тестируемом` выпуске от них не зависит ни один пакет. (Удаления из `тестируемого` выпуска возможны при отправке сообщения об ошибке в псевдопакете `release.debian.org`. См. раздел 5.13.2.2, «Удаление из тестируемого выпуска».)

Имеется одно исключение, при котором явный запрос об удалении не требуется: если двоичный пакет или пакет с исходным кодом более не может быть собран из исходного кода, в этом случае он будет удалён в полуавтоматическом режиме. Для двоичного пакета это означает, что у него более не имеется пакета с исходным кодом, который создаёт данный двоичный пакет; если двоичный пакет не создаётся на некоторых архитектурах, запрос об удалении этого пакета всё равно требуется. Для пакета с исходным кодом это означает, что все двоичные пакеты, на которые ссылается этот пакет, были приняты другим пакетом с исходным кодом.

В вашем запросе об удалении вам следует подробно описать причины, обосновывающие ваше требование. Это позволит избежать нежелательных удалений и отследить то, почему пакет был удалён. Например, вы можете указать имя пакета, которые заменяет удаляемый пакет.

Обычно просят удалить те пакеты, которые сопровождаются самим запрашивающим об удалении. Если вы хотите удалить другой пакет, вам следует получить на это разрешение от сопровождающего этого пакета. Если пакет должен быть признан осиротевшим, если он не имеет сопровождающего, то для начала вам следует обсудить запрос об удалении в debian-qa@lists.debian.org. В случае если было принято решение об удалении пакета, вам следует переназначить сообщение об ошибке в пакете `wnpp` и изменить его заголовок на `O:` вместо того, чтобы отправлять новое сообщение об ошибке с запросом об удалении.

Дополнительная информация, связанная с этими вопросами, а также по теме удаления других пакетов может быть найдена по адресу `https://wiki.debian.org/ftpmaster_Removals` и `https://qa.debian.org/howto-remove.html`.

Если вы сомневаетесь в том, может ли быть удалён какой-либо пакет, обратитесь по адресу debian-devel@lists.debian.org с этим вопросом. Кроме того, интерес представляет программа **apt-cache** из пакета `apt`. При запуске `apt-cache showpkg пакет`, эта программа отобразит подробную информацию о *пакете*, включая его обратные зависимости. Другими полезными программами являются **apt-cache rdepends**, **apt-rdepends**, **build-rdeps** (из пакета `devscripts`) и **grep-dctrl**. Удаление осиротевших пакетов обсуждается в debian-qa@lists.debian.org.

После удаления пакета следует разобраться с его сообщениями об ошибках. Они должны быть либо переназначены другому пакету в случае, когда фактический код ПО развился в другой пакет (напр., `libfoo12` был удалён из-за того, что `libfoo13` его вытеснил), либо закрыты если это ПО более не является частью Debian. Закрывая сообщения об ошибках, не отмечайте их как исправленный в версиях пакета из предыдущих выпусков Debian, они должны быть отмечены как исправленные в `<наиболее -свежей-версии-в-Debian>+rm`.

5.9.2.1 Удаление пакетов из каталога `Incoming` (каталога входящих пакетов)

Раньше можно было удалять пакеты из каталога `incoming`. Тем не менее, в введением новой системы входящих пакетов это более не возможно. Вместо этого вам следует загрузить новую версию вашего пакета с более высоким номером версии, чем имеет тот пакет, который вы хотите заменить. Обе версии будут установлены в архив, но лишь пакет с более высокой версией будет фактически доступен в `нестабильном` выпуске, поскольку предыдущая версия будет сразу же заменена более новой версией. Тем не менее, если вы хорошо тестируете ваши пакеты, нужды заменять пакет скорее всего не возникнет.

5.9.3 Замена или переименование пакетов

Если сопровождающие основной ветки разработки решают изменить название своего ПО (либо если вы дали вашему пакету неверное имя), вам необходимо следовать процессу по переименованию пакета, который включает в себя два шага. На первом шаге измените файл `debian/control` так, чтобы в нём было отражено новое имя пакета, а также для того, чтобы устаревшее имя пакета было указано для замены, предоставления и в списке конфликтующих пакетов (см. Руководство по политике Debian). Заметьте, что вам следует добавлять отношение `Provides` только тогда, когда все пакет, зависящие от устаревшего пакета, продолжают работать после переименования. Когда вы загрузите пакет, и он будет перемещён в архив, отправьте сообщение об ошибке в псевдопакете `ftp.debian.org` с просьбой удалить пакет с устаревшим именем (см. раздел 5.9.2, «Удаление пакетов»). Не забудьте правильно переназначить сообщения об ошибке, чтобы они отсылали к новому имени пакета.

Вы можете допустить ошибку при создании вашего пакета, тогда вы захотите заменить свой пакет. Единственным способом сделать это является увеличение номера версии и загрузка новой версии пакет. Старая версия как обычно станет устаревшей. Заметьте, что это касается каждой части вашего пакета, включая и исходный код: если вы хотите заменить архив tarball с исходным кодом из основной ветки, вам придётся загрузить ваш пакет с другим номером версии. Проще всего заменить `foo_1.00.orig.tar.gz` на `foo_1.00+0.orig.tar.gz`, либо `foo_1.00.orig.tar.bz2`. Данное ограничение позволяет каждому файлу на ftp иметь уникальное имя, что помогает гарантировать согласованность в сети зеркал.

5.9.4 Придание статуса осиротевшего пакета

Если вы не можете более сопровождать пакет, вам нужно сообщить об этом остальным, и убедиться, что ваш пакет отмечен как осиротевший. Вам следует установить сопровождающего пакета в значение `Debian QA Group <packages@qa.debian.org>` и отправить сообщение об ошибке в псевдопакете `wnpp`. Заголовок сообщения об ошибке должен иметь вид `O: пакет -- краткое описание`, что показывает, что пакет осиротел. Важность сообщения об ошибке должна иметь значение `normal`; если пакет имеет стандартный приоритет или выше, важность сообщения об ошибке должна иметь значение important. Если вы считаете это необходимым, отправьте копию сообщения на адрес debian-devel@lists.debian.org, добавив этот адрес в заголовок X-Debbugs-CC: вашего сообщения (не используйте CC:, так как тогда тема сообщения не будет содержать номер сообщения об ошибке).

Если вы хотите отдать пакет, но вы можете пока продолжать сопровождать его, то вам следует отправить сообщения об ошибке в псевдопакете `wnpp`, сообщение должно иметь заголовок вида `RFA: пакет -- краткое описание`. `RFA` означает `Request For Adoption` (запрос об усыновлении).

Дополнительная информация доступна на веб-страницах WNPP.

5.9.5 Усыновление пакета

Список пакетов, которым требуются новые сопровождающие, доступен на странице Требующих доработки и будущих пакетов (WNPP). Если вы хотите заняться сопровождением какого-либо пакета из тех, что приведены в списке WNPP, обратитесь к вышеупомянутой странице для получения информации и сведений о процедуре.

Нельзя просто взять и заняться работой над пакетом, который кажется вам заброшенным — это было бы кражей пакета. Вы, конечно, можете связаться с текущим сопровождающим и спросить его об этом. Если у

вас имеются основания полагать, что сопровождающий просто отсутствует без уведомления об этом (AWOL, absent without leave), см. раздел 7.4, «Работа с неактивными и/или недоступными сопровождающими».

Как правило, вы не можете заняться работой над пакетом без специального согласия на это от текущего сопровождающего. Даже если он вас игнорирует это всё равно не является основанием для начала работы над пакетом. Жалобы на сопровождающих должны высказываться в списке рассылки для разработчиков. Если дискуссия не заканчивается позитивным заключением, а проблема имеет техническую природу, подумайте над тем, чтобы привлечь к ней внимание технического комитета (дополнительную информацию см. на странице технического комитета).

Если вы занялись работой над старым пакетом, вам вероятно захочется, чтобы в системе отслеживания ошибок в качестве официального сопровождающего этого пакета были указаны вы. Это будет сделано автоматически как только вы загрузите новую версию с обновлённым полем `Maintainer`, хотя на это может потребоваться несколько часов после выполнения загрузки. Если вы пока не собираетесь загружать новую версию, вы можете использовать информацию из раздела 4.10, «Система отслеживания пакетов Debian» для получения сообщений об ошибках. Тем не менее, убедитесь, что старый сопровождающий не против того факта, что он будет получать сообщения об ошибках в течении какого-то времени.

5.9.6 Повторное добавление пакетов

Пакеты часто удаляются из-за наличия в них критичных для выпуска ошибок, отсутствия сопровождающих, слишком малого числа пользователей или низкого качества. Хотя процесс повторного добавления пакета схож с первоначальной работой над пакетом, вы можете избежать некоторых ловушек, выполнив для начала небольшое историческое исследование.

Для начала вам следует проверить, почему пакет был удалён. Эта информация может быть найдена в сообщении об удалении в разделе новостей на странице системы отслеживания пакетов данного пакета, либо при просмотре журнала удалений. Сообщение об удалении в системе отслеживания ошибок содержит сведения о том, почему пакет был удалён, а также покажет вам то, над чем вам следует поработать для того, чтобы заново добавить пакет. В сообщении может содержаться сведения о том, что вместо повторного добавления пакета лучше всего переключиться на какое-то другое ПО.

Хорошо было бы связаться с предыдущими сопровождающими и выяснить, работают ли они над повторным добавлением пакета, заинтересованы ли они в совместном сопровождении пакета, заинтересованы ли они быть наставником пакета, если это необходимо.

Вам следует сделать всё, что требуется, до того как добавлять новые пакеты (раздел 5.1, «Новые пакеты»).

В качестве основы для своей работы вам следует взять самый последний из доступных и подходящих пакетов. Это может быть самая последняя версия из `нестабильного` выпуска, которая всё ещё доступна в снимке архива.

Система контроля версий, которая использовалась предыдущим сопровождающим, может содержать полезные изменения, поэтому проверка репозитория может оказаться хорошей идеей. Проверьте, содержит ли файл `control` в предыдущем пакете какие-либо заголовки со ссылками на систему контроля версий для данного пакета, и если да, то существует ли всё ещё этот репозиторий.

Удаление пакета из `нестабильного` выпуска (не из `тестируемого`, `стабильного` или `предыдущего стабильного` выпусков) приводит к закрытию всех сообщений об ошибках, которые связаны с удаляемым пакетом. Вам следует проверить все закрытые сообщения об ошибках (включая архивированные сообщения об ошибках), разархивировать и заново открыть любые закрытые сообщения об ошибках в версии, заканчивающейся на `+rm`, и которые всё ещё актуальны. Любые неактуальные сообщения об ошибках должны быть помечены как исправленные в той версии, в которой они были исправлены, если это, конечно, известно.

5.10 Работа на переносом и перенос пакетов

Debian поддерживает всё увеличивающееся число архитектур. Даже если вы не занимаетесь переносом и используете только одну архитектуру, вы как сопровождающий обязаны знать о проблемах, связанных с переносимостью. Следовательно, даже если вы не занимаетесь переносом, вам следует прочесть большую часть данной главы.

Перенос представляет собой сборку пакетов Debian для архитектур, которые отличаются от архитектуры для которой сопровождающим был изначально собран двоичный пакет. Это уникальная и важная работа. В действительности, те, кто занимаются переносом, выполняют большую часть фактической компиляции пакетов Debian. Например, когда сопровождающий загружает (переносимые) пакеты с исходным кодом с

двоичными пакетами для архитектуры `i386`, двоичные пакеты будут собраны для каждой из оставшихся архитектур, что равно 11 сборкам.

5.10.1 Будьте добры к тем, кто занимается переносом

У тех, кто занимается переносом, трудная и необычная задача, поскольку им необходимо работать с большим объёмом пакетов. В идеале каждый пакет с исходным кодом должен собираться прямо из коробки. К сожалению, зачастую это не так. Данный раздел содержит список „ошибочек“, которые часть совершают сопровождающие Debian — общих проблем, которые ставят в тупик тех, кто занимается переносом, и делают их работу неоправданно сложной.

В первую очередь (и это самое важное) следует быстро отвечать на сообщения об ошибках или проблемах, которые были присланы теми, кто занимается переносом. Пожалуйста, обращайтесь с этими людьми вежливо, как будто они являются вашими помощниками в работе по сопровождению вашего пакета (а они фактически ими и являются). Будьте терпимы к кратким или даже неясным сообщениям об ошибках; приложите все усилия для того, чтобы выяснить, в чем состоит проблема.

По большей части, почти все проблемы, на которые обращают внимание те, кто занимает переносом, вызваны *ошибками, допущенными при создании пакетов*, в пакетах с исходным кодом. Ниже приведён список того, что вам следует проверить или о чём нужно всегда помнить.

1. Убедитесь, что ваши настройки `Build-Depends` и `Build-Depends-Indep` в `debian/control` выставлены правильно. Лучшим способом проверить это является использование пакета `debootstrap` для создания на базе `нестабильного` выпуска окружения chroot (см. раздел A.4.2, «`debootstrap`»). Внутри окружения chroot установите пакет `build-essential` и указанные в `Build-Depends` и/или `Build-Depends-Indep` зависимости вашего пакета. Наконец, попытайтесь собрать ваш пакет в полученном окружении chroot. Эти шаги могут быть автоматизированы, если вы будете использовать программу **pbuilder**, которая входит в пакет с тем же именем (см. раздел A.4.3, «`pbuilder`»).

   Если вы не можете правильно настроить chroot, вам может помочь **dpkg-depcheck** (см. раздел A.6.6, «**dpkg-depcheck**»).

   Инструкции по настройке сборочных зависимостей см. в Руководстве по политике Debian.

2. Не указывайте в качестве архитектуры отличное от `all` ли `any` значение, если только вы действительно не имеете это в виду. В большинстве случаев сопровождающие не следуют инструкциям Руководства по политике Debian. Установка архитектуры в значение только какой-то одной архитектуры (такой как `i386` или `amd64`) обычно оказывается неправильным.

3. Убедитесь, что ваш пакет с исходным кодом корректен. Выполните `dpkg-source -x` *`пакет`*`.dsc` для того, чтобы убедиться, что ваш пакет с исходным кодом распаковывается правильно. Далее, попытайтесь собрать ваш пакет с нуля с помощью **dpkg-buildpackage**.

4. Убедитесь, что в вашем пакете с исходным кодом нет файлов `debian/files` или `debian/substvars`. Они должны быть удалены при помощи цели `clean` из `debian/rules`.

5. Убедитесь, что вам не требуются локально установленные или исправленные настройки или программы. Например, вам никогда не следует вызывать программы из `/usr/local/bin` и других подобных место. Попытайтесь не использовать программы, которые были настроены каким-то специальным способом. Попытайтесь собрать ваш пакет на другой машине, даже если она принадлежит к той же архитектуре.

6. Не используйте при сборке пакета какие-либо уже установленные пакеты (это более конкретный вариант приведённого выше случая). Конечно, бывают и исключения для этого правила, но помните, что любой подобный случай требует ручного вмешательства и не может быть выполнен автоматическими сборщиками пакетов.

7. Если это возможно, не используйте компилятор какой-то конкретной версии. Если это невозможно, то убедитесь, что ваши сборочные зависимости отражают это ограничение, хотя вы, вероятно, просите о чём-то проблематичном, поскольку разные архитектуры иногда основываются на разных компиляторах.

8. Убедитесь, что ваш файл `debian/rules` содержит отдельные цели `binary-arch` и `binary-indep`, как то требуется в Руководстве по политике Debian. Убедитесь, что обе цели работают независимо друг от друга, то есть, что вы можете вызвать одну цель, не вызывая до этого другой. Для проверки этого, попытайтесь запустить **dpkg-buildpackage -B**.

5.10.2 Руководство для загрузок теми, кто занимается переносом

Если пакет собирается из коробки для той архитектуры, на которую он должен быть перенесён, то вам повезло и ваша работа довольно проста. Данный раздел касается этого случая; в разделе описывается то, как собирать и загружать ваш двоичный пакет так, чтобы он был правильно установлен в архив. Если вам требуется внести изменения в пакет для того, чтобы он мог быть скомпилирован для других архитектур, вам будет нужно выполнить NMU пакета с исходным кодом, поэтому обратитесь к разделу 5.11.1, «Когда и как делать NMU».

При загрузке тем, кто занимается переносом, изменения исходного кода не вносятся. Вам не нужно трогать какие-либо файлы в пакете с исходным кодом. Это предполагает и файл `debian/changelog`.

Можно вызвать **dpkg-buildpackage** как `dpkg-buildpackage -B -m`*`porter-email`*. Конечно, установите ваш адрес электронной почты в качестве значения поля *`porter-email`*. При использовании цели `binary-arch` в `debian/rules` будет выполнена сборка двоичных пакетов только для зависящих от архитектуры частей пакета.

Если вы работаете на машине Debian для того, чтобы заниматься переносом, и вам локально требуется подписать вашу загрузку для того, чтобы она была принята в архив, вы можете запустить **debsign**, указав ваш файл `.changes` в целях удобства, файл будет подписан, либо вы можете использовать удалённый режим подписывания командой **dpkg-sig**.

5.10.2.1 Повторная компиляция или только двоичные NMU

Иногда первоначальная загрузка, связанная с переносом, проблематична из-за того, что окружение, в котором был собран пакет, не было достаточно хорошо (устаревшая или не используемая более библиотека, плохой компилятор и т. д.). Тогда вам может потребоваться заново скомпилировать пакет в обновлённом окружении. Тем не менее, в этом случае вам придётся увеличить номер версии пакета, чтобы старый плохой пакет был заменён на новый в архиве Debian (**dak** отказывается устанавливать новые пакеты, если номер их версии не больше уже доступных пакетов).

Вам следует убедиться, что ваша двоичная NMU не приведёт к тому, что пакет перестанет устанавливаться. Это может произойти в случае, если пакет с исходным кодом порождает зависящие и независящие от архитектуры пакеты, взаимные зависимости которые созданы подстановкой переменной dpkg `$(Source-Version)`.

Несмотря на необходимое изменение журнала изменений, это называется двоичной NMU — в этом случае нет необходимости делать пакеты других архитектур устаревшими и заново их компилировать.

Для такой повторной компиляции требуется специальное „магическое“ назначение версий, так чтобы инструменты для сопровождения архива распознали, что хотя даже и имеется новая Debian-версия, соответствующего обновления исходного кода не было. Если вы сделаете это неправильно, сопровождающие архива отклонят вашу загрузку (из-за отсутствия соответствующего исходного кода).

„Магия“ для NMU повторной компиляции включается путём использования суффикса в номере версии пакет, имеющего следующий вид: `b`*`номер`*. Например, если последняя версия пакета, которую вы заново компилируете, была версией `2.9-3`, ваша двоичная NMU должна иметь версию `2.9-3+b1`. Если последней версией была `3.4+b1` (т. е., родной пакет, для которого в предыдущий раз уже была выполнена NMU с повторной компиляцией), ваша двоичная NMU должна иметь номер версии `3.4+b2`.[2]

Подобно первоначальной загрузке, связанной с переносом, правильный вызов **dpkg-buildpackage** имеет вид `dpkg-buildpackage -B`, что означает, что будут собраны только зависящие от архитектуры части пакета.

5.10.2.2 Когда следует делать NMU, если вы занимаетесь переносом

Те, кто занимаются переносом и выполняют NMU исходного кода, следуют советам из раздела 5.11, «Загрузки не-сопровождающим (NMU)», как и те, кто переносом не занимается. Тем не менее, ожидается, что цикл ожидания для NMU исходного кода, выполненной тем, кто занимается переносом, меньше, чем для того, кто переносом не занимается, поскольку те, кто занимается переносом, должны охватить большое количество пакетов. Опять же, ситуация зависит от того, в какой выпуск они хотят осуществить загрузку. Также важно то, является ли данная архитектура кандидатом на включение в следующий стабильный выпуск; управляющие выпуском решают и сообщают о том, какие архитектуры являются такими кандидатами.

[2] В прошло подобные NMU использовали номер третьего уровня в части Debian редакции для обозначения статуса повторной компиляции; тем не менее, этот синтаксис был двусмыслен в случае родных пакетов и не позволяя правильно определять порядок заново скомпилированных NMU, NMU исходного кода и NMU безопасности одного и того же пакета, потому он был отброшен и заменён новым синтаксисом.

Если вы не занимаетесь переносом и выполняете NMU для нестабильного выпуска, вам следует придерживаться приведённых выше советов по переносу, но с двумя изменениями. Во-первых, период ожидания принятия — время между тем, когда было сообщено об ошибке в систему отслеживания ошибок, и тем, когда для NMU наступает подходящее время — равен семи дням для тех, кто занимается переносом, и работает над нестабильным выпуском. Этот период может быть уменьшен по усмотрению тех, кто занимается переносом, в случае, если проблема является критической и представляет собой серьёзную трудность для процесса переноса пакета. (Помните, это не является Политикой, это лишь негласно принято в сообществе.) Для выполнения загрузки в стабильный или тестируемый выпуски свяжитесь с соответствующей выпускающей командой.

Во-вторых, те, кто занимается переносом и выполняют NMU исходного кода, должны убедиться, что сообщение об ошибке, отправленное в систему отслеживания ошибок, имеет важность serious или выше. Это гарантирует, что отдельный пакет с исходным кодом может использоваться для компиляции пакета для каждой поддерживаемой Debian на момент выпуска архитектуры. Важно, что у нас имеется одна версия двоичного пакета и пакета с исходным кодом для всех архитектур для того, чтобы соответствовать множеству лицензий.

Те, кто занимаются переносом, должны попытаться избежать заплат, которые просто обходят ошибки в текущей версии окружения компиляции, ядра или libc. Иногда такие клуджи не могут помочь. Если вам нужно обойти ошибки компилятора или чего-то подобного, убедитесь, что вы выполнили корректно #ifdef для своей работы; также, документируйте ваши клуджи, чтобы люди знали, как удалить их когда внешние проблемы будут исправлены.

У тех, кто занимается переносом, также имеется неофициальное место, куда они могут поместить результаты своей работы во время периода ожидания. Это помогает остальным, кто занимается переносом, использовать вашу работу даже во время периода ожидания. Конечно, такие места не являются официальными и не имеют какого-либо специального статуса, поэтому будьте внимательны.

5.10.3 Инфраструктура переноса и автоматизация

Имеется специальная инфраструктура и некоторые инструменты, необходимые для автоматизации переноса пакетов. Данный раздел сдержит краткий обзор автоматизации и переноса с помощью данных инструментов; дополнительную информацию см. в документации пакетов или по другим ссылкам.

5.10.3.1 Списки рассылки и веб-страницы

Веб-страницы, содержащие информацию о статусе каждого переноса, можно найти по адресу: https://www.debian.org/ports/.

У каждого переноса Debian имеется свой список рассылки. Список списков рассылки различных проектов по переносу может быть найден по адресу: https://lists.debian.org/ports.html. Эти списки рассылки используются для координации работы и связи пользователей с теми, кто работает над переносом.

5.10.3.2 Инструменты переноса

Описания некоторых инструментов переноса могут быть найдены в разделе A.7, «Инструменты для переноса».

5.10.3.3 wanna-build

Система wanna-build используется как распределённая клиент-серверная система сборки дистрибутива. Обычно она используется вместе со сборочными службами, запускающими программу buildd. Сборочные службы являются „подчинёнными“ узлами, они связываются с центральной системой wanna-build для получения списка пакетов, которые следует собрать.

Система wanna-build пока недоступна в виде пакета; тем не менее, она используется для переноса Debian в режиме автоматической сборки пакетов. Инструмент, используемый для создания фактических сборок пакетов, sbuild, доступен в виде пакета, см. его описание в разделе A.4.4, «sbuild». Заметьте, что версия в пакете отличается от той, что используется в сборочных службах, но она достаточно близка к ней и поэтому подходит для воспроизведения проблем.

Большая часть данных, создаваемых системой wanna-build, которая обычно бывает полезна для тех, кто занимается переносом, доступна на следующей веб-странице: https://buildd.debian.org/. Эта информация включает в себя обновляемую каждую ночь статистику, информацию об очереди и журналы попыток сборки.

Мы очень гордимся этой системой, поскольку у неё так много возможных пользователей. Независимые от разработки группы могут использовать данную систему для подготовки различных вариантов Debian, которые иногда могут быть интересы и более широкому кругу пользователей (например, вариант Debian, собранный с поддержкой проверки связывания средствами **gcc**). Кроме того, она позволяет довольно быстро заново собрать целый выпуск Debian.

Связаться с командой wanna-build, ответственной за службы buildd, можно по адресу: debian-wb-team@lists.deb
Для того, чтобы определить, с кем (команда wanna-build, команда выпуска) и как (почта, система отслеживания ошибок) следует связаться в конкретном случае, см. https://lists.debian.org/debian-project/2009/03/msg00096.html.

При запросе binNMU или возвратов (повторов после неудачной сборки) используйте описанный в https://release.debian.org/wanna-build.txt формат.

5.10.4 Если ваш пакет *не* может быть перенесён

Некоторые пакеты всё равно имеют проблемы со сборкой и/или с работой на некоторых поддерживаемых Debian архитектурах, они вообще не могут быть перенесены, либо для их переноса требуется слишком много времени. Примером этого является пакет, который конкретно связан с SVGA (доступен только для i386 и amd64), либо использует другие специфические возможности оборудования, которые совсем не поддерживаются на всех архитектурах.

Для того, чтобы сломанные пакеты не были загружены в архив и не потратили зря время работы buildd, вам следует выполнить несколько вещей:

- Во-первых, убедитесь, что ваш пакет *действительно* не может быть собран на архитектурах, которые он не может поддерживать. Для достижения этого имеется несколько способов. Предпочтительный способ состоит в том, чтобы иметь небольшой тестовый набор для использования во время сборки, который будет проверять функциональность пакета и который будет завершаться неудачей в случае, если пакет не работает. Отличной идеей является и то, что это помешает (некоторым) сломанным загрузкам на все архитектуры, а также позволит пакету собираться сразу как только требуемая функциональность будет доступна.

  Кроме того, если вы убеждены, что список поддерживаемых архитектур вполне постоянен, вам следует изменить any на список поддерживаемых архитектур в debian/control. При этом способе сборка опять же завершиться неудачно, пользователю будет сообщено об этом, даже хотя попытки сборки не было.

- Для того, чтобы ПО для автоматической сборки не пыталось без необходимости на то собрать ваш пакет, в Packages-arch-specific должен быть включён список, используемый сценарием **wanna-build**. Текущая версия доступна как https://anonscm.debian.org/cgit/mirror/packages-arch-spe
  git/tree/Packages-arch-specific; с кем можно связаться по поводу изменений см. в начале файла.

Заметьте, что это недостаточно просто добавить ваш пакет в Packages-arch-specific без того, чтобы его сборка на неподдерживаемых архитектурах завершилась неудачно: тот, кто занимается переносом, либо кто-либо другой, пытающиеся собрать ваш пакет, может случайно загрузить его, не заметив, что пакет не работает. Если в прошлом некоторые двоичные пакеты были загружены и для неподдерживаемых архитектур, запросите их удаление, заполнив сообщение об ошибке в ftp.debian.org.

5.10.5 Отмечаем несвободные пакеты как собираемые автоматически (auto-buildable)

По умолчанию пакеты из раздела non-free не собираются сетью автоматической сборки (по большей части из-за того, что это может быть запрещено лицензией пакета). Чтобы разрешить сборку пакета, вам следует предпринять следующие шаги:

1. Проверьте, законно ли и возможно ли технически автоматически собирать данный пакет;
2. Добавьте XS-Autobuild: yes в заголовок файла debian/control;
3. Отправьте сообщение по электронной почте на адрес nonfree@release.debian.org, объясните, почему пакет можно собрать как с правовой, так и с технической точки зрения с помощью автоматической сборки.

5.11 Загрузки не-сопровождающим (NMU)

Каждый пакет имеет одного или нескольких сопровождающих. Обычно это люди, которые работают над пакетом и выполняют загрузки новых версий. В некоторых ситуациях бывает полезно, если и другие разработчики смогут загрузить новую версию, например, если они хотят исправить ошибку в пакете, сопровождением которого они не занимаются, и если сопровождающий нуждается в помощи для решения проблем с пакетом. Такие загрузки называются *загрузки не-сопровождающими, Non-Maintainer Uploads (NMU)*.

5.11.1 Когда и как делать NMU

До выполнения NMU, ответьте на следующие вопросы:

- Предприняли ли вы NMU для того, чтобы помочь сопровождающему? Поскольку по поводу того, требуется ли сопровождающему помощь, могут возникнуть споры, существует очередь DELAYED, загрузка в которую позволяет сопровождающему отреагировать на вашу загрузку, а также позволяет другим оценить вклад вашей NMU-загрузки.
- Исправляет ли ваша NMU-загрузка ошибки? (Под "ошибками" подразумеваются любые ошибки, напр. пожелания по созданию пакета для версии из основной ветки разработки, но следует постараться минимизировать проблема сопровождающего.) Исправление косметических проблем или изменение стиля создания пакета (напр., переход с cdbs на dh) в NMU не желательны.
- Даёте ли вы достаточное количество времени сопровождающему? Когда в системе отслеживания ошибок было прислано сообщение об ошибке? Человек может быть занят одну или две недели, в этом не ничего необычного. Так ли серьёзная ошибка, что её необходимо исправить прямо сейчас, можно ли подождать ещё несколько дней?
- Насколько вы уверены в своих изменениях? Помните клятву Гиппократа: "Не навреди." Лучше оставить пакет даже с самой серьёзной ошибкой, чем применять к нему неработающую заплату, либо заплату, которая скорее скрывает ошибку, а не решает её. Если вы не уверены на 100% в том, что вы делаете, лучше будет попросить совета у других. Помните, что если вы что-то сломаете во время NMU, многие люди будут недовольны.
- Выразили ли вы ясно ваше намерение сделать NMU, по меньшей мере в системе отслеживания ошибок? Если вы не получили какого-либо ответа, то попытайтесь связаться с сопровождающим другими способами (напишите сообщение на адрес электронной почты сопровождающего, на его частный адрес электронной почты, через IRC).
- Если сопровождающий обычно активен и отзывчив, попытались ли вы с ним связаться? Вообще же предпочтительно, чтобы сопровождающие самостоятельно решали проблемы в своих пакетах, нужно дать им возможность проверить вашу заплату и при необходимости исправить её, поскольку они скорее всего лучше осведомлены о потенциальных проблемах, которые могут быть упущены в ходе NMU. Часто время каждого используется значительно лучше, если у сопровождающего будет возможность самому загрузить исправление.

Если вы выполняете NMU, вам для начала следует убедиться, что ваше намерение сделать NMU ясно и понятно. Затем вам необходимо отправить заплату с различиями между текущим пакетом и предполагаемой NMU в системе отслеживания ошибок. Сценарий **nmudiff** из пакета `devscripts` может вам в этом помочь.

Во время подготовки заплаты вам следует хорошо разобраться с практикой создания данного пакета. Если вы будете это учитывать, что это уменьшит бремя интеграции ваших изменений в обычный ход работы над пакетом и, таким образом, увеличит шансы на то, что ваши изменения будут в дальнейшем интегрированы. О практиках сборки данного пакета можно прочитать в файле `debian/README.source`.

Если у вас нет каких-либо веских причин не давать сопровождающему время для самостоятельной работы, вам следует дать последнему это время (например, загрузив пакет в очередь `DELAYED`). Для задержек рекомендуется использовать следующие значения:

- Загрузка исправлений критичных для выпуска ошибок, о которых было сообщено более 7 дней назад, сопровождающий не проявлял активности в течении 7 дней, нет никакого указания на то, что исправление находится в стадии подготовки: 0 дней
- Загрузка исправления только критичных для выпуска ошибок, о которых было сообщено более 7 дней назад: 2 дня

- Исправление только критичных для выпуска ошибок и важных ошибок: 5 дней
- Другие NMU: 10 дней

Эти задержки являются лишь примерами. В некоторых случаях, таких как загрузка исправлений безопасности, либо исправлений тривиальных ошибок, блокирующий перемещение пакета, желательно, чтобы исправленный пакет попал в `нестабильный выпуск` как можно скорее.

Иногда управляющие выпуском принимают решение разрешить NMU с более короткими задержками для некоторого подмножества ошибок (напр., для критичных для выпуска ошибок, о которых было сообщено более 7 дней назад). Кроме того, некоторые сопровождающие добавляют себя в список NMU с низким порогом и разрешают загружать NMU без какой-либо задержки. Но даже в этих случаях лучше всего дать сопровождающему несколько дней на самостоятельную работу и только потом осуществлять загрузку, особенно если заплата ранее не была доступна в системе отслеживания ошибок, либо если вы знаете, что сопровождающий в принципе активен.

После загрузки NMU вы становитесь ответственным за возможные проблемы, которые вы могли создать. Вам необходимо следить за пакетом (чтобы сделать это, подпишитесь на пакет в систем отслеживания пакетов).

Не разрешается бездумно выполнять NMU. Если вы выполняете NMU, и ясно, что сопровождающие активны и приняли бы заплату немного позже, либо если вы игнорируете рекомендации, приведённые в данном документе, ваша загрузка может привести к конфликту с сопровождающим. Вам всегда следует быть готовым к отстаиванию своего решения относительно любой загрузки NMU, которую вы выполняете.

5.11.2 NMU и файл `debian/changelog`

Как и во время любой другой загрузки (исходного кода) при выполнении NMU следует добавить запись в файл `debian/changelog`, сообщающую о том, что было изменено в данной загрузке. В первой строке записи должно быть явно указано, что это загрузка NMU, напр.:

```
  * Non-maintainer upload.
```

Присваивание версий при выполнении NMU разница для родных и неродных пакетов.

Если пакет является родным пакетом (без номера ревизии Debian в номере версии), версия должны совпадать с версией загрузки её последним сопровождающим, плюс `+nmu`*`X`*, где *`X`* представляет собой счётчик, начинающийся с `1`. Если последняя загрузка также была NMU, то счётчик следует увеличить. Например, если текущая версия пакета равна `1.5`, то загрузка NMU должна получить версию `1.5+nmu1`.

Если пакет не является родным пакетом, вам следует добавить минорный номер версии к части о ревизии Debian номера версии (та часть, которая идёт после последнего знака тире). Этот дополнительный номер должен начинаться с `1`. Например, если текущей версией является `1.5-2`, то загрузка NMU должна иметь версию `1.5-2.1`. Если в ходе NMU создаётся пакет для новой версии из основной ветки разработки, то номер ревизии Debian устанавливается в `0`, например, `1.6-0.1`.

В обоих случаях если последняя загрузка также была загрузкой NMU, счётчик должен быть увеличен. Например, если текущей версией является `1.5+nmu3` (родной пакет, для которого уже была выполнена загрузка NMU в прошлый раз), загрузка NMU должна получить версию `1.5+nmu4`.

Специальная схема версий требуется для того, чтобы избежать срыва работы сопровождающего, поскольку использование целого числа для ревизии Debian потенциально может привести к конфликту с загрузкой, которая уже находится в стадии подготовки самим сопровождающим в то время, как вы выполняете NMU, либо даже уже включена в очередь NEW на ftp. Кроме того, это полезно для визуального выделения того, что пакет в архиве был подготовлен не тем, кто является его официальным сопровождающим.

Если вы загружаете пакет в тестируемый или стабильный выпуски, иногда вам необходимо выполнить "разветвление" древа номера версии. Это делается, например, в случае подготовки загрузок с исправлениями безопасности. Для этого следует использовать версию вида `+deb`*`X`*`u`*`Y`*, где *`X`* представляет собой мажорный номер версии, а *`Y`* является счётчиком, начинающимся с `1`. Например, поскольку jessie (Debian 8) является стабильным выпуском, загрузка NMU для исправления проблем безопасности стабильного выпуска для пакета, имеющего версию `1.5-3` будет иметь версию `1.5-3+deb8u1`, а загрузка NMU с исправлением безопасности для stretch будет иметь версию `1.5-3+deb9u1`.

5.11.3 Использование очереди `DELAYED/`

Ожидание ответа на ваш запрос разрешения выполнить NMU не эффективно, так как это предполагает, что тот, кто выполняет NMU, должен будет отвлечься от проблемы, а затем снова вернутся к ней. Очередь

`DELAYED` (см. раздел 5.6.2, «Задержанные загрузки») позволяет разработчику, занимающемуся NMU, в то же самое время выполнять все необходимые задачи. Например, вместо того, чтобы сообщить сопровождающему о том, что вы собираетесь загрузить обновлённый пакет в течении 7 дней, вам следует загрузить пакет в `DELAYED/7` и сообщить сопровождающему, что у него имеется 7 дней для того, чтобы отреагировать на это. В течении этого времени сопровождающий может попросить вас ещё немного задержать загрузку, либо отменить её.

Очередь `DELAYED` не должна использоваться для дополнительного давления на сопровождающего. В частности, важно, что у вас имеется возможность отметить или задержать загрузку ещё немного до того момента, как срок изначальной задержки закончиться, так как сопровождающий не может сам отметить вашу загрузку.

Если вы выполняете NMU в `DELAYED`, а сопровождающий обновляет пакет до того, как закончится срок задержки, ваша загрузку будет отклонена, поскольку в архиве уже будет доступна более новая версия. В идеале сопровождающий позаботится о включении предлагаемых вами изменений (или по меньшей мере о решении проблем, для решения который предназначаются ваши изменения) в своей загрузке.

5.11.4 NMU с точки зрения сопровождающего

Когда кто-то выполняет NMU для вашего пакета, то это означает, что он хочет помочь вам поддерживать пакет в хорошей форме. Это позволяет пользователям быстрее получать исправленные пакеты. Вы можете попросить того, кто выполнил NMU, стать помощником сопровождающего для данного пакета. Когда кто-то выполнил NMU для вашего пакета, это вовсе не плохо; это лишь означает, что пакет интересен достаточному количеству людей, которые готовы работать над ним.

Для подтверждения NMU, добавьте предлагаемые изменения и запись журнала изменений в вашу собственную загрузку. Если вы не подтвердите NMU, добавив запись из журнала изменений NMU в ваш журнал изменений, сообщения об ошибках в системе отслеживания ошибок останутся закрытыми, но будут указаны как актуальные для версии пакета, которую загрузите вы.

5.11.5 NMU исходного кода и двоичные NMU (binNMU)

Полностью NMU называется *NMU исходного кода*. Есть и другой тип NMU, а именно *двоичные NMU* или *binNMU*. BinNMU также представляет собой загрузку пакета тем, что не является его сопровождающим. Тем не менее, это загрузка исключительно двоичного кода.

При обновлении библиотеки (либо другой зависимости) пакеты, использующие её вероятно потребуется собрать заново. Поскольку изменения исходного кода не требуются, используется тот же самый пакет с исходным кодом.

BinNMU обычно выполняются на buildd по инструкции wanna-build. В файл `debian/changelog` добавляется запись, объясняющая то, почему потребовалась загрузка, номер версии увеличивается в соответствии с тем, как это описано в разделе 5.10.2.1, «Повторная компиляция или только двоичные NMU». Эту запись не следует включать в следующую загрузку.

Buildd загружает пакеты для своей архитектуры в архив как двоичные загрузки. Строго говоря, они являются binNMU. Тем не менее, обычно они не называются NMU, они не добавляют запись в `debian/changelog`.

5.11.6 NMU и загрузки командой контроля качества

NMU представляют собой загрузки пакетов теми, кто не является сопровождающим этих пакетов. Существует также другой тип загрузок, когда загружаемый пакет не является вашим: это загрузки команды контроля качества. Загрузки, выполняемые командой контроля качества, являются загрузками осиротевших пакетов.

Загрузки, выполняемые командой контроля качества, очень похожи на загрузки обычных сопровождающих: они могут исправлять всё, что угодно, даже незначительные проблемы; присвоение номера версии происходит обычным путём, также нет необходимости использовать задержку при загрузке. Отличие состоит в том, что вы не указаны в полях `Maintainer` и `Uploader` данного пакета. Кроме того, запись журнала изменений при загрузке, выполняемой командой контроля качества, содержит специальную первую строку:

```
* QA upload.
```

Если вы хотите выполнить NMU, и у вас складывается впечатление, что сопровождающий не активен, вам следует проверить, является ли данный пакет осиротевшим пакетом (эта информация отображается на странице пакета в системе отслеживания пакетов). При выполнении первой загрузки командой контроля качества

в качестве сопровождающего устанавливается `Debian QA Group <packages@qa.debian.org>`. Для осиротевших пакетов, загрузка которых командой контроля качества ещё не выполнялась, в качестве сопровождающего имеют своего старого сопровождающего. Имеется список таких пакетов: https://qa.debian.org/orphaned.html.

Вместо выполнения загрузки командой контроля качестве, вы можете усыновить пакет и стать его сопровождающим. Вам не нужно разрешение кого бы то ни было для того, чтобы усыновить осиротевший пакет, вы можете установить себя его сопровождающим и загрузить новую версию (см. раздел 5.9.5, «Усыновление пакета»).

5.11.7 NMU и командные загрузки

Иногда вы занимаетесь исправлением и/или загрузкой пакета потому, что вы являетесь членом команды по созданию пакетов (команда использует адрес списка рассылки в качестве значения `Maintainer` или `Uploader`, см. раздел 5.12, «Совместное сопровождение»), но вы не хотите добавлять себя в поле `Uploaders`, поскольку вы не планируете постоянно участвовать в поддержке данного конкретного пакета. Если это соответствует политике вашей команды, вы можете выполнить обычную загрузку, не указывая себя в полях `Maintainer` или `Uploader`. В этом случае вам следует начать вашу запись в журнале изменения со следующей строки:

```
* Team upload.
```

5.12 Совместное сопровождение

Совместное сопровождение представляет собой термин, описывающий разделение обязанностей по сопровождению пакета Debian между несколькими людьми. Эта совместная работа почти всегда является отличной идеей, поскольку её результатом обычно является более высокое качество пакета и более быстрое исправление ошибок. Настоятельно рекомендуется, чтобы пакет с приоритетом `standard`, а также пакеты, являющиеся частью базового набора пакетов, имели несколько сопровождающих.

Обычно имеется главный сопровождающий и один или больше помощников. Главный сопровождающий является тем, чьё имя указано в поле `Maintainer` файла `debian/control`. Помощники сопровождающего — это все остальные сопровождающие, обычно они указаны в поле `Uploaders` файла `debian/control`.

В наиболее простом виде процесс добавления новых помощников сопровождающий крайне лёгок:

- Предоставьте помощнику сопровождающего доступ к исходному коду, из которого вы собираете ваш пакет. Обычно это предполагает, что вы используете систему контроля версий, поддерживающую работу в сети, такую как `CVS` или `Subversion`. Alioth (см. раздел 4.12, «Установка Debian FusionForge: Alioth») предоставляет помимо прочего и такие инструменты.
- Добавьте имя и адрес электронной почты вашего помощника в поле `Uploaders` из первого раздела файла `debian/control`.

  ```
  Uploaders: John Buzz <jbuzz@debian.org>, Adam Rex <arex@debian.org>
  ```

- Используя систему отслеживания пакетов (раздел 4.10, «Система отслеживания пакетов Debian»), помощники сопровождающего должны оформить подписку на соответствующий пакет с исходным кодом.

Другой формой совместного сопровождения является командное сопровождение, которое рекомендуется в случае если вы сопровождаете несколько пакетов вместе группой одних и тех же разработчиков. В этом случае на поля `Maintainer` и `Uploaders` каждого пакеты следует обратить особое внимание. Рекомендуется выбрать одну из двух следующих схем:

1. Поместите в поле `Maintainer` члена команды, который будет ответственен за данный пакет. В поле `Uploaders` поместите адрес списка рассылки, а также тех членов команды, которые также будут следить за пакетов.
2. Поместите адрес списка рассылки в поле `Maintainer`. В поле `Uploaders` поместите членов команды, которые будут следить за пакетом. В этом случае убедитесь, что список рассылки принимает сообщения об ошибках без какого-либо дополнительного взаимодействия (типа модерации для отправителей, не являющихся членами списка рассылки).

В любом случае не следует помещать всех членов команды в поле Uploaders. Это приводит к путанице — в обзорный список пакетов разработчика (см. раздел 4.11, «Обзор пакетов разработчика») попадают пакеты, о который данный разработчик фактически не заботится, и создаёт у пользователей ложное чувство того, что пакет хорошо сопровождается. По той же самой причине члены команды не должны добавлять себя в поле Uploaders только потому, что они загрузили данный пакет один раз, они могут выполнить "командную загрузку" (см. раздел 5.11.7, «NMU и командные загрузки»). И наоборот, не следует оставлять пакет только с одним адресом списка рассылки в поле Maintainer, когда поле Uploaders пусто.

5.13 Тестируемый выпуск

5.13.1 Основы

Пакеты обычно устанавливаются в тестируемый выпуск после того, как они прошли некоторую проверку в нестабильном выпуске.

На всех архитектурах должна быть доступна одна и та же версия пакета, у пакета не должно быть зависимостей, которые помешают установить этот пакет; также они не должны иметь на момент установки в тестируемый выпуск известных критичных для выпуска ошибок. Таким образом, тестируемый выпуск всегда должен быть близок кандидату на выпуск. Подробности см. ниже.

5.13.2 Обновления из нестабильного выпуска

Сценарии, обновляющие тестируемый выпуск, запускаются дважды каждый день, сразу же после установки обновлённых пакетов; эти сценарии имеют имя britney. Они создают файлы Packages для тестируемого выпуска, но делают они это разумным способом; они пытаются избежать противоречивости и использовать только те пакеты, которые не имеют ошибок.

Включение пакета из нестабильного выпуска выполняется в соответствии со следующими условиями:

- Пакет должен быть доступен в нестабильном выпуске в течении 2, 5 или 10 дней в зависимости от его срочности (высокая, средняя или низкая). Заметьте, что учитывается также и срочность прошлых загрузок, точнее учитывается наивысшая срочность загрузки с момента последнего перехода пакета в тестируемый выпуск;
- Пакет не должен иметь новых ошибок, критичных для выпуска (то есть, критичных для выпуска ошибок в версии пакета, доступной в нестабильном выпуске, но отсутствующих в версии из тестируемого выпуска);
- Пакет должен быть доступен на всех архитектурах, на которых он ранее был собран в нестабильном выпуске. dak ls может помочь проверить эту информацию;
- Пакет не должен ломать зависимости какого-либо уже доступного в тестируемом выпуске пакета;
- Пакеты, от которых зависит данный пакет, должны либо быть доступны в тестируемом выпуске, либо должны быть приняты в тестируемый выпуск одновременно с этим пакетом (эти пакеты будут приняты в случае, если они удовлетворяют всем необходимым критериям);
- Фаза проекта. Напр., автоматические переходы отключаются во время *заморозки* тестируемого выпуска.

Чтобы узнать, продвигается пакет к переходу в тестируемый выпуск или нет, см. вывод сценария тестируемого выпуска на веб-странице тестируемого выпуска, либо используйте программу **grep-excuses**, которая является частью пакета devscripts. Данная утилита легко может использоваться в crontab(5) для того, чтобы у вас всегда имелась информация о продвижении ваших пакетов в тестируемый выпуск.

Файл update_excuses не всегда содержит точную причину того, почему пакет был отклонён; вам может потребоваться выяснить это самостоятельно, проверяя, что может сломаться из-за добавления вашего пакета. Веб-страница тестируемого выпуска содержит немного больше информации об обычных проблемах, которые могут вызывать подобные затруднения.

Бывает так, что некоторые пакет никогда не включаются в тестируемый выпуск из-за того, что набор взаимных зависимостей слишком сложен и не может быть разобран сценариями. Подробности см. ниже.

Некоторый дополнительный анализ зависимостей доступен по адресу https://release.debian.org/migration/ — но предупреждаем, эта страница также содержит сборочные зависимости, которые не были рассмотрены britney.

5.13.2.1 Устаревание

Для сценария миграции в тестируемый выпуск устаревание означает следующее: в нестабильном выпуске имеются разные версии для выпускаемых архитектур (за исключением архитектур из fuckedarches; fuckedarches представляет собой список архитектур, которые не поддерживаются (в update_out.py), но в настоящее время он пуст). Устаревание не касается архитектур данного пакета, имеющихся в тестируемом выпуске.

Рассмотрим следующий пример:

	alpha	**arm**
testing	1	-
unstable	1	2

Пакет является устаревшим на архитектуре alpha в нестабильном выпуске, и он не попадёт в тестируемый выпуск. Удаление пакета не поможет, пакет всё равно считается устаревшим на архитектуре alpha, и он не будет перемещён в тестируемый выпуск.

Тем не менее, если сопровождающий ftp удаляет пакет в нестабильном выпуске (здесь — на архитектуре arm):

	alpha	**arm**	**hurd-i386**
testing	1	1	-
unstable	2	-	1

В этом случае пакет считается обновлённым на всех выпускаемых архитектурах в нестабильном выпуске (дополнительная архитектура hurd-i386 не имеет значения, поскольку она не является выпускаемой архитектурой).

Время от времени возникает вопрос о том, можно ли разрешить переход пакетов, которые ещё не были собраны на всех архитектурах. Нет. Просто нет и всё. (За исключением случая, если вы сопровождаете glibc или что-то подобное.)

5.13.2.2 Удаление из тестируемого выпуска

Иногда какой-то пакет удаляется для того, чтобы был осуществлён переход другого пакета. Это происходит только для того, чтобы позволить *другому* пакету осуществить переход в случае, если он уже готов. Допустим, напр., что пакет a не может быть установлен вместе с новой версией пакета b; тогда пакет a может быть удалён для того, чтобы был осуществлён переход пакета b.

Конечно, имеется и другая причина для удаления пакета из тестируемого выпуска. Он имеет слишком много ошибок (хотя даже одной критичной для выпуска ошибки достаточно).

Более того, если пакет был удалён из нестабильного выпуска, и ни один пакет в тестируемом выпуске не зависит от него, то этот пакет будет автоматически удалён.

5.13.2.3 Круговые зависимости

Ситуация, которая не может быть правильно обработана britney, заключает в том, что пакет a зависит от новой версии пакета b, и наоборот.

Пример:

	testing	**unstable**
a	1; depends: b=1	2; depends: b=2
b	1; depends: a=1	2; depends: a=2

Ни пакет a, ни пакет b не рассматриваются для обновления.

В настоящее время такая ситуация требует вмешательства выпускающей команды. Свяжитесь с ними, отправив сообщение по адресу debian-release@lists.debian.org, если такая ситуация возникла для одного из ваших пакетов.

5.13.2.4 Влияние пакета в тестируемом выпуске

Вообще же, статус пакета в тестируемом выпуске ничего не значит для перехода следующей версии этого пакета из нестабильного выпуска в тестируемый, с двумя исключениями. Во-первых, если у пакета уменьшается количество критичных для выпуска ошибок, то он может осуществить переход даже в том случае, если у него всё ещё остаются критичные для выпуска ошибки. Во-вторых, если версия пакета в тестируемом выпуске не синхронизирована на разных архитектурах. Тогда любая архитектура может быть обновлена до версии пакета с исходным кодом; тем не менее, это может произойти только в том случае, если пакет ранее был перемещён вручную, данная архитектура входит в список fuckedarches, либо для данной архитектуры не было двоичного пакета в нестабильном выпуске во время перехода в тестируемый выпуск.

Короче говоря, это означает следующее: единственный фактор, на который влияет нахождение пакета в тестируемом выпуске, состоит в том, что новая версия этого пакета может быть проще добавлена.

5.13.2.5 Подробности

Если вам интересны подробности, то britney работает следующим образом:

Просматриваются пакеты на предмет выявления того, являются они корректными кандидатами или нет. Это даёт нам список оснований для отказа в обновлении. Наиболее частыми основаниями того, почему пакет не рассматривается для обновления, являются то, что он слишком нов, имеет критичные для выпуска ошибки, устарел на каких-то архитектурах. Для этой части britney у управляющих выпуском имеются различные инструменты, называемые подсказками (см. ниже), которые используются для того, чтобы заставить britney рассмотреть данный пакет.

Далее начинается более сложная часть. Britney пытается обновить тестируемый выпуск путём установки корректных кандидатов. Для этого britney пытается добавить каждого корректного кандидата в тестируемый выпуск. Если число неустанавливаемых пакетов в тестируемом выпуске не увеличивается, то пакет принимается. С этой точки зрения принятый пакет считается частью тестируемого выпуска, такой частью, что все последующий проверки установок будут включать в себя этот пакет. Подсказки выпускающей команды обрабатываются после или до этой основной работы в зависимости от типа подсказок.

Если вы хотите ознакомиться с дополнительными подробностями, вы можете прочитать https://ftp-master.debian.org/testing/update_output/.

Подсказки доступны в https://ftp-master.debian.org/testing/hints/, также там вы можете найти описание. При помощи подсказок выпускающая команда Debian может блокировать или разблокировать пакеты, замедлять или ускорять переход пакетов в тестируемый выпуск, удалять пакеты из тестируемого выпуска, принимать загрузки в testing-proposed-updates, либо изменять срочность загрузки.

5.13.3 Прямые обновления тестируемого выпуска

Тестируемый выпуск получает пакеты из нестабильного выпуска в соответствии с описанными ранее правилами. Тем не менее, в некоторых случаях необходимо загружать пакеты, собранные только для тестируемого выпуска. Для этого вы можете использовать загрузку в testing-proposed-updates.

Помните, что пакеты, загруженные туда, не обрабатываются автоматически, они должны пройти через руки управляющего выпуском. Поэтому для того, чтобы загружать свои пакеты туда, вам следует иметь для этого хорошее основание. Для того, чтобы знать, что управляющие выпуском считаются хорошим основанием, вам следует прочитать инструкции, которые они регулярно публикуют в debian-devel-announce@lists.debian.org.

Вам не следует загружать в testing-proposed-updates, если вы можете обновить ваши пакеты через нестабильный выпуск. Если вы не можете этого сделать (например, из-за того, что у вас имеется более новая версия в нестабильном выпуске), вы можете использовать эту возможность. Однако рекомендуется сначала попросить на это разрешение у управляющего выпуском. Даже если пакет заморожен обновления через нестабильный выпуск всё равно возможны, если загрузка через нестабильный выпуск не тянет за собой какие-либо новые зависимости.

Номера версий обычно выбираются путём добавления +deb*X*u*Y*, где *X* представляет собой мажорный номер выпуска Debian, а *Y* является счётчиком, начинающимся с 1. Напр., 1:2.4.3-4+deb8u1.

Убедитесь, что в вашей загрузке вы ничего не пропустили из следующего списка:

- Убедитесь, что ваш пакет действительно должен пройти через testing-proposed-updates, и что он не может пройти через нестабильный выпуск;
- Убедитесь, что вы включили в пакет лишь минимальное число изменений;
- Убедитесь, что вы включили соответствующее объяснение в журнал изменений пакета;

- Убедитесь, что вы указали кодовое-имя тестируемого выпуска (напр., `stretch`) в качестве целевого выпуска;
- Убедитесь, что вы собрали и протестировали ваш пакет в `тестируемом`, а не в `нестабильном` выпуске;
- Убедитесь, что номер версии пакета выше номера версии, входящей в `тестируемый` выпуск и в `testing-proposed-updates`, а также ниже, чем в `нестабильном` выпуске;
- После загрузки и успешной сборки на всех платформах, свяжитесь с выпускающей командой по адресу debian-release@lists.debian.org и попросите их одобрить вашу загрузку.

5.13.4 Часто задаваемые вопросы

5.13.4.1 Что такое критичные для выпуска ошибки, как производится их подсчёт?

Все ошибки, имеющие высокую важность, по умолчанию считаются критичными для выпуска; в настоящее время это ошибки с важностью `critical`, `grave` и `serious`.

Предполагается, что такие ошибки влияют на шансы того, будет пакет выпущен в составе `стабильного` выпуска Debian или нет. В общем случае, если пакет имеет открытые сообщения о критичных для выпуска ошибках, он не попадёт в `тестируемый` выпуск и, соответственно, не будет выпущен в составе `стабильного` выпуска.

Число ошибок в пакетах `нестабильного` выпуска представляет собой все критичные для выпуска ошибки, которые помечены как применимые к комбинациям *`пакет`*/*`версия`*, доступным в нестабильном выпуске для выпускаемой архитектуры. Число ошибок в пакетах `тестируемого` выпуска определяется аналогичным образом.

5.13.4.2 Как установка какого-то пакета в `тестируемый` выпуск может сломать другие пакеты?

Структура архивов выпусков такова, что они могут содержать только одну версию пакета; пакет определяется его именем. Поэтому, когда пакет с исходным кодом `acmefoo` устанавливается в `тестируемый` выпуск вместе с соответствующими двоичными пакетами `acme-foo-bin`, `acme-bar-bin`, `libacme-foo1` и `libacme-foo-dev`, старая версии пакетов удаляются.

Тем не менее, старая версия может предоставлять двоичный пакет со старой библиотекой с тем же именем, такой как `libacme-foo0`. Удаление старого пакета `acmefoo` приведёт к удалению `libacme-foo0`, что в свою очередь приведёт к удалению всех зависящих от неё пакетов.

Очевидно, эта ситуация касается пакетов, которые в своих разных версиях предоставляют разные наборы двоичных пакетов (в основном библиотеки). Тем не менее, она также касается пакетов, которые имеют зависимости от конкретных версий, задаваемые через ==, <= или <<.

Если набор двоичных пакетов, предоставляемый пакетом с исходным кодом, меняется подобным образом, все пакеты, которые зависят от старых двоичных пакетов должны быть обновлены таким образом, что они должны зависеть от новых двоичных пакетов. Поскольку установка такого пакета с исходным кодом в `тестируемый` выпуск ломает все пакеты, которые зависят от него в `тестируемом` выпуске, следует быть немного более внимательным: все зависящие пакеты сами должны быть обновлены и готовы к установке, так чтобы они не сломалось, и когда всё будет готово, обычно требуется ручное вмешательство управляющего выпуском или его ассистента.

Если у вас имеются подобные проблемы со сложной группой пакетов, свяжитесь с debian-devel@lists.debian.org or debian-release@lists.debian.org for help.

Глава 6

Лучшие практики создания пакетов

Качество Debian во многом определяется Политикой Debian, определяющей точные базовые требования, которым должны соответствовать все пакеты Debian. Помимо этого существует и общая история, выходящая за пределы Политики Debian, которая была накоплена в течении многих лет работы по созданию пакетов. Многие очень талантливые люди создали превосходные инструменты, инструменты, которые помогут вам, сопровождающему Debian, создавать превосходные пакеты и сопровождать их.

Эта глава содержит описание некоторых лучших практик для разработчиков Debian. Все рекомендации являются только рекомендациями, а не требованиями или правилами. Они представляют собой лишь субъективные подсказки, советы и указания, позаимствованные у других разработчиков Debian. Выбирайте то, что лучше всего подходит именно вам.

6.1 Лучшие практики для `debian/rules`

Следующие рекомендации применимы к файлу `debian/rules`. Поскольку файл `debian/rules` контролирует процесс сборки и выбирает файлы, которые войдут затем в пакет (напрямую или косвенно), он является тем файлом, на который обычно сопровождающие тратят больше всего времени.

6.1.1 Сценарии-помощники

Основная причина использования сценариев-помощников в файле `debian/rules` заключается в том, что последние позволяют сопровождающим распространять общую логику на множество пакетов и использовать её. Рассмотрим для примера вопрос установки пунктов меню: вам необходимо поместить файл в `/usr/share/menu` (или, если это требуется, `/usr/lib/menu` для выполняемых двоичных файлов меню), а также добавить команды сценариям сопровождающего для регистрации и отмены регистрации этих пунктов меню. Поскольку это довольно частая задача, зачем каждому сопровождающему переписывать всё это самостоятельно, а иногда и с ошибками? Кроме того, допустим, что каталог меню изменился, в этом случае пришлось бы изменить каждый пакет.

Сценарии-помощники могут позаботиться об этих проблемах. Допустим, вы соблюдаете конвенции, ожидаемые сценарием-помощником, а последний заботится обо всех деталях. Если политика изменится, то это изменение может быть отражено в сценарии-помощнике; тогда пакеты нужно будет лишь собрать заново с новой версией помощника без каких-либо изменений вручную.

Раздел A, «Обзор инструментов Debian для сопровождающего» содержит описание различных помощников. Наиболее часто используемой и лучшей (по нашему мнению) системой-помощником является `debhelper`. Предшествующие системы-помощники, такие как `debmake`, были монолитны: вы не могли выбрать только ту часть помощника, которая кажется вам полезной, вам приходилось использовать этот помощник для всего сразу. Тем не менее, `debhelper` представляет собой ряд отдельных небольших программ **dh_***. Например, **dh_installman** устанавливает и сжимает страницы руководства, **dh_installmenu** устанавливает файлы меню и так далее. Таким образом, этот помощник предлагает достаточный уровень гибкости, позволяющий использовать небольшие сценарии-помощники там, где это подходит, а также команды, добавленные в файл `debian/rules` вручную.

Вы можете начать своё знакомство с `debhelper` с чтения debhelper(1) и изучения примеров, которые поставляются вместе с содержащим этот помощник пакетом. **dh_make** из пакета `dh-make` (см. раздел A.3.2, «`dh-make`»), может использоваться для перевода ванильного пакета с исходным кодом в `debhelper`ианизированый пакет. Этот короткий путь не должен, однако, создать у вас впечатление, что вам не нужно беспокоиться о

понимании каждого отдельного помощника **dh_***. Если вы собираетесь использовать помощник, вам следует потратить время на изучение того, как этот помощник используется, и изучить его ожидания и поведение.

6.1.2 Отделение ваших заплат в несколько файлов

Большие, сложные пакеты могут содержать множество ошибок, с которыми вам предстоит иметь дело. Если вы исправляете несколько ошибок напрямую в исходном коде, и при этом вы недостаточно внимательны, дифференцировать различные заплаты, которые вы применили, может стать довольно трудной задачей. Пакет может превратиться в бардак, когда вам придётся обновить его для новой версии из основной ветки разработки, содержащей некоторые (но не все) применённые вами исправления. Вы не можете просто взять полный набор diff'ов (напр., из `.diff.gz`) и определить то, какой набор заплат представляет собой тот единый компонент кода, содержащий исправление ошибок для основной ветки разработки.

К счастью, с форматом исходного кода "3.0 (quilt)" теперь возможно хранить заплаты отдельно, не меняя файл `debian/rules` для настройки системы заплат. Заплаты хранятся в `debian/patches/`, и когда пакет с исходным кодом распаковывается, заплаты, указанные в `debian/patches/series`, применяются автоматически. Как предполагается из самого названия формата, заплаты могут обрабатываться командой **quilt**.

При использовании более старого формата пакетов с исходным кодом "1.0" также можно выделить заплаты, но тогда необходимо использовать выделенную систему заплат: файлы заплат поставляются в файле заплат Debian (`.diff.gz`), который обычно располагается в каталоге `debian/`. Единственное отличие состоит в том, что в таком случае заплаты не применяются непосредственно командой **dpkg-source**, но правилом `build` файла `debian/rules`, через зависимость от правила `patch`. И наоборот, они отменяются в правиле `clean`, через зависимость от правила `unpatch`.

Рекомендованным инструментом для этого является команда **quilt**. Она делает всё вышеперечисленное, также она позволяет обрабатывать цепочки заплат. Дополнительную информацию смотрите в пакете `quilt`.

Для обработки заплат существуют и другие инструменты, такие как **dpatch**, и система заплат, интегрированная в `cdbs`.

6.1.3 Множественные двоичные пакеты

Один пакет с исходным кодом обычно используется для сборки нескольких двоичных пакетов, либо для предоставления нескольких разновидностей одного и того же ПО (напр., пакет с исходным кодом `vim`), либо для создания нескольких небольших пакетов вместо одного большого (напр., так пользователь может установить только необходимое ему подмножество пакетов и таким образом сохранить место на диске).

Со вторым случае можно легко управиться при помощи настроек в файле `debian/rules`. Вам нужно лишь переместить соответствующие файлы из каталога сборки во временные деревья пакета. Вы можете сделать это при помощи команд **install** или **dh_install** из пакета `debhelper`. Не забудьте проверить разные изменения различных пакетов на предмет того, что вы правильно установили межпакетные зависимости в файле `debian/control`.

Первый случай немного сложнее, поскольку он предполагает множественные перекомпиляции одного и того же ПО, но с различными опциями настройки. Пакет с исходным кодом `vim` представляет собой пример того, как можно справиться с этим, используя настроенный вручную файл `debian/rules`.

6.2 Лучшие практики для `debian/control`

Следующие практики относятся к файлу `debian/control`. Они дополняют Политику описаний пакетов.

Описание пакета, как оно определено в соответствующем поле файла `control`, содержит резюме пакета и длинное описание этого пакета. раздел 6.2.1, «Общие принципы описания пакетов» описывает общие принципы обеих этих частей описания пакета. Далее, раздел 6.2.2, «Резюме пакета или короткое описание» содержит принципы, относящиеся к резюме, а раздел 6.2.3, «Длинное описание» представляет собой принципы, относящиеся к описанию.

6.2.1 Общие принципы описания пакетов

Описание пакета должно быть написано для среднего пользователя, среднего человека, который будет использовать и извлекать пользу из данного пакета. Например, пакеты для разработки предназначены для разработчиков, и могут быть описаны техническим языком. Описания приложений общего назначения, таких

как текстовые редакторы, должны быть написаны для менее подкованного в техническом плане пользователя.

Наш обзор описаний пакетов привёл нас к заключению, что большая часть описаний пакетов являются техническими, то есть они не написаны так, чтобы быть понятными технически не подкованным пользователям. Если ваш пакет предназначен не только для технических пользователей, это является проблемой.

Как следует писать описания для технически не подкованных пользователей? Избегайте жаргона. Избегайте указания на другие приложения или наборы приложений, которые могут быть незнакомы пользователю — GNOME или KDE это нормально, так как пользователи вероятно знакомы с этими терминами, но термин GTK+ скорее всего им не знаком. Попытайтесь вообще не допускать какого-либо знания. Если вам нужно использовать технические термины, определите их.

Будьте объективны. Описания пакетов — это не то место, где следует пропагандировать свой пакет, не важно, насколько вы его любите. Помните, что читатель может и не заботиться о тех же вещах, что и вы.

Указания на названия любых других пакетов ПО, названия протоколов, стандартов или спецификаций должны иметь каноническую форму, если таковая существует. Например, используйте X Window System, X11 или X; но не X Windows, X-Windows или X Window. Используйте GTK+, а не GTK или gtk. Используйте GNOME, а не Gnome. Используйте PostScript, а не Postscript или postscript.

Если у вас возникли проблемы с написанием описания, вы можете отправить его по адресу debian-l10n-english@lists.debian.org с запросом помощи.

6.2.2 Резюме пакета или короткое описание

Политика говорит, что строка резюме (короткое описание) должна быть точной, не должна повторять имя пакета, но также должна быть информативной.

Резюме представляет собой фразу, описывающую пакет, а не полное предложение, поэтому пунктуация, характерная для предложений, тут неуместна: дополнительные заглавные буквы или точка в конце (как знак полной остановки) не нужны. Также следует пропустить любой начальный неопределённый или определённый артикль — ”a”, ”an” или ”the”. Таким образом, например:

```
Package: libeg0
Description: exemplification support library
```

Технически это именная фраза без артиклей в противоположность глагольной фразе. Хорошая эвристика состоит в том, что должно быть возможно подставить *имя* пакета вместо его *резюме* в следующую формулу:

Пакет *имя пакета* предоставляет {a,an,the,some} *резюме пакета*.

Наборы связанных пакетов могут использовать альтернативную схему, которая разделяет резюме на две части, на, во-первых, описание всего набора и, во-вторых, резюме роли конкретного пакета в этом наборе:

```
Package: eg-tools
Description: simple exemplification system (utilities)

Package: eg-doc
Description: simple exemplification system - documentation
```

Эти резюме следуют изменённой формуле. В ней пакет ”*имя пакета*” имеет резюме ”*набор пакетов* (*роль*)” или ”*набор пакетов* - *роль*”, элементы резюме должны быть сформулированы так, чтобы они подходили под следующую формулу:

Пакет *имя пакета* предоставляет {a,an,the} *роль* для *набора пакетов*.

6.2.3 Длинное описание

Длинное описание является первичной информацией, доступной пользователю о пакете до того, как он его установит. Оно должно предоставлять всю информацию, которая необходима для того, чтобы пользователь мог решить устанавливать ему этот пакет или нет. Допускайте, что пользователь уже прочитал резюме пакета.

Длинное описание должно состоять из полных и законченных предложений.

Первый абзац длинного описания должен содержать ответ на следующие вопросы: кто делает этот пакет? какие задачи он помогает решить пользователю? Важно описать это без использования технических терминов, если, конечно, аудиторией данного пакета не являются с необходимостью технически подкованные пользователи.

Следующие абзацы должны давать ответы на следующие вопросы: Почему я как пользователь нуждаюсь в этом пакете? Какими другими возможностями обладает данный пакет? Каковы его выдающиеся особенности

и недостатки по сравнению с другими пакетами (напр., если вам нужен X, используйте вместо него Y)? Связан ли этот пакет с другими пакетами так, что эта связь не обрабатывается менеджером пакетов (напр., это клиент для сервера foo)?

Будьте внимательны, избегайте орфографических и грамматических ошибок. Проверьте описание на ошибки. И **ispell**, и **aspell** имеют специальные режимы для проверки файлов `debian/control`:

```
ispell -d american -g debian/control
```

```
aspell -d en -D -c debian/control
```

Обычно пользователи ожидают найти в описании пакета ответы на следующие вопросы:

- Что этот пакет делает? Если он является дополнением для другого пакета, то в описании следует поместить краткое описание того пакета, дополнением которого является данный пакет.
- Почему я хочу получить этот пакет? Ответ на этот вопрос связан с ответом на предыдущий вопрос, но это разные вопросы (это клиент электронной почты; он классный, быстрый, работает с PGP и LDAP, а также с IMAP, он имеет X, Y и Z).
- Если этот пакет не должен быть установлен напрямую, но тянется за другим пакетом, это обстоятельство должно быть явно указано.
- Если пакет является `экспериментальным`, либо имеются другие причины, по которым этот пакет не следуем использовать, если имеются другие пакеты, которые следует использовать вместо данного, это также должно быть отдельно указано.
- Чем этот пакет отличается от других таких же пакетов? Является он этот пакет лучшей реализацией? имеет больше возможностей? имеет другие возможности? Почему я должен выбрать этот пакет.

6.2.4 Домашняя страница основной ветки разработки

Мы рекомендуем вам добавить ссылку на домашнюю страницу пакета в поле `Homepage` раздела `Source` в файле `debian/control`. Добавление этой информации в само описание пакета считается устаревшей нормой.

6.2.5 Размещение системы контроля версий

В файле `debian/control` имеются дополнительные поля для указания размещения системы контроля версий.

6.2.5.1 Vcs-Browser

Значением этого поля должна быть ссылка вида `https://`, указывающая на копию репозитория системы управления версиями, используемой для сопровождения данного пакета, которую можно просмотреть в веб-браузере, если таковая имеется.

Подразумевается, что данная информация будет полезна для конечного пользователя, который захочет ознакомиться с последней работой, связанной с этим пакетом (напр., захочет найти заплату для исправления ошибки, обозначенной с системе отслеживания ошибок тегом `pending`).

6.2.5.2 Vcs-*

Значением этого поля должна быть строка, однозначно определяющая размещение репозитория системы контроля версий, используемого для сопровождения данного пакета, если таковая имеется. `*` определяет систему управления версиями; в настоящее время системой отслеживания пакетов поддерживаются следующие системы: `arch`, `bzr` (Bazaar), `cvs`, `darcs`, `git`, `hg` (Mercurial), `mtn` (Monotone), `svn` (Subversion). Разрешается добавлять разные поля систем управления версиями для одного и того же пакета: они будут показаны в веб-интерфейсе системы отслеживания пакетов.

Подразумевается, что эта информация будет полезна пользователю, которых осведомлён о принципах работы данной системы управления версиями и желает собрать текущую версию пакета из исходного кода, размещённого там. Другие способы использования этой информации могут включать автоматическую сборку последней версии данного пакета из системы управления версиями. Для этого размещение, указанное

в данном поле, не должно иметь привязку к версии и должно указывать на основную ветку (это касается систем управления версиями, поддерживающих эту возможность). Кроме того, указываемое размещение должно быть доступно конечному пользователю; для выполнения этого требования может потребоваться указание на анонимный доступ к репозиторию вместо указания версии, доступной через SSH.

В следующем примере демонстрируется пакет `vim` в соответствующем поле которого указан репозиторий Subversion. Заметьте, что URL приведён по схеме `svn://` (а не по схеме `svn+ssh://`), а также то, что указана ветвь `trunk/`. Также демонстрируется использование полей `Vcs-Browser` и `Homepage`, описанных выше.

```
Source: vim
Section: editors
Priority: optional
<snip>
Vcs-Svn: svn://svn.debian.org/svn/pkg-vim/trunk/packages/vim
Vcs-Browser: https://svn.debian.org/wsvn/pkg-vim/trunk/packages/vim
Homepage: http://www.vim.org
```

6.3 Лучшие практики для `debian/changelog`

Следующие практики дополняют Политику о файлах изменений.

6.3.1 Написание полезных пунктов файла изменений

Записи в файле изменений ревизии пакета описывают изменения данной ревизии и только их. Сосредоточьтесь на описании существенных и заметных для пользователя изменений, которые были произведены с момента выпуска последней версии.

Обратите внимание на то, *что* было изменено — кто, как и когда обычно менее важно. Указав это, не забудьте вежливо указать людей, которые оказали заметную помощь в создании пакета (напр., тех, что выслал заплаты).

Нет необходимости указывать тривиальные и очевидные изменения. Также вы можете объединять несколько изменений в одну запись. С другой стороны, не будьте чересчур кратки, если внесли крупное изменение. Будьте особенно ясны, если были произведены изменения, которые оказывают влияние на поведение программы. Для указания дальнейших объяснений используйте файл `README.Debian`.

Используйте общепринятый вариант английского языка, чтобы большинство читателей смогли вас понять. Избегайте аббревиатур, технического языка и жаргона, когда вы описываете изменения, закрывающие ошибки, особенно это касается ошибок, о которых сообщили пользователи, и которые не кажутся вам в каком-то особом смысле технически сложными. Будьте вежливы, не ругайтесь.

Иногда желательно указать в начале записи об изменениях имена файлов, которые были изменены. Тем не менее, нет необходимости явно указывать каждый изменённый файл, особенно если изменения были небольшими или повторяющимися. Используйте шаблоны с метасимволами.

Если вы ссылаетесь на ошибки, не предполагайте у пользователей какого-либо знания. Сообщите, в чём заключалась проблема, как она была исправлена, и добавьте строку closes: #nnnnn. Дополнительную информацию см. в разделе 5.8.4, «Когда ошибки исправляются путём новых загрузок».

6.3.2 Распространённые неправильные представления о записях об изменениях

Записи об изменениях **не** должны описывать общие проблемы при создании пакета (Эй, если вы ищете foo.conf, он находится в /etc/бла-бла-бла/.), поскольку предполагается, что администраторы и пользователи по меньшей мере отдалённо знакомы с тем, как такие вещи обычно улаживаются в системах Debian. Сообщите, тем не менее, о том, что вы изменили расположение файла настройки, если вы произвели такое изменение.

Единственными закрываемыми с помощью записей об изменениях ошибками должны быть те, которые фактически исправляются именно в этой ревизии пакета. Закрытие несвязанных ошибок в журнале изменений является плохой практикой. См. раздел 5.8.4, «Когда ошибки исправляются путём новых загрузок».

Записи об изменениях **не** должны использоваться для случайных дискуссий с теми, кто сообщил об ошибках (Я не наблюдаю ошибки сегментирования, когда запускаю foo с опцией bar; вышлите дополнительную информацию), общих утверждений о жизни, Вселенной и вообще (извините, загрузка этой версии заняла так много времени, но у меня была простуда), или просьб о помощи (список ошибок этого пакета очень велик, пожалуйста, помогите мне с этим). Скорее всего всё это не будет замечено целевой аудиторией, но может

надоедать тем людям, которые хотят читать информацию о фактических изменениях в пакете. Подробную информацию о том, как использовать систему отслеживания ошибок см. в разделе 5.8.2, «Ответ на ошибки».

Традиционно ошибки, исправленные в загрузках, которые произведены не сопровождающими, упоминаются в первой записи об изменениях той загрузки, которая произведена собственно сопровождающим. Поскольку сейчас у нас имеется система отслеживания версий, достаточно сохранить записи об изменениях, сделанных несопровождающими, и просто упомянуть этот факт в вашей записи об изменениях.

6.3.3 Общие ошибки в записях об изменениях

Следующие примеры демонстрируют некоторых общие ошибки или примеры плохого стиля в записях об изменениях.

```
* Fixed all outstanding bugs.
```

Очевидно, эта запись не сообщает читателям ничего полезного.

```
* Applied patch from Jane Random.
```

Что это была за заплата?

```
* Late night install target overhaul.
```

Что это была за переделка? Упоминание поздней ночи предполагается как сообщение о том, что этому коду не следует доверять?

```
* Fix vsync FU w/ ancient CRTs.
```

Слишком много акронимов, и мне совсем не ясно, что, ёбтвмть (ой, мат!), это вообще такое и что было исправлено.

```
* This is not a bug, closes: #nnnnnn.
```

Во-первых, вовсе не нужно загружать пакет, чтобы сообщить эту информацию; вместо этого используйте систему отслеживания ошибок. Во-вторых, отсутствует объяснение того, почему данный отчет об ошибке ошибкой не является.

```
* Has been fixed for ages, but I forgot to close; closes: #54321.
```

Если по какой-то причине вы не указали ранее в записи об изменениях номер ошибки, это не проблема, просто закройте ошибку обычным способом в системе отслеживания ошибок. Не нужно трогать файл журнала, если описание исправления уже внесено в него (это относится также к исправлениям, внесённым разработчиками основной ветки разработки и сопровождающими, вам не нужно указывать в журнале ошибки, которые были исправлены ими давным давно).

```
* Closes: #12345, #12346, #15432
```

Где описание? Если вы не можете придумать описание, начните со вставки заголовка каждой отдельной ошибки.

6.3.4 Дополнение журналов файлами `NEWS.Debian`

Важные новости об изменениях в пакете можно также поместить в файл `NEWS.Debian`. Новости будут отображены такими инструментами как `apt-listchanges` до всех остальных журналов изменений. Это является предпочтительным способом сообщения пользователям о существенных изменениях в пакете. Это лучше, чем использование примечаний `debconf`, поскольку этот способ менее надоедлив и пользователь может вернуться и просмотреть файл `NEWS.Debian` после установки. И это лучше, чем указание крупных изменений в файле `README.Debian`, поскольку пользователь легко может не заметить вашего сообщения.

Формат этот файла такой же как и формат файла журнала изменений, но забудьте о звёздочках и описывайте каждую новость в отдельном полном параграфе, если это необходимо, вместо того, чтобы давать более краткие резюме, которые подходят для журнала изменений. Хорошо бы пропустить ваш файл через `dpkg-parsechangelog` для проверки форматирования, поскольку в отличии от журнала изменений он не будет проверен автоматически во время сборки. Ниже приведён пример настоящего файла `NEWS.Debian`:

```
cron (3.0pl1-74) unstable; urgency=low

    The checksecurity script is no longer included with the cron package:
    it now has its own package, checksecurity. If you liked the
    functionality provided with that script, please install the new
    package.

 -- Steve Greenland <stevegr@debian.org>  Sat,  6 Sep 2003 17:15:03 -0500
```

Файл `NEWS.Debian` устанавливается как `/usr/share/doc/`*`пакет`*`/NEWS.Debian.gz`. Он сжимается, и он всегда имеет данное имя даже в родных пакетах Debian. Если вы используете `debhelper`, `dh_installchange` установит файлы `debian/NEWS` за вас.

В отличии от файлов журналов вам не нужно обновлять файлы `NEWS.Debian` в каждом выпуске. Обновляйте их только если имеется что-то действительно заслуживающее упоминания, что должно быть известно пользователю. Если у вас нет никаких новостей, не нужно добавлять файл `NEWS.Debian` в ваш пакет. Отсутствие новостей — уже хорошие новости!

6.4 Лучшие практики для сценариев сопровождающих

Сценарии сопровождающих включают в себя файлы `debian/postinst`, `debian/preinst`, `debian/prerm` и `debian/postrm`. Эти сценарии занимаются настройкой установки или удаления любого пакета, которые не обрабатываются простым созданием или удалением файлов и каталогов. Следующие инструкции дополняют Политику Debian.

Сценарии сопровождающего должны быть идемпотентны. Это означает, что вам следует убедиться, что не произойдёт ничего плохого в том случае, когда сценарий будет вызван дважды, хотя обычно он вызывается только один раз.

Стандартные ввод и вывод могут быть перенаправлены (напр., конвейерами) с целью записи журнала для отладки, поэтому вам не следует считать, что они представляют собой лишь tty.

Все подсказки и интерактивные настройки должны быть сведены к минимуму. Когда это необходимо, для предоставления интерфейса следует использовать пакет `debconf`. Помните, что подсказки в любом случае могут быть выведены лишь на этапе `настройки` сценария `postinst`.

Делайте сценарии сопровождающего насколько простыми, насколько это возможно. Мы предлагаем использовать чистые сценарии оболочки POSIX. Помните, если вам нужны какие-либо возможности bash, сценарий сопровождающего должен содержать строку, объявляющую использование bash. Оболочка POSIX или Bash предпочтительны по отношению к Perl, поскольку они позволяют `debhelper` легко добавлять что-то новое в эти сценарии.

Если вы изменяете ваши сценарии сопровождающего, проверьте удаление пакета, двойную установку и очистку. Убедитесь, что очищенный пакет убран полностью, то есть, очистка должна удалить любые напрямую или косвенно созданные любым сценарием сопровождающего файлы.

Если вам необходимо проверить существование команды, используйте что-то вроде этого:

```
if [ -x /usr/sbin/install-docs ]; then ...
```

Если вы не хотите указывать точный путь команды в вашем сценарии сопровождающего, вам может быть полезна следующая POSIX-совместимая функция:

```
pathfind() {
    OLDIFS="$IFS"
    IFS=:
    for p in $PATH; do
        if [ -x "$p/$*" ]; then
            IFS="$OLDIFS"
            return 0
        fi
    done
    IFS="$OLDIFS"
    return 1
}
```

Вы можете использовать эту функцию для поиска `$PATH` для имения команды, переданной в качестве аргумента. Она возвращает истинное значение (ноль), если команда была найдена, и ложь, если нет. В действительности, это наиболее переносимый способ, поскольку `command -v`, **type** и **which** не совместимы с POSIX.

Хотя **which** является допустимой альтернативой, так как она поставляется в требуемом пакете `debianutils`, она находится не на корневом разделе. То есть, она располагается в `/usr/bin`, а не в `/bin`, поэтому её нельзя использовать в сценариях, которые запускаются до монтирования раздела `/usr`. Хотя, большинство сценариев не имеют такой проблемы.

6.5 Управление настройкой с помощью `debconf`

`Debconf` представляет собой систему управления настройкой, которая может использоваться всеми различными сценариями пакета (в основном `postinst`) для запроса ответной реакции пользователя касательно того, как настроить пакет. Следует избегать прямых взаимодействий с пользователем в пользу взаимодействия через `debconf`. В будущем это позволит осуществлять неинтерактивные установки.

Debconf является превосходным инструментом, но часто он плохо используется. Многие общие ошибки приведены на странице руководства debconf-devel(7). Вам следует прочитать это руководство, если вы решили использовать debconf. Кроме этого, здесь мы приводим ниже лучшие практики использования этого инструмента.

Эти принципы включают некоторые рекомендации по типографии и стилю письма, общие соображения об использовании debconf, а также более конкретные рекомендации для некоторых частей дистрибутива (например, системы установки).

6.5.1 Не злоупотребляйте debconf

Поскольку debconf возник в Debian, им часто злоупотребляют, и дистрибутив Debian получил изрядную долю критики за злоупотребление debconf необходимостью отвечать на большое количество вопросов для установки даже очень небольшого пакета.

Оставьте сообщения об использовании там, где они должны быть: в файлах `NEWS.Debian` или `README.Debian`. Используйте сообщения только для важных сообщений, которые могут напрямую оказать влияние на использование и работу пакета. Помните, что сообщения всегда блокируют установку до явного подтверждения пользователя, либо надоедают пользователю по электронной почте.

Аккуратно выбирайте приоритет вопросов в сценариях сопровождающего. Подробную информацию о приоритетах см. в debconf-devel(7). Большинство вопросов должны использовать средний и низкий приоритеты.

6.5.2 Общие рекомендации для авторов и переводчиков

6.5.2.1 Пишите на правильном английском

Большинство сопровождающих Debian не являются носителями английского языка. Поэтому написание правильно сформулированных шаблонов может быть для них нелегким делом.

Используйте список рассылки debian-l10n-english@lists.debian.org (и злоупотребляйте им). Пусть ваши шаблоны будут вычитаны.

Плохо написанные шаблоны создают плохое представление о вашем пакете, вашей работе... и даже самом Debian.

Избегайте технического жаргона насколько это возможно. Если некоторые термины звучат знакомо для вам, они всё равно могут быть непонятны остальным. Если вы не можете избежать этих терминов, попытайтесь объяснить их (используйте подробное описание). Если вы пишете описание, пытайтесь балансировать на грани подробности и простоты.

6.5.2.2 Будьте добры к переводчикам

Шаблоны debconf могут быть переведены. Debconf, а также его родственный пакет **po-debconf**, предлагают простую инфраструктуру для перевода шаблонов командами переводчиков или даже отдельными людьми.

Используйте шаблоны на основе gettext. Установите `po-debconf` в вашу систему, используемую для разработки, и прочтите документацию этого пакета (**man po-debconf** является хорошим началом).

Избегайте частого изменения шаблонов. Изменение текста шаблона приводит к дополнительной работе для переводчиков, поскольку их перевод будет отмечен как неясный. Неясный перевод представляет собой

строку, оригинальная строка которой была изменена с момента перевода, следовательно, от переводчика требуется обновление этой строки, чтобы она снова могла использоваться. Когда изменения достаточно малы, оригинальный перевод сохраняется в файлах PO, но отмечается как `неясный`.

Если вы планируете изменить ваши оригинальные шаблоны, для связи с переводчиками используйте систему оповещения, предоставляемую пакетом `po-debconf`, а именно **podebconf-report-po**. Наиболее активные переводчики очень отзывчивы, и включение их работы вместе с вашими изменёнными шаблонами позволит избежать вам дополнительных загрузок. Если вы используете шаблоны на основе gettext, имя переводчика и его адрес электронной почты указаны в заголовках PO-файлов и будут использоваться командой **podebconf-report-po**.

Рекомендованное использование этой утилиты:

```
cd debian/po && podebconf-report-po --call --languageteam --withtranslators -- ↩
    deadline="+10 days"
```

Эта команда, во-первых, синхронизирует PO-файлы и файлы POT в `debian/po` с файлами шаблонов, указанными в `debian/po/POTFILES.in`. Далее, она вышлет сообщение с призывом подготовить новые переводы в список рассылки debian-i18n@lists.debian.org. Наконец, она также вышлет сообщение с призывом обновить переводы команде переводчиков (указана в поле `Language-Team` в каждом PO-файле), а также последнему переводчику (указан в поле `Last-translator`).

Всегда ценится точный срок, отведённый для работы над переводом, чтобы переводчики могли организовать свою работу. Помните, что некоторые команды переводчиков используют формализованный процесс перевода/проверки, и задержка менее 10 дней считается чрезвычайно короткой. Более короткая задержка слишком сильно давит на команды переводчиков и должна устанавливаться только для очень небольших изменений.

Если у вас возникли сомнения, вы также можете связаться с командой перевода данного языка (debian-l10n-xxxxx@lists.debian.org) или списком рассылки debian-i18n@lists.debian.org.

6.5.2.3 Отменяйте статус неясных строк когда исправляете опечатки или орфографию

Когда текст шаблона debconf был изменён, и вы **уверены**, что это изменение **не** влияет на переводы, будьте добры по отношению к переводчикам и *отмените статус неясных строк* для их переводов.

Если вы не сделаете этого, весь шаблон не будет переведён пока переводчик не вышлет вам обновление.

Чтобы *отменить статус неясных строк* для переводов, вы можете использовать **msguntypot** (часть пакета `po4a`).

1. Создание файлов POT и PO заново.

   ```
   debconf-updatepo
   ```

2. Создание копии POT-файла.

   ```
   cp templates.pot templates.pot.orig
   ```

3. Создание копии всех PO-файлов.

   ```
   mkdir po_fridge; cp *.po po_fridge
   ```

4. Изменение файлов шаблона debconf для исправления ошибок.

5. Создание файлов POT и PO заново (опять).

   ```
   debconf-updatepo
   ```

   Сейчас исправление опечатки привело к тому, что все переводы были отмечены как неясные, и это несчастное изменение является единственным различием между файлами PO вашего основного каталога и сохранёнными во временном каталоге файлами. Вот как решить эту проблема.

6. Отбросьте неясный перевод, восстановите перевод из сохранённых во временном каталоге файлов.

   ```
   cp po_fridge/*.po .
   ```

7. Вручную слейте файлы PO с новым POT-файлов, но учтите бесполезные неясности.

```
msguntypot -o templates.pot.orig -n templates.pot *.po
```

8. Очистите.

```
rm -rf templates.pot.orig po_fridge
```

6.5.2.4 Не допускайте ничего по поводу интерфейсов

Текст шаблона не должен отсылать к виджетам, относящимся к каким-либо интерфейсам debconf. Предложения вида *If you answer Yes...* не имеют смысла для пользователей графических интерфейсов, которые используют флажки для вопросов с двумя вариантами ответа.

Строки шаблонов должны также избегать в своём описании упоминания значений по умолчанию. Во-первых, это излишне, так как значения видны пользователям. Также это излишне потому, что значения по умолчанию могут различаться в зависимости от выбора сопровождающего (например, когда база данных debconf была автоматизирована).

Вообще говоря, попытайтесь избежать указания на действия пользователя. Просто представляйте факты.

6.5.2.5 Не используйте первое лицо

Вам следует избегать использования первого лица (*I will do this...* или *We recommend...*). Компьютер не является личностью, шаблоны debconf не говорят за разработчиков Debian. Вам следует использовать нейтральные конструкции. Те их вас, кто уже писал ранее научные работы, просто пишите ваши шаблоны также, как вы писали бы научную статью. Тем не менее, попытайтесь использовать активный залог, если это возможно, например *Enable this if ...* вместо *This can be enabled if...*.

6.5.2.6 Будьте нейтральны в гендерном отношении

Мир состоит из мужчин и женщин. Используйте при написании нейтральные в отношении гендера конструкции.

6.5.3 Определение полей шаблонов

Эта часть содержит информацию, которая по большей части заимствована из страницы руководства debconf-devel(7).

6.5.3.1 Тип

6.5.3.1.1 string

Создаёт поле свободного ввода, в которое пользователь может ввести любую строку.

6.5.3.1.2 password

Запрашивает у пользователя пароль. Используйте с осторожностью; помните, что пароль, вводимый пользователем, будет записан в базу данных debconf. Вероятно, вам следует как можно скорее очистить это значение в базе данных.

6.5.3.1.3 boolean

Выбор истинно/ложно. Помните: истинно/ложно, **а не да/нет**...

6.5.3.1.4 select

Выбор одного значения из нескольких предложенных. Выбор должен быть определён в поле с именем 'Choices'. Разделите возможные значения запятыми и пробелами, например так: `Choices: yes, no, maybe`.

Если варианты выбора представляют собой переводимые строки, поле 'Choices' может быть отмечено как переводимое так: `__Choices`. Двойное подчёркивание выделит каждый вариант выбора в отдельную строку.

Также система **po-debconf** предлагает интересные возможности для отметки только **некоторых** вариантов выбора в качестве переводимых. Пример:

```
Template: foo/bar
Type: Select
#flag:translate:3
__Choices: PAL, SECAM, Other
_Description: TV standard:
 Please choose the TV standard used in your country.
```

В этом примере только строка 'Other' является переводимой, остальные же представляют собой акронимы, которые не нужно переводить. Этот пример разрешает включение в файлы PO и POT только варианта 'Other'.

Система флагов для шаблонов debconf предлагает множество таких возможностей. В странице руководства po-debconf(7) приведены все эти возможности.

6.5.3.1.5 multiselect

Подобен типу данных select, но пользователь может выбрать любое число вариантов из списка вариантов (или не выбрать ни один из них).

6.5.3.1.6 note

Этот тип данных не является вопросом самим по себе, а представляет собой примечание, которое может быть показано пользователю. Он должен использоваться только для важных примечаний, которые пользователи действительно должны заметить, поскольку debconf приложит максимум усилий для гарантии того, что пользователь заметит это примечание; установка будет остановлена до нажатия клавиши, в некоторых случаях пользователю будет даже выслано это примечание по почте.

6.5.3.1.7 text

Данный тип считается устаревшим: не используйте его.

6.5.3.1.8 error

Этот тип разработан для обработки сообщений об ошибках. Он практически схож с типом note. Различные пользовательские интерфейсы могут представлять его по-разному (например, диалоговый интерфейс cdebconf отрисовывает красный экран вместо обычного синего).

Рекомендуется использовать этот тип для любых сообщений, не которые необходимо обратить внимание пользователя для внесения исправлений любого вида.

6.5.3.2 Описание: краткое и расширенное описания

Описания шаблонов имеют две части: краткую и расширенную. Краткое описание даётся в строке Description: данного шаблона.

Краткое описание должно быть коротким (50 символов или около того), чтобы оно могло быть обработано большинством интерфейсов debconf. Краткое описание также помогает переводчикам, поскольку переводы обычно длиннее оригиналов.

Краткое описание должно быть самодостаточным. Некоторые интерфейсы по умолчанию не отображают длинное описание, либо отображают его только в том случае, если пользователь явно запросит об это, либо вообще никогда не отображают его. Избегайте кратких описаний на подобие What do you want to do?

Краткое описание не обязательно должно быть полным предложением. Это лишь часть, она должна быть краткой и эффективной рекомендацией.

Расширенное описание не должно слово в слово повторять краткое описание. Если вы не можете придумать длинное описание, то для начала подумайте ещё. Напишите в debian-devel. Попросите помощи. Прослушайте курс письма! Расширенное описание очень важно. Если в конце концов вы всё еще не можете ничего придумать, оставьте его пустым.

Расширенное описание должно состоять из полных предложений. Параграфы должны быть короткими и удобочитаемыми. Не смешивайте две идеи в одном параграфе, лучше разделите их на два.

Не будьте слишком подробны. Пользователи обычно игнорируют слишком длинные экраны. По опыту, 20 строк — это та граница, которую вам не следует пересекать, поскольку это означает, что в классическом диалоговом интерфейсе придётся прокручивать экран, а большинство пользователей этого просто не делают.

Расширенное описание **никогда** не должно включать в себя вопрос.

Для ознакомления с конкретными правилами, зависящими от типа шаблона (string, boolean и т. д.), прочтите нижеследующий текст.

6.5.3.3 Choices

Это поле должно использоваться для типов select и multiselect. Оно содержит возможные варианты выбора, которые будут представлены пользователям. Варианты выбора должны быть разделены запятыми.

6.5.3.4 Default

Это поле опционально. Оно содержит ответ по умолчанию для шаблонов string, select и multiselect. Для шаблонов multiselect это поле может содержать список вариантов выбора, разделённых запятыми.

6.5.4 Руководство по стилю в зависимости от полей шаблона

6.5.4.1 Тип поля

Без конкретного указания за исключением следующего: используйте соответствующий тип из предыдущего раздела.

6.5.4.2 Поле Description

Ниже приведены конкретные инструкции для правильного написания Description (краткого и расширенного описаний) в зависимости от типа шаблона.

6.5.4.2.1 Шаблоны string/password

- Краткое описание является приглашением, а **не** заголовком. Избегайте приглашений в виде вопросов (IP Address?) в пользу открытых приглашений (IP address:). Рекомендуется использовать двоеточие.
- Расширенное описание дополняет краткое описание. В расширенной части объясните пользователю, что спрашивается, а не задавайте ему тот же вопрос при помощи более длинных слов. Используйте полные предложения. Краткие стиль письма крайне не рекомендуется.

6.5.4.2.2 Шаблоны boolean

- Краткое описание должно быть сформулировано в виде вопроса, который должен быть кратким и обычно должен заканчиваться вопросительным знаком. Разрешается (и даже рекомендуется) использовать краткий стиль письма в том случае, если вопрос является длинным (помните, что переводы чаще длиннее, чем оригиналы).
- Опять же, избегайте обращения к конкретным виджетам интерфейса. Частой ошибкой таких шаблонов являются ответы на конструкции Да-типа.

6.5.4.2.3 select/multiselect

- Краткое описание является приглашением, а **не** заголовком. **Не** используйте бесполезные конструкции вида Please choose... Пользователи достаточно умны, чтобы понять, что им нужно что-то выбрать...:)
- Расширенное описание должно заканчивать краткое описание. Оно может отсылать к доступным вариантам выбора. Также оно может содержать напоминание о том, что пользователь может выбрать более одного варианта из доступных, если выбран шаблон multiselect (хотя интерфейс обычно делает это очевидными).

6.5.4.2.4 Примечания

- Краткое описание должно рассматриваться как **заголовок**.
- Расширенное описание представляет собой то, что будет отображено в качестве более подробного объяснения данного примечания. Фразы, подробный стиль письма.

- **Не злоупотребляйте debconf.** Примечания представляют собой наиболее распространённый тип злоупотребления debconf. Как указано в странице руководства debconf-devel: лучше использовать их только для предупреждения об очень серьёзных проблемах. Большинство примечаний следует помещать в файл `NEWS.Debian` или файл `README.Debian`. Если, читая это, вы решили перевести ваши шаблоны типа note в записи в файле `NEWS.Debian` или файле `README.Debian`, подумайте над тем, как сохранить на будущее существующие переводы.

6.5.4.3 Поле Choices

Если скорее всего поле Choices будет часто меняться, попробуйте использовать трюк __Choices. Так каждый индивидуальный выбор будет выделен в отдельную строку, что поможет переводчикам делать их работу.

6.5.4.4 Поле Default

Если скорее всего значение по умолчанию для выбранного шаблона будет изменяться в зависимости от языка пользователя (например, если это выбор языка), используйте трюк _Default.

Это специальное поле позволяет переводчикам указывать наиболее подходящий вариант выбора в соответствии с их собственным языком. Он будет вариантом по умолчанию при использовании этого языка, в то время как ваш собственный вариант Default Choice будет использоваться в том случае, если используется английский язык.

Пример, взятый из шаблонов пакета geneweb:

```
Template: geneweb/lang
Type: select
__Choices: Afrikaans (af), Bulgarian (bg), Catalan (ca), Chinese (zh), Czech (cs) ↩
    , Danish (da), Dutch (nl), English (en), Esperanto (eo), Estonian (et),  ↩
    Finnish (fi), French (fr), German (de), Hebrew (he), Icelandic (is), Italian  ↩
    (it), Latvian (lv), Norwegian (no), Polish (pl), Portuguese (pt), Romanian ( ↩
    ro), Russian (ru), Spanish (es), Swedish (sv)
# This is the default choice. Translators may put their own language here
# instead of the default.
# WARNING : you MUST use the ENGLISH NAME of your language
# For instance, the french translator will need to put French (fr) here.
_Default: English[ translators, please see comment in PO files]
_Description: Geneweb default language:
```

Обратите внимание на использование скобок, которые позволяют указывать внутренние комментарии в полях debconf. Кроме того, обратите внимание на использование комментариев, которые показываются в файлах, с которыми работают переводчики.

Эти комментарии необходимы постольку, поскольку трюк _Default несколько сбивает с толку: переводчики могут указать свой собственный вариант выбора

6.5.4.5 Поле Default

НЕ используйте пустое поле default. Если вы не хотите использовать значения по умолчанию, не используйте Default вообще.

Если вы используете po-debconf (а вы **должны** это делать, см. раздел 6.5.2.2, «Будьте добры к переводчикам»), постарайтесь сделать это поле так, чтобы его можно было перевести, если вы считаете, что оно может быть переведено.

Если значение по умолчанию может изменяться в зависимости от языка/страны (например, значение по умолчанию для выбора языка), используйте специальный тип _Default, описываемый в po-debconf(7).

6.6 Интернационализация

Этот раздел содержит глобальную информацию для разработчиком по облегчению жизни переводчиков. Дополнительная информация для переводчиком и разработчиков, заинтересованных в интернационализации, доступна в руководстве Интернационализация и локализация в Debian.

6.6.1 Обработка переводов debconf

Подобно тем, кто занимается переносами, перед переводчиками стоит сложная задача. Они работают над многими пакетами и должны сотрудничать с большим количеством разных сопровождающих. Более того, по большей части они не являются носителями английского языка, поэтому вам следует быть с ними особенно вежливыми.

Целью `debconf` является облегчение сопровождающим и пользователям настройки пакетов. Изначально перевод шаблонов debconf осуществлялся с помощью **debconf-mergetemplate**. Тем не менее, сейчас эта техника устарела; лучшим способом выполнить `debconf` интернационализацию состоит в использовании пакета `po-debconf` package. Этот метод проще и для сопровождающих и для переводчиков; предоставляются сценарии перевода.

При использовании `po-debconf`, перевод сохраняется в файлах `.po` (формате, извлечённом из **gettext**). Специальные файлы шаблонов содержат оригинальные сообщения и пометки для тех полей, которые могут быть переведены. Когда вы изменяете значение поля, которое может быть переведено, вызывая команду **debconf-updatepo**, перевод отмечается как требующий внимания переводчиков. Тогда, во время сборки, программа **dh_installdebconf** заботится обо всей требуемой магии по добавлению шаблона вместе с обновлёнными переводами в двоичные пакеты. За подробностями обратитесь к странице руководства po-debconf(7).

6.6.2 Интернационализованная документация

Интернационализация документации очень важна для пользователей, но требует большого труда. Не существует способа сделать так, чтобы не нужно было делать эту работу, но вы можете упростить её для переводчиков.

Если вы сопровождаете документацию любого размера, для переводчиков будет проще, если они будут иметь доступ к системе управления исходным кодом. Это позволяет переводчикам видеть разницу между двумя версиями документации, поэтому, например, они могут видеть, что нужно перевести заново. Рекомендуется, чтобы в переведённой документации содержалась информация о том, на базе какой системы управления изменениями исходного кода основывается разработка документации. Интересная система предоставляется doc-check из пакета `debian-installer`, которая показывает статус перевода для любого данного языка и использует структурированные комментарии для текущей редакции файла, который должен быть переведён, для переведённого файла, редакции оригинального файла, на котором основан перевод. Вероятно, вы захотите использовать и предоставлять такую информацию в вашей системе управления версиями.

Если вы сопровождаете документацию в форматах XML или SGML, мы предлагаем вам выделить отдельно всю независящую от языка информацию и определить её в виде сущности в отдельном файле, которые бы включался всеми разными переводами. Это позволяет упростить, например, контроль за актуальностью URL в нескольких файлах.

Некоторые инструменты (напр., `po4a`, `poxml` или `translate-toolkit`) специализируются на извлечении переводимой информации из различных форматов. Они создают файлы PO, этот формат довольно хорошо знаком переводчикам, он позволяет видеть то, что следует перевести заново, если переведённый документ был обновлён.

6.7 Общие ситуации при создании пакетов

6.7.1 Пакеты, использующие autoconf/automake

Поддержка файлов **autoconf** `config.sub` и `config.guess` в актуальном состоянии является критической задачей для тех, кто занимается переносом, особенно на более волатильные архитектуры. Некоторые очень хорошие практики по созданию пакетов для любого пакета, использующего **autoconf** и/или **automake** были синтезированы в `/usr/share/doc/autotools-dev/README.Debian.gz` из пакета `autotools-dev` package. Настойчиво рекомендуем прочитать этот файл и следовать приведённым в нём рекомендациям.

6.7.2 Библиотеки

По ряду причина для библиотек всегда трудно создавать пакеты. Политика налагает множество ограничений для облегчения из сопровождения и для того, чтобы гарантировать, что обновления будут настолько просты, насколько это возможно, когда выходит новая версия из основной ветки разработки. Поломка в библиотеке может вызвать поломку множества зависимых пакетов.

Хорошие практики по созданию пакетов для библиотек были сгруппированы в справочнику по созданию пакетов библиотек.

6.7.3 Документация

Обязательно пройдите по ссылке Политика документации.

Если ваш пакет содержит документацию, собираемую из XML или SGML, рекомендуем вам не добавлять исходный код в форматах XML или SGML в двоичный пакет (-ы). Если пользователи захотят получить исходный код документации, им следует загрузить пакет с исходным кодом.

Политика определяет, что документация должна поставляться в формате HTML. Мы также рекомендуем поставлять документацию в формате PDF и в виде обычного текста, если это удобно, и если возможно обеспечить вывод достаточного качества. Тем не менее, предоставление документации, исходным форматом которой является HTML, в виде обычного текста не является подходящим.

Крупные руководства должны при установке регистрироваться в `doc-base`. Дополнительную информацию см. в документации пакета `doc-base`.

Политика Debian (раздел 12.1) указывает, что справочные страницы должны поставляться со всякой программой, утилитой и функцией, и предлагает поставку справочных страниц для других объектов, таких как файлы настройки. Если та работа, для которой вы создаёте пакеты, не имеет таких справочных страниц, постарайтесь написать их самостоятельно для включения в ваш пакет и отправки в основную ветку разработки.

Справочные страницы не должны быть написаны непосредственно в формате troff. Популярными исходными форматами являются Docbook, POD и reST, которые могут быть конвертированы с помощью **xsltproc**, **pod2man** и **rst2man** соответственно. В меньшей степени для написания заглушки может использоваться программа **help2man**.

6.7.4 Конкретные типы пакетов

Некоторые конкретные типы пакетов имеют свои специальные подполитики и соответствующие правила и практики создания пакетов:

- Пакеты, связанные с Perl, имеют Политику Perl, некоторые примеры пакетов, следующих этой политике: `libdbd-pg-perl` (двоичный модуль perl) или `libmldbm-perl` (зависящий от архитектуры модуль perl).
- Связанные с Python пакеты имеют свою Политику Python; см. `/usr/share/doc/python/python-policy.txt.gz` в пакете `python`.
- Связанные с Emacs пакеты имеют Политику Emacs.
- Связанные с Java пакеты имеют свою Политику Java.
- Связанные с Ocaml имеют свою собственную политику, см. в `/usr/share/doc/ocaml/ocaml_packaging_policy.gz` из пакета `ocaml`. Хорошим примером является пакет с исходным кодом `camlzip`.
- Пакеты, предоставляющие XML или SGML DTD должны соответствовать рекомендациям из пакета `sgml-base-doc`.
- Пакеты Lisp должны регистрироваться в `common-lisp-controller`, об этом см. `/usr/share/doc/common-lisp-controller/README.packaging`.

6.7.5 Независящие от архитектуры данные

Добавление большого количества независящих от архитектуры данных в пакеты с программами не является в каком бы то ни было смысле необычным явлением при создании пакетов. Например, аудио файлы, наборы иконок, шаблоны обоев рабочего стола или другие графические файлы. Если размер этих данных незначителен по сравнению с размером остального пакета, возможно лучшим решением будет сохранение всех данных в одном пакете.

Тем не менее, если размер данных значителен, постарайтесь выделить их в отдельный независящий от архитектуры пакет (`_all.deb`). Делая это, вы избежите ненужного дублирования одних и тех же данных в одиннадцати или более файлах .debs, по одному на каждую архитектуру. Хотя это добавляет некоторую дополнительную информацию в файлы `Packages`, это сохраняет большое количество места на зеркалах

Debian. Выделение независящих от архитектуры данных также снижает время обработки **lintian** (см. раздел A.2, «Инструменты для проверки пакетов на предмет ошибок и соответствия стандартам») при проверке всего архива Debian.

6.7.6 Необходимость в конкретной локали во время сборки

Если вам требуется определённая локаль во время сборки, вы можете создать временный файл с помощью следующих команд:

Если вы установите `LOCPATH` значение, эквивалентное `/usr/lib/locale`, а значение `LC_ALL` будет имя той локали, которую вы создаёте, у вас будет то, что вам нужно при работе из-за учётной записи суперпользователя. Что-то вроде этого:

```
LOCALE_PATH=debian/tmpdir/usr/lib/locale
LOCALE_NAME=en_IN
LOCALE_CHARSET=UTF-8

mkdir -p $LOCALE_PATH
localedef -i $LOCALE_NAME.$LOCALE_CHARSET -f $LOCALE_CHARSET $LOCALE_PATH/ ↩
    $LOCALE_NAME.$LOCALE_CHARSET

# Используйте локаль
LOCPATH=$LOCALE_PATH LC_ALL=$LOCALE_NAME.$LOCALE_CHARSET date
```

6.7.7 Делаем так, чтобы deborphan определят переходные пакеты

deborphan — программам, помогающая пользователям определить то, какие пакеты могут быть безопасно удалены из системы, т. е. те, от которых не зависит более ни один пакет. По умолчанию поиск осуществляется только в разделах libs и oldlibs для обнаружения неиспользуемых библиотек. Но когда вы передали этой программе необходимых аргумент, она пытается найти все бесполезные пакеты.

Например, с аргументом `--guess-dummy`, **deborphan** пытается найти все переходные пакеты, которые нужны для обновления, но которые теперь могут быть легко удалены. Для этого она ищет строку dummy или transitional в их кратком описании.

Поэтому когда вы создаёте такой пакет, убедитесь, что этот текст добавлен в ваше краткое описание. Если вы хотите ознакомиться с примерами, запустите: **apt-cache search .|grep dummy** или **apt-cache search .|grep transitional**.

Кроме того, рекомендуется установить раздел в значение `oldlibs`, а приоритет в значение `extra` для того, чтобы упростить работу **deborphan**.

6.7.8 Лучшие практики для файлов `.orig.tar.{gz,bz2,xz}`

Существует два вида оригинальных tar-архивов с исходным кодом: чистый исходный код и запакованный заново исходный код основной ветки разработки.

6.7.8.1 Чистый исходный код

Определяющей характеристикой чистого tar-архива с исходным кодом является то, что файл `.orig.tar.{gz,bz2,xz}` побайтово идентичен tar-архиву, который официальной распространялся автором основной ветки разработки.[1] Это позволяет использовать контрольные суммы для простой проверки, что все изменения между версией Debian и версией основной ветки содержаться в db=iff-файле Debian. Кроме того, если оригинальный исходный код имеет большой размер, авторы основной ветки разработки и другие, кто уже имеют tar-архив из основной ветки, могут сохранить время, затрачиваемое на скачивание, в том случае, если они хотят подробно проверить ваш пакет.

Нет никаких универсально принимаемых рекомендаций, которым бы следовали авторы основной ветки разработки, по поводу структуры каталогов в tar-архивах, но **dpkg-source** может, тем не менее, обрабатывать

[1] Мы не можем запретить авторам основной ветки разработки изменять tar-архив, поставляемый ими, без одновременного изменения номера версии, поэтому не может быть никаких гарантий, что чистый исходный код идентичен тому, что поставляется основной веткой разработки *в данный момент*. Всё, что можно ожидать, состоит в том, что он идентичен чему-то, кто когда-то *распространялось* основной веткой разработки. Если позже обнаруживается разница (например, если основная ветка разработки извещает о том, что они не использовали максимальное сжатие для своих оригинальных файлов и затем заново сжали их с помощью **gzip**), это просто очень плохо. Поскольку нет какого-либо хорошего способа загрузить новый файл `.orig.tar.{gz,bz2,xz}` для той же версии, то нет и оснований считать эту ситуацию ошибкой.

большинство tar-архивов основных веток разработки как чистый исходный код. Его стратегия эквивалентна следующему:

1. Он распаковывает tar-архив в пустой временный каталог, выполняя

```
zcat путь/к/имя-пакета_версия-основной-ветки.orig.tar.gz | tar xf -
```

2. Если после этого временный каталог не содержит ничего кроме одного каталога и не содержит других файлов, **dpkg-source** переименовывает этот каталог в *имяпакета-версия-основной-ветки*`(.orig)`. Имя каталога верхнего уровня в tar-архиве не имеет значения и просто забывается.

3. В противном случае, tar-архив основной ветки должен быть запакован без общего каталога верхнего уровня (позор автору основной ветки!). В этом случае **dpkg-source** переименовывает *самостоятельно* временный каталог в *имяпакета-версия-основной-ветки*`(.orig)`.

6.7.8.2 Повторная упаковка исходного кода основной ветки

Вы **должны** загружать пакеты с чистым исходным кодом, если это возможно, но имеются причины, по которым это может быть невозможно. Это имеет место, когда основная ветка вообще не поставляет исходный код в виде сжатого при помощи gzip tar-архива, либо если tar-архив основной ветки содержит не свободный в соответствии с критериями DFSG материал, который вам следует удалить до момента осуществления загрузки.

В этих случаях разработчик должен создать подходящий файл `.orig.tar.{gz,bz2,xz}` самостоятельно. Мы говорим о таком tar-архиве как о запакованном заново исходном коде основной ветки. Заметьте, что запакованный заново исходный код основной ветки отличается от родного пакета Debian. Запакованный заново исходный код содержит специфичные для Debian изменения в отдельных файлах `.diff.gz` или `.debian.tar.{gz,bz2,xz}` и всё ещё имеет номер версии, составленный из *версия-основной-ветки* и *версия-debian*.

Возможны такие случаи, когда желательно запаковать исходный код заново, даже несмотря на то, что основная ветка разработки поставляет `.tar.{gz,bz2,xz}`, который в принципе может использовать в его чистой форме. Наиболее очевидным случаем является ситуация, когда может быть достигнута *значительная* экономия места путём повторной упаковки tar-архива или путём удаления действительно ненужного хлама из архива основной ветки. Используйте это на своё усмотрение, но будьте готовы защищать своё решение, если вы запаковали заново исходный код, которые мог бы быть чистым.

Запакованный заново `.orig.tar.{gz,bz2,xz}`

1. **должен** быть документирован в получившемся пакете с исходным кодом. Подробная информация о том, как был получен запакованный заново исходный код, а также о том, как этот процесс может быть воспроизведён, должна быть предоставлена в `debian/copyright`. Также рекомендуется предоставлять цель `get-orig-source` в вашем файле `debian/rules`, которая повторяет этот процесс, это описано в Руководстве по политике, Основной сборочный сценарий: `debian/rules`.

2. **не должен** содержать какие-либо файлы, которые получены не от авторов основной ветки разработки, или содержание которых было изменено вами.[2]

3. **должен**, за исключением случаев, когда это невозможно по юридическим причинам, сохранять всю целиком сборочную инфраструктуру и инфраструктуру для переноса, предоставляемую автором основной ветки разработки. Например, то, что какой-то файл используется только для сборки под MS-DOS, не является достаточным основанием для его удаления. Сходным образом, `Makefile`, предоставляемый основной веткой разработки не должен быть удалён, даже если ваш файл `debian/rules` перезаписывает его, запуская сценарий настройки.

   (*Обоснование:* Обычно пользователи Debian, желающие собрать ПО для отличных от Debian платформ, загружают исходный код с зеркала Debian, а не пытаются найти каноничный выпуск основной ветки разработки).

4. **должен** использовать *имяпакета-версия-основной-ветки*`.orig` в качестве имени каталога верхнего уровня в своём tar-архиве. Это позволяет отличить чистые tar-архивы от запакованных заново.

5. **должен** быть сжат с помощью gzip или bzip с максимальным сжатием.

[2] Как специальное исключение если удаление несвободных файлов приведёт к тому, что исходный код перестанет собираться без помощи Debian diff, возможно отредактировать файлы, исключая из них лишь несвободные части, и/или объяснить ситуацию в файле `README.source` в корневом каталоге дерева исходного кода. Однако в этом случае, пожалуйста, также постарайтесь убедить автора основной ветки разработки облегчить отделение несвободных компонентов от остального исходного кода.

6.7.8.3 Изменение двоичных файлов

Иногда необходимо изменить двоичные файлы, содержащиеся в оригинальных tar-архивах, либо добавить двоичные файлы, которые не включены в оригинальный архив. Это полностью поддерживается, если используется формат пакетов с исходным кодом “3.0 (quilt)”, подробности см. на странице руководства dpkg-source(1). Если используется более старый формат “1.0”, двоичные файлы не могут быть сохранены в `.diff.gz`, поэтому вам придётся сохранять их в текстовой форме, в виде **uuencode** (или подобных) версий файлов и декодировать их во время сборки в `debian/rules` (и перемещать их в официальный каталог).

6.7.9 Лучшие практики для отладочных пакетов

Отладочный пакет - пакет, имя которого заканчивается на -dbg, содержащий дополнительную информацию, которую может использовать **gdb**. Поскольку двоичные файлы в Debian по умолчанию поставляются в сокращённом варианте, отладочная информация, включающая имена функций и номера строк, не доступна при запуске **gdb** для двоичных файлов Debian. Отладочные пакеты позволяют пользователю, которым требуется эта дополнительная информация, установить её, не засоряя обычную систему этой информацией.

От сопровождающего пакета зависит то, будет создан отладочный пакет или нет. Мы призываем сопровождающих создавать отладочные пакеты для пакетов с библиотеками, поскольку это может помочь в отладке многих программ, слинкованых с библиотекой. Вообще, отладочные пакеты не нужно добавлять для всех программ; это засорит архив. Однако если сопровождающий считает, что пользователям часто требуется отладочная версия программы, стоит создать для неё отладочный пакет. Программы, относящиеся к центральной инфраструктуре, такие как apache и сервер X, также являются хорошими кандидатами на создание отладочных пакетов.

Некоторые отладочные пакеты могут содержать всю целиком специальную отладочную сборку библиотеки или других двоичных файлов, но большая часть этих пакетов может сохранить место на диске и время, требуемое для сборки, если они включают в себя отдельные отладочные символы, которые могут быть найдены и загружен на лету **gdb** при отладке программы или библиотеки. По договорённости, в Debian эти символы сохраняются в `/usr/lib/debug/`*`путь`*, где *`путь`* является путём к исполняемому файлу или библиотеке. Например, отладочные символы для `/usr/bin/foo` помещаются в `/usr/lib/debug/usr/bin/foo`, а отладочные символы для `/usr/lib/libfoo.so.1` помещаются в `/usr/lib/debug/usr/lib/libfoo.so.1`.

Отладочные символы могут быть извлечены их объектного файла при помощи команды **objcopy --only-keep-debug**. Далее, объектный файл может быть очищен, а команда **objcopy --add-gnu-debuglink** используется для определения пути к файлу с отладочными символами. objcopy(1) подробно объясняет то, как это работает.

Команда **dh_strip** в `debhelper` поддерживает создание отладочных пакетов, она может позаботиться об использовании **objcopy** для выделения для вас отладочных символов. Если ваш пакет использует `debhelper`, всё, что вам нужно сделать, - вызвать **dh_strip --dbg-package=libfoo-dbg** и добавить пункт в `debian/control` для отладочного пакета.

Заметьте, отладочный пакет должен зависеть от пакета, для которого он предоставляет отладочные символы, и эта зависимость должна быть зависимостью от конкретной версии. Например:

```
Depends: libfoo (= ${binary:Version})
```

6.7.10 Лучшие практики для метапакетов

Метапакет обычно является пустым пакетом, который облегчает установку связанного набора пакетов, которые со временем могут развиваться. Это достигается благодаря тому, что метапакет зависит от всех пакетов данного набора. Благодаря мощи APT, сопровождающий метапакета может изменять зависимости, и пользовательская система будет автоматически получать дополнительные пакеты. Кроме того, отброшенные пакеты, которые были установлены автоматически, будут помечены как кандидаты на удаление (и даже будут автоматически удалены с помощью **aptitude**). `gnome` и `linux-image-amd64` являются примерами метапакетов (собранных из пакетов с исходным кодом `meta-gnome2` и `linux-latest`).

Длинное описание метапакета должно явным образом содержать информацию о его цели, чтобы пользователи знали, что они потеряют, если удалят данный пакет. Рекомендуется ясно прописывать последствия. Особенно это важно для сопровождающих метапакетов, которые устанавливаются во время начальной установки и которые не были явным образом установлены пользователем. Эти пакеты важны для обеспечения гладкого обновления системы, следует отговаривать пользователей от удаления этих пакетов, чтобы избежать потенциальных поломок.

Глава 7

Помимо создания пакетов

Debian предполагает гораздо больше, чем простое создание пакетов ПО и сопровождение этих пакетов. Данная глава содержит информацию о способах, часто действительно критичных способах, участия в Debian помимо создания и сопровождения пакетов.

Будучи добровольной организацией, Debian полагается на благоразумие своих членов в их выборе того, над чем они хотят работать, а также в выборе наиболее критических задач, исходя из своего собственного времени.

7.1 Отправка отчётов об ошибках

Мы призываем вас сообщать об ошибках в пакетах Debian как только вы найдёте их. Фактически, разработчики Debian зачастую являются тестировщиками первой волны. Обнаружение ошибки и сообщение о ней другим разработчикам пакетов повышает качество Debian.

Прочтите инструкции по отправке отчётов об ошибках в систему отслеживания ошибок Debian.

Попытайтесь отправить сообщение об ошибке от учётной записи обычного пользователя, на которую вы получаете почту, чтобы люди могли с вами связаться, если им потребуется дополнительная информация об ошибке. Не отправляйте отчёты от учётной записи суперпользователя.

Вы можете использовать инструмент на подобие reportbug(1) для отправки сообщений об ошибках. Это может автоматизировать и в целом упростить процесс.

Убедитесь, что о данной ошибке в данном пакете ещё не сообщали. Всякий пакет имеет свой список ошибок, доступ к которому можно легко получить, по адресу `https://bugs.debian.org/`*`имя-пакета`*. Такие утилиты как querybts(1) также могут предоставить вам информацию (**reportbug** обычно вызывает команду **querybts** до отправки отчёта).

Попытайтесь направить ваше сообщение об ошибках туда, где они нужны. Когда, например, ваша ошибка касается пакета, который перезаписывает файлы из другого пакета, проверьте список ошибок, *оба* этих пакета нужны для того, чтобы избежать заполнения дублирующих отчётов об ошибках.

Для получения дополнительного кредита доверия вы можете просмотреть другие пакеты, сливая сообщения об ошибках, о которых было сообщено несколько раз, либо помечая ошибки как исправленные (fixed'), если они уже исправлены. Заметьте, что если вы не являетесь ни автором сообщения об ошибке, ни сопровождающим пакета, вам не следует фактически закрывать ошибку (если только вы не получили на это разрешение от сопровождающего).

Время от времени вам может захотеться проверить, что происходит с отчётами об ошибках, которые вы отправили. Используйте эту возможность, чтобы закрыть те ошибки, которые вы более не можете воспроизвести. Чтобы найти все ошибки, о которых вы отправляли отчёты, вам следует перейти на страницу `https://bugs.debian.org/from:`*`ваш-адрес-электронной-почты`*.

7.1.1 Отправка множества отчётов об ошибках за один раз (массовое заполнение отчётов об ошибках)

Сообщение о большом количестве ошибок, связанных с одной и той же проблемой в большом количестве различных пакетов — напр., более 10 — является плохой практикой. Предпримите все возможные шаги, чтобы не допустить отправки массы сообщений об ошибках. Например, если проверка данной проблемы может быть автоматизирована, добавьте новую проверку в `lintian`, так чтобы выводилось сообщение об ошибке или предупреждение.

Если вы сообщаете о более чем 10 ошибках на одну и ту же тему за раз, рекомендуется выслать сообщение на адрес debian-devel@lists.debian.org с описанием вашего намерения до того, как вы вышлете свой отчёт, укажите этот факт в теме сообщения. Это позволит другим разработчикам проверить, что ошибка действительно является проблемой. Кроме того, это позволит избежать ситуации, при которой несколько сопровождающих одновременно начинают заполнять один и тот же отчёт об ошибке.

Пожалуйста, используйте программы **dd-list** и, если это подходит, **whodepends** (из пакета `devscripts`) для создания списка всех подверженных данной ошибке пакетов и добавьте вывод этих программ в ваше сообщение по адресу debian-devel@lists.debian.org.

Заметьте, что когда вы высылаете сразу много сообщений об ошибках на одну и ту же тему, вам следует выслать отчёт об ошибке по адресу maintonly@bugs.debian.org, чтобы ваш отчёт об ошибке не был переслан в список рассылки об ошибках.

7.1.1.1 Пользовательские метки

Возможно вам захочется использовать пользовательские метки системы отслеживания ошибок при отправке сообщений об ошибках в нескольких пакетах. Пользовательские метки (usertags) схожи с обычные метками, такими как 'patch' и 'wishlist', но отличаются от них в том, что они определяются пользователями и занимают пространство имён, уникальное для каждого отдельного пользователя. Это позволяет нескольким подмножествам разработчиков отмечать пользовательскими метками одну и ту же ошибку разными способами без возникновения конфликтов.

Чтобы добавить пользовательские метки при заполнении отчётов об ошибках, укажите псевдозаголовки `User` и `Usertags`:

```
To: submit@bugs.debian.org
Subject: заголовок-ошибки

Package: имяпакета
[ ... ]
User: адрес-email
Usertags: имя-метки [ имя-метки ... ]

описание-ошибки ...
```

Заметьте, что метки отделяются пробелами и не могут содержать подчёркиваний. Если вы заполняете отчёты об ошибках для конкретной группы или команды, рекомендуется установить псевдозаголовок `User` в значение соответствующего списка рассылки после сообщения о вашем намерении в этом списке рассылки.

Чтобы просмотреть сообщения об ошибках, помеченные конкретной пользовательской меткой, посетите страницу `https://bugs.debian.org/cgi-bin/pkgreport.cgi?users=`*адрес-электронной-почты*`&tag=`*имя-мет*

7.2 Работа по контролю качества

7.2.1 Ежедневная работа

Даже несмотря на то, что у нас имеется выделенная группа людей, которые занимаются контролем качества (Quality Assurance), обязанности по контролю качества не резервируются исключительно за ними. Вы можете принять участие в этой работе, стараясь сделать так, чтобы в ваших пакетах было как можно меньше ошибок, и чтобы ваши пакеты проходили как можно большее количество проверок lintian (см. раздел A.2.1, «`lintian`»). Если считаете, что это невозможно, вам, вероятно, следует отказаться (осиротить) некоторые из ваших пакетов (см. раздел 5.9.4, «Придание статуса осиротевшего пакета»). С другой стороны, вы можете попросить о помощи других людей, чтобы наверстать отставание по количеству исправленных ошибок (вы можете попросить о помощи в списках рассылки debian-qa@lists.debian.org или debian-devel@lists.debian.org). В то же время вы можете искать помощников сопровождающего (см. раздел 5.12, «Совместное сопровождение»).

7.2.2 Вечеринки по исправлению ошибок

Время от времени группа QA организует вечерники по исправлению ошибок для того, чтобы избавиться от как можно большего количества проблем. Они анонсируются в списке рассылки debian-devel-announce@lists.debian.org, анонс содержит информацию о том, на каких областях будет сфокусирована вечеринка: обычно они концентрируются на критичных для выпуска ошибках, но может случиться и так, что они решать помочь завершить

крупное обновление (например, до новой версии **perl**, что потребует перекомпиляции всех двоичных модулей).

Во время таких вечеринок действуют другие правила для загрузок от несопровождающих (NMU), поскольку анонс вечеринки делается в первую очередь для NMU. Если у вас имеются пакеты, которые могут быть затронуты во время вечеринки (поскольку в них имеются, например, критичные для выпуска ошибки), вам следует выслать обновление к каждой соответствующей ошибке с объяснением текущего статуса и того, что вы ждёте от данной вечеринки. Если вы не хотите иметь дело с NMU, или если вы заинтересованы только в заплате, либо если вы сами разберётесь с данной ошибкой, сообщите об этом в систему отслеживания ошибок.

Люди, принимающие участие в такой вечеринке, имеют специальные правила для NMU, они могут осуществлять NMU без предварительного объявления об этом, если они загружают свою NMU по меньшей мере в DELAYED/3-day. Все другие правила NMU применяются как обычно; они должны выслать заплату NMU в систему отслеживания ошибок (в один из открытых отчётов об ошибке, исправленных NMU, либо в новый отчёт об ошибке, пометив его как исправленный). Также они должны уважать любых конкретные пожелания сопровождающего.

Если вы не чувствуете уверенности по поводу NMU, просто вышлите заплату в систему отслеживания ошибок. Это намного лучше, чем сломанная NMU.

7.3 Связь с другими сопровождающими

Во время срока вашей деятельности в Debian вам придётся связываться с другими сопровождающими по совершенно разным причинам. Возможно, вы захотите обсудить новый способ взаимодействия между наборами связных пакетов, или просто напомнить кому-то о том, что теперь доступна новая версия основной ветки разработки, и она вам нужна.

Поиск адреса электронной почты сопровождающего пакета может отвлекать. К счастью, существуют простые псевдонимы для электронной почты, *пакет*`@packages.debian.org`, которые предоставляют способ отправки сообщения сопровождающему, каким бы ни был его индивидуальный адрес электронной почты (или адреса). Замените *пакет* на имя пакета с исходным кодом или имя двоичного пакета.

Возможно, вы захотите связаться с теми, кто подписан на данный пакет с исходным кодом через систему отслеживания пакетов (раздел 4.10, «Система отслеживания пакетов Debian»). Вы можете сделать это, используя адрес электронной почты *пакет*`@packages.qa.debian.org`.

7.4 Работа с неактивными и/или недоступными сопровождающими

Если вы заметили, что пакет плохо сопровождается, вам следует убедиться, что сопровождающий активен и будет продолжать работу над своими пакетами. Возможно, что сопровождающий более не активен, но, так сказать, ещё не отменил своей регистрации в системе. С другой стороны, также возможно, что этому сопровождающему просто нужно напомнить о его обязанностях.

Существует простая система (база данных MIA), в которую записывается информация о сопровождающих, которые считаются пропавшими без вести (Missing In Action). Когда член группы QA связывается с неактивным сопровождающим или находит дополнительную информацию о нём, это записывается в базу данных MIA. Эта система доступна в `/org/qa.debian.org/mia` на узле `qa.debian.org` и может быть запрошена с помощью инструмента **mia-query**. Используйте **mia-query --help**, чтобы узнать, как запрашивать базу данных. Если вы обнаружите, что о конкретном неактивном сопровождающем нет никакой информации, что вы можете добавить дополнительную информацию, вам следует действовать следующим образом.

В качестве первого шага вежливо свяжитесь с сопровождающим, ждите его ответ разумное количество времени. Определить, какое количество времени является разумным, очень сложно, но важно принимать в расчёт то, что фактическая жизнь иногда может быть очень беспокойной. Например, можно отправить ещё одно напоминание через две недели.

Если сопровождающий не отвечает в течении четырёх недель (месяца), можно предположить, что он вероятнее всего более не ответит. Если это произошло, вам нужно расследовать дальше и попытаться собрать как можно больше полезной информации о данном сопровождающем. Это предполагает следующее:

- `Эшелон` информации доступен через базу данных LDAP разработчиков, в которой указывается, когда последний раз данный разработчик писал в список рассылки Debian. (База данных содержит информацию о почте по поводу загрузок, распространяемой через список рассылки debian-devel-changes@lists.debian.org.)

Кроме того, проверьте, отмечен ли данный сопровождающий в этой базе данных как находящийся в отпуске.

- Число пакетов, за которые ответственен этот сопровождающий, и состояние этих пакетов. В частности, имеются ли какие-либо критичные для выпуска ошибки, которые остаются открытыми долгое время? Более того, как много вообще ошибок? Другой важной информацией является то, были ли эти пакеты загружены несопровождающим, и если да, то кем.

- Имеется ли какая-либо активность этого сопровождающего за пределами Debian? Например, он может написать что-то в список рассылки, не связанный с Debian, или в какую-нибудь новостную группу.

Некоторой проблемой являются пакеты, которые были поручены — их сопровождающий не является официальным разработчиком Debian. `Эшелон` информации не доступен для людей с поручителями, поэтому вам следует найти и связаться с разработчиком Debian, который фактически загрузил этот пакет. Учитывая, что они подписали пакет, они всё равно ответственны за загрузку и должны знать, что случилось с тем, за кого они поручались.

Также разрешается писать запрос в debian-devel@lists.debian.org, спрашивая о том, знает ли кто-нибудь о том, где находится пропавший сопровождающий. Пожалуйста, вышлите копию письма этому человеку.

Когда вы собрали всю эту информацию, вы можете связаться с mia@qa.debian.org. Люди, получающие почту с этого псевдонима, будут использовать предоставленную вами информацию для того, чтобы решить, как действовать дальше. Например, они могут осиротить один или все пакеты данного сопровождающего. Если пакет был загружен несопровождающим, они могут связаться с тем, что осуществил эту загрузку, до того, как осиротят пакет — возможно, человек, который осуществил эту загрузку, заинтересован в этом пакете.

Одно последнее слово: пожалуйста, будьте вежливы. Все мы являемся добровольцами, и мы не можем выделать всё наше время Debian. Кроме того, вам не известны обстоятельства, в которых находится этот человек. Возможно, он серьезно болен или даже мог умереть — вы не знаете, кто получает его почту. Представьте, как будут чувствовать себя родные, если они прочтут сообщение, адресованное умершему, и это сообщение будет невежливым, злым или будет содержать обвинения!

С другой стороны, хотя мы и являемся добровольцами, мы несём ответственность. Поэтому вы можете подчеркнуть важность большей пользы — если у сопровождающего нет времени, или он более не заинтересован в работе над своими пакетами, он должен передать свои пакеты тому, у кого времени больше.

Если вы заинтересованы в работе в команде MIA, посмотрите файл `README` в `/org/qa.debian.org/mia` на `qa.debian.org`, где описаны технические детали и процедуры MIA, а также свяжитесь с mia@qa.debian.org.

7.5 Взаимодействие с будущими разработчиками Debian

Успех Debian зависит от его способности привлекать и сохранять новых и талантливых добровольцев. Если вы являетесь опытным разработчиком, мы рекомендуем вам принять участие в процессе привлечения новых разработчиков. Данный раздел описывает то, как помогать новым будущим разработчиками.

7.5.1 Поручение пакетов

Поручение пакета предполагает загрузку пакета для сопровождающего, который сам этого сделать не может. Это нетривиальная задача, поручитель должен проверить то, как создан пакет, и гарантировать его высокое качество, к которому стремится Debian.

Разработчики Debian могут выступать поручителями для пакетов. Сопровождающие Debian не могут.

Процесс поручения пакета таков:

1. Сопровождающий готовит пакет с исходным кодом (`.dsc`) и помещает его куда-нибудь в сеть (например, на mentors.debian.net) или, что даже лучше, предоставляет ссылку на публичный репозиторий системы управления версиями (см. раздел 4.4.5, «Серверы систем управления версиями»), в рамках которого осуществляется сопровождение пакета.

2. Поручитель скачивает (или получает) пакет с исходным кодом.

3. Поручитель проверяет пакет с исходным кодом. Если будут найдены какие-либо проблемы, он информируем об этом сопровождающего и просит его предоставить исправленную версию (процесс начинается снова с шага 1).

4. Поручитель не нашёл каких-либо оставшихся проблем. Он собирает пакет, подписывает его и загружает пакет в Debian.

До того как вникнуть в детали, как осуществлять поручение пакета, вам следует спросить себя, является ли добавление предложенного пакета полезным для Debian.

Нет никакого простого правила для ответа на этот вопрос, это может зависеть от множества факторов: является ли кодовая база основной ветки разработки достаточно зрела и в ней не так много дыр в безопасности? Имеется ли уже существующий пакет, которые может выполнять ту же задачу, и каково различие между существующими пакетами и этим новым пакетом? Был ли новый пакет запрошен пользователями, какова его пользовательская база? Насколько активны разработчики основной ветки?

Кроме того, вам следует убедиться, что предполагаемый сопровождающий будет хорошим сопровождающим. Имеет ли он какой-либо опыт работы с другими пакетами? Если да, то хорошо ли он с ними работал (проверьте несколько ошибок)? Знаком ли он с пакетом и языком программирования, на котором написана программа? Имеет ли он требуемые для данного пакета навыки? Если нет, то способен ли он ими овладеть?

Выяснение того, как он относится к Debian, также является хорошей идеей: согласен ли он с философией Debian и хочет ли присоединиться к Debian? Учитывая то, насколько просто стать сопровождающим Debian, вы вероятно захотите поручиться за пакеты только тех людей, которые хотят присоединиться к Проекту. В этом случае вы с самого начала будет знать, что вам не придётся быть поручителем в течении какого-то неопределённого времени.

7.5.1.1 Поручительство нового пакета

Обычно новые сопровождающие испытывают определённые трудности при создании пакетов Debian — это вполне можно понять. Они будут совершать ошибки. Вот почему поручительство нового пакета в Debian требует проведения тщательного обзора процесса создания пакетов Debian. Иногда потребуется несколько повторений для того, чтобы пакет стал достаточно хорошим для его загрузки в архив Debian. Таким образом, поручительство предполагает менторство.

Никогда не поручайтесь за новый пакет, если вы его не изучили. Изучение новых пакетов осуществляется сопровождающими FTP-архива, это гарантирует, что ПО действительно является свободным. Конечно, иногда они обнаруживают проблемы создания пакета, но вообще они не должны этим заниматься. Это ваша задача, вы должны гарантировать, что загружаемый пакет соответствует Критериям Debian по определению Свободного ПО, и что он создан качественно.

Сборка пакета и тестирование ПО являются частью исследования пакета, но этого не достаточно. Оставшаяся часть данного раздела содержит неполный список того, на что необходимо обратить внимание. [1]

- Убедитесь, что включённый в пакет tarball-архив, не отличается от архива, распространяемого автором основной ветки разработки (если архив с исходным кодом для Debian был создан заново, создайте изменённый tarball-архив самостоятельно).
- Запустите **lintian** (см. раздел A.2.1, «`lintian`»). Он обнаружил многие часто встречающиеся проблемы. Убедитесь, что все настройки по игнорированию **lintian**, выполненные сопровождающим, обоснованы.
- Запустите **licensecheck** (часть раздела A.6.1, «`devscripts`») и убедитесь, что файл `debian/copyright` корректен и полон. Проверьте проблемы с лицензиями (напр., файлы с заголовками “All rights reserved”, либо распространяемые под несовместимой с Критериями Debian лицензией). Команда **grep -ri** поможет вам в решении этой задачи.
- Соберите пакет с помощью **pbuilder** (или любого другого схожего инструмента, см. раздел A.4.3, «`pbuilder`») для того, чтобы гарантировать, что сборочные зависимости полны.
- Проверьте `debian/control`: подготовлен ли он в соответствии с лучшими практиками (см. раздел 6.2, «Лучшие практики для `debian/control`»)? Полны ли зависимости?
- Проверьте `debian/rules`: подготовлен ли он в соответствии с лучшими практиками (см. раздел 6.1, «Лучшие практики для `debian/rules`»)? Можно ли что-либо улучшить?
- Проверьте сценарии сопровождающего (`preinst`, `postinst`, `prerm`, `postrm`, `config`): будет ли `preinst`/`postrm` работать в том случае, если зависимости не установлены? Все ли сценарии идемпотентны (т. е., можете ли вы запустить их несколько раз без каких-либо последствий)?

[1] Дополнительные проверки вы можете найти в wiki, там некоторые разработчики размещают свои собственные списки требований для поручения пакетов.

- Проверьте изменения в файлах из основной ветки разработки (либо в `.diff.gz`, либо в `debian/patches/`, либо в самом встроенном tarball-архиве `debian` для двоичных файлов). Обоснованы ли эти изменения? Правильно ли они документированы (с DEP-3 для заплат)?
- Относительно каждого файла задайте себе вопрос о том, почему этот файл находится здесь, правильно достигнут желаемый результат. Следует ли сопровождающий лучшим практикам (см. раздел 6, «Лучшие практики создания пакетов»)?
- Соберите пакеты, установите их и попробуйте поработать с эти ПО. Убедитесь, что вы можете удалить и очистить пакеты. Можно протестировать пакеты с помощью **piuparts**.

Если аудит не выявил каких-либо проблем, вы можете собрать пакет и загрузить его в Debian. Помните, что даже если вы не являетесь сопровождающим, как поручитель вы всё равно ответственны за то, что вы загружаете в Debian. Вот почему вам нужно отслеживать пакет при помощи информации из раздела 4.10, «Система отслеживания пакетов Debian».

Заметьте, что вам не нужно изменять пакет с исходным кодом для указания своего имени в файле `changelog` или в файле `control`. Поле `Maintainer` файла `control` и файла `changelog` должно сдержать имя того, кто создал пакет, т. е., того, за кого вы поручились. Так этот человек будет получить всю почту от системы отслеживания ошибок.

Вместо этого вам следует явным образом настроить **dpkg-buildpackage** на использование вашего ключа для подписи пакета. Вы можете сделать это при помощи опции `-k`:

```
dpkg-buildpackage -kИДЕНТИФИКАТОР-КЛЮЧА
```

Если вы используете **debuild** и **debsign**, вы даже можете создать постоянную настройку в файле `~/.devscripts`:

```
DEBSIGN_KEYID=ИДЕНТИФИКАТОР-КЛЮЧА
```

7.5.1.2 Поручение обновления существующего пакета

Обычно предполагают, что пакет уже прошёл полную проверку. Поэтому вместо того, чтобы производить её ещё раз, внимание обращают на проверку различий между текущей версией и новой версией, которая была подготовлена сопровождающим. Если вы самостоятельно не произвели изначальную проверку, вам, вероятно, будет нужно более детально ознакомиться с пакетом как будто тот, кто производил изначальную проверку, был недостаточно внимателен.

Чтобы проверить различия вам нужны обе версии пакета. Загрузите текущую версию пакета с исходным кодом (при помощи команды **apt-get source**) и соберите его заново (либо загрузите текущий двоичный пакет при помощи команды **aptitude download**). Загрузите поручаемый пакет с исходным кодом sponsor (обычно при помощи команды **dget**).

Прочтите новую запись в журнале изменений, она должна сообщить вам, что следует ожидать во время проверки. Основным инструментом, который вы будете использовать, является **debdiff** (предоставляется пакетом `devscripts`), вы можете запустить её, указав два пакета с исходным кодом (файлы `.dsc`), либо двумя двоичными пакетами, либо двумя файлами `.changes` (затем эта утилита сравнит все двоичные пакеты из списка `.changes`).

Если вы сравниваете пакеты с исходным кодом (исключая файлы из основной ветки разработки в случае новой версии из основной ветки разработки, например, фильтруя вывод команды **debdiff** с помощью **filterdiff -i '*/debian/*'**), вам следует понять все изменения, которые вы видите, и они должны быть соответствующим образом документированы в журнале изменений Debian.

Если всё хорошо, соберите пакет и сравните двоичные пакеты для того, чтобы проверить, что изменения пакета с исходным кодом не создаёт каких-либо неожиданных последствий (как например, некоторые файлы по ошибке оказываются потерянными, отсутствуют зависимости и т. д.).

Возможно, вам захочется обратиться к системе отслеживания пакетов (см. раздел 4.10, «Система отслеживания пакетов Debian»), чтобы убедиться в том, что сопровождающий не пропустил ничего важного. Возможно, в системе отслеживания ошибок имеются обновления перевода, которое может быть добавлено в пакет. Возможно, пакет был загружен тем, кто сопровождающим не является, и сопровождающий забыл добавить изменения, добавленные в этой загрузке, в свой пакет. Возможно, имеется критичная для выпуска ошибка, на которую не обратили внимания и которая блокирует миграцию пакета в `тестируемый выпуск`. Если обнаружите что-то, что может быть сделано (лучше), нужно сообщить сопровождающему, что он может улучшить в следующий раз, так он будет лучше понимать свою ответственность.

Если вы не нашли серьёзных проблем, загрузите новую версию. В противном случае попросите сопровождающего предоставить вам исправленную версию.

7.5.2 Поддержка новых разработчиков

Смотрите страницу о поддержке будущих разработчиков на веб-сайте Debian.

7.5.3 Обработка заявок новых сопровождающих

Пожалуйста, посмотрите Перечень для менеджеров заявок на веб-сайте Debian.

Глава 8

Интернационализация и переводы

Debian поддерживает постоянно увеличивающееся число языков. Даже если вашим родным языком является английский, и вы не говорите ни на одном другом языке, как сопровождающий вы должны знать о процессе интернационализации (сокращённо i18n, так как в слове internationalization между 'i' и 'n' 18 букв). Следовательно, даже если вам вполне подходят программы с интерфейсом на английском языке, вам следует прочесть большую часть этой главы.

Согласно Введению в интернационализацию, написанному Томохиро Кубота, I18N (интернационализация) предполагает изменение ПО или связанных технологий таким образом, чтобы это ПО потенциально могло работать на нескольких языках, поддерживало несколько обычаев и т. д., L10N (локализация) же означает реализацию конкретного языка для уже интернационализированного ПО.

l10n и i18n взаимосвязаны, но трудности, касающиеся каждого из этих процессов, весьма различны. Вообще-то не сложно сделать так, чтобы программа изменяла язык, на котором отображается текст, на основе пользовательских настроек, но перевести все эти сообщения весьма затратно по времени. С другой стороны, установить кодировку символов весьма легко, но изменить код так, чтобы можно было использовать несколько кодировок, довольно трудно.

Если оставить проблемы i18n, по поводу которых нет общего руководства, в стороне, можно видеть, что для l10n в Debian нет какой-либо центральной инфраструктуре, которая бы была сравнима с механизмом buildd для переноса на другие архитектуры. Поэтому большая часть работы выполняется вручную.

8.1 Как переводы обрабатываются в Debian

Обработка переводов текстов, содержащихся в пакете, всё ещё является задачей, которую приходится решать вручную, сам же этот процесс зависит от того, к какому виду относится текст, который вы желаете перевести.

Для перевода сообщений программ почти всегда используется инфраструктура gettext. В большинстве случаев перевод осуществляется в основной ветке разработки в таких проектах как the Free Translation Project, проект по переводу Gnome или проект по переводу KDE. Единственным централизованным ресурсом в Debian является Центральная статистика переводов Debian, где вы можете найти некоторую статистику относительно файлов переводов, входящих в текущие пакеты, но для облегчения процесса перевода нет никакой инфраструктуры.

Попытка перевести описания пакетов предпринимается уже давно, инструменты, использующие описания пакетов предоставляют довольно небольшую поддержку переводов (напр., только APT может использовать эти переводы при условии правильной настройки). Сопровождающим не нужно предпринимать каких-либо специальных действия для поддержки переведённых описаний пакетов; переводчикам же следует пользоваться проектом по переводу описаний пакетов Debian (DDTP).

Для шаблонов `debconf`, сопровождающим следует использовать пакет `po-debconf` для облегчения работы переводчиков, которые могут использовать DDTP для выполнения своей работы (но французская и бразильская команды не используют DDTP). Некоторая статистика может быть найдена и на сайте DDTP (о том, что фактически переведено), и на сайте центральной статистики переводов Debian (о том, что интегрировано в пакеты).

Для веб-страницы у каждой команды l10n имеется доступ к релевантной системе управления версиями, статистика доступна на сайте центральной статистики переводов Debian.

Для общей документации относительно Debian процесс перевода более или менее совпадает с процессом перевода веб-страниц (переводчики имеют доступ к системе управления версиями), но страницы со стати-

стикой отсутствуют.

Для документации по пакетам (страницы man, документы info, другие форматы), необходимо сделать почти всё.

Перевод документации в проект KDE осуществляется по тому же принципу, как перевод сообщений программ.

В специальном репозитории системы управления версиями предпринята попытка перевода страниц man, непосредственно связанных с Проектом Debian.

8.2 ЧаВО по I18N и L10N для сопровождающих

Это список проблем i18n и l10n, с которыми могут столкнуться сопровождающие. Читая данный документ, помните, что в настоящее время никакого действительного согласия по этим вопросам в Debian нет, и что в документе содержатся только советы. Если у вас имеются более подходящие идеи для некоторой данной проблемы, либо если вы не согласны с чем-то, не стесняйтесь сообщить об этом, это поспособствует улучшению данного документа.

8.2.1 Как перевести некоторый данный текст

Чтобы перевести описания пакетов или шаблоны `debconf`, вам не нужно ничего делать; инфраструктура DDTP перешлёт материал для перевода добровольцам, это совершенно не требует вашего вмешательства.

Касательно всех остальных материалов (файлы gettext, страницы руководства man и другая документация) лучшим решением будет поместить ваш текст где-нибудь в сети Интернет и попросить в списке рассылки debian-i18n перевести его на различные языки. Некоторые члены команд переводчиков подписаны на этот список рассылки, они позаботятся о переводе и проверке. Когда они закончат с этим, в вашем почтовом ящике вы найдёте ваш переведённый документ.

8.2.2 Как проверить перевод

Время от времени кто-то переводит различные тексты из ваших пакетов, эти люди будут просить вас включить их перевод в пакет. Это может быть проблемой, если вы не владеете данным языком. Лучше всего выслать данный документ в соответствующий список рассылки l10n и попросить о проверке. Когда это будет сделано, вы будете более уверены в качестве перевода, тогда уже можете спокойно добавлять его в ваш пакет.

8.2.3 Как обновить перевод

Если у вас имеются переводы какого-то текста, то всякий раз, когда вы обновляете оригинал, вам следует попросить предыдущего переводчика обновить его перевод для включения новых изменений. Помните, что эта задача требует некоторого времени; по меньшей мере, нужна одна неделя, чтобы обновить и проверить весь перевод.

Если переводчик не отвечает, вы можете попросить о помощи в соответствующем списке рассылки l10n. Если ничего не получилось, не забудьте добавить предупреждение в переведённый документ о том, что данный перевод устарел, и что читателю по возможности следует обратиться к оригинальному документу.

Избегайте полного удаления перевода даже в случае, если он сильно устарел. Для тех, что не говорит на английском устаревшая документация зачастую лучше, чем её полное отсутствие.

8.2.4 Как работать с отчётом об ошибке, касающемся перевода

Лучшим решением может состоять в следующем: отметьте сообщение об ошибке как пересланное в основную ветку разработки, затем перешлите её предыдущему переводчику и соответствующей команде локализации (используя соответствующий список рассылки debian-l10n-XXX).

8.3 ЧаВО по I18N и L10N для переводчиков

Читая данный документ, помните, что в Debian отсутствует какая-либо общая процедура, определяющая работу над переводами, в любом случае вам следует взаимодействовать с вашей командой и сопровождающим пакета.

8.3.1 Как помочь с переводом

Выберите, что бы вы хотели перевести, убедитесь, что никто не работает над переводом этого документа (используя ваш список рассылки debian-l10n-XXX), переведите, получите отзывы от других носителей языка в вашем списке рассылки l10n, предоставьте перевод сопровождающему пакета (см. следующий пункт).

8.3.2 Как предоставить перевод для добавления его в пакет

Убедитесь, что перевод верен (попросив проверить его в вашем списке рассылки l10n) до отправки его для включения в пакет. Это сохранит всем время и позволит избежать хаоса, когда в сообщениях об ошибках имеются несколько версий одного и того же документа.

Лучшим решением будет отправить обычное сообщение об ошибке в данном пакете, содержащее перевод. Убедитесь, что вы используете метку 'PATCH', а важность ошибки не выше, чем 'wishlist', так как отсутствие перевода не мешает программе работать.

8.4 Лучшие текущие практики, касающиеся локализации

- Как сопровождающему вам никогда не следует редактировать переводы (даже для изменения форматирования), не спросив об этом в соответствующем списке рассылки l10n. Вы рискуете, например, испортить кодировку файлов. Более того, то, что вы считаете ошибкой, может оказаться правильным (или даже необходимым) для данного языка.

- Как переводчику, если вы находите ошибку в оригинальном тексте, вам следует сообщить об этом. Переводчики часто являются более внимательными читателями некоторого данного текста, и если они не сообщают о найденных ошибках, то о них не сообщит никто.

- В любом случае помните, что главная проблема l10n состоит в том, что для локализации требуется взаимодействие нескольких людей, и что из-за непонимания очень и очень легко начать войну по поводу небольших проблем. Поэтому если у вас возникли проблемы с вашим собеседником, попросите о помощи в соответствующем списке рассылки l10n, в списке рассылки debian-i18n или даже в debian-devel (но остерегайтесь, обсуждения l10n в этом списке рассылки часто переходят в настоящие войны :)

- В любом случае взаимодействие может быть достигнут только благодаря **взаимному уважению**.

Приложение A

Обзор инструментов Debian для сопровождающего

Данный раздел содержит краткий обзор доступных сопровождающим инструментов. Приведённая ниже информация ни в коем мере не является полной или окончательной, она представляет собой лишь руководство по некоторым более популярным утилитам.

Инструменты сопровождающего Debian предназначены для облегчения работы разработчиком и освобождения их времени для решения критических задач. Как говорит Ларри Уолл, существует более одного способа сделать это.

Некоторые предпочитают использовать высокоуровневые инструменты для сопровождения пакетов, некоторые же нет. Официально Debian это без разницы; подходит любой инструмент, которые делает свою работу. Следовательно, данный раздел не предполагается для того, чтобы навязать всякому то, какие инструменты следует использовать или как следует выполнять свои обязанности по сопровождению пакетов. Также не предполагается рекомендовать какой-либо конкретный инструмент вплоть до исключения конкурирующего инструмента.

Большинство описаний этих пакетов берётся из самих фактических описаний пакетов. Дополнительная информация может быть найдена в самой документации по пакету. Также вы можете получить дополнительную информацию при помощи команды **apt-cache show** ***имя-пакета***.

A.1 Базовые инструменты

Следующие инструменты весьма нужны любому сопровождающими.

A.1.1 `dpkg-dev`

`dpkg-dev` содержит инструменты (включая **dpkg-source**), необходимые для распаковки, сборки и загрузки пакетов Debian с исходным кодом. Эти утилиты предоставляют базовую, низкоуровневую функциональность, необходимую для создания пакетов и для манипуляции ими; как таковые, эти утилиты являются необходимыми любому сопровождающему Debian.

A.1.2 `debconf`

`debconf` предоставляет единообразный интерфейс для интерактивной настройки пакетов. Его пользовательских интерфейс независим, что позволяет конечным пользователям настраивать пакеты при помощи текстового интерфейса, интерфейса HTML, либо диалогового интерфейса. Новые варианты интерфейса могут быть добавлены в виде модулей.

Документацию для этого пакета вы можете найти в пакете `debconf-doc`.

Многие считают, что эта система должна использоваться для всех пакетов, которым требуется интерактивная настройка; см. раздел 6.5, «Управление настройкой с помощью `debconf`». В настоящее время Политика Debian не требует использовать `debconf`, но это может быть изменено в будущем.

A.1.3 `fakeroot`

`fakeroot` имитирует привилегии суперпользователя. Это позволяет вам собирать пакеты без привилегий суперпользователя (обычно пакеты хотят установить файлы, владельцем которых является суперпользователь). Если у вас установлен `fakeroot`, `dpkg-buildpackage` будет автоматически использовать его.

A.2 Инструменты для проверки пакетов на предмет ошибок и соответствия стандартам

Согласно свободному онлайн словарю вычислительной техники (FOLDOC), lint' является обработчиком языка Unix C, который выполняет больше проверок кода, чем обычные компиляторы C. Инструменты для проверки пакетов помогают сопровождающим, автоматически находя распространённые ошибки и нарушения Политики в пакетах.

A.2.1 `lintian`

`lintian` анализирует пакеты Debian и выдаёт информацию об ошибках и нарушениях Политики. Он содержит автоматические проверки множества аспектов Политики Debian, а также некоторые проверки на наличие распространённых ошибок.

Периодически вам следует получить наиболее свежую версию `lintian` из `нестабильного` выпуска и проверять все ваши пакеты. Заметьте, что опция `-i` предоставляет подробное объяснение того, что означает каждая ошибка или предупреждение, на каком пункте Политики они основываются, а иногда и то, как вы можете исправить проблему.

Для получения дополнительной информации о том, как и когда использовать Lintian, обратитесь к разделу 5.3, «Тестирование пакета».

Также вы можете посмотреть обзор всех проблем, о которых было сообщено Lintian для ваших пакетов, по адресу `https://lintian.debian.org/`. Эти отчёты содержат наиболее свежий вывод **lintian** для всего разрабатываемого выпуска (`нестабильного` выпуска).

A.2.2 debdiff

debdiff (из пакета `devscripts`, раздел A.6.1, «`devscripts`») сравнивает списки файлов и управляющие файлы двух пакетов. Это простая проверка на наличие регрессий, поскольку она позволяет вам заметить, что число двоичных пакетов изменилось с момента последней загрузки, либо если что-то было изменено в управляющем файле. Конечно, некоторые изменения, о которых сообщает эта утилита, не являются чем-то плохим, но она может помочь вам предотвратить различные случайные проблемы.

Вы можете запустить её, указав два двоичных пакета:

```
debdiff package_1-1_arch.deb package_2-1_arch.deb
```

Или даже два файла changes:

```
debdiff package_1-1_arch.changes package_2-1_arch.changes
```

Дополнительную информацию см. в debdiff(1).

A.3 Помощники для `debian/rules`

Инструменты сборки пакетов делают написание файла `debian/rules` значительно проще. Дополнительную информацию о том, почему желательно или не желательно использовать их, см. в разделе 6.1.1, «Сценарии-помощники».

A.3.1 `debhelper`

`debhelper` представляет собой набор программ, которые могут использовать в `debian/rules` для автоматизации распространённых задач, связанных со сборкой двоичных пакетов Debian. `debhelper` включает в себя программы для установки различных файлов в ваши пакеты, сжатия файлов, исправления прав доступа к файлам, а также интеграции вашего пакета в систему меню Debian.

В отличии от других подходов, `debhelper` разбит на несколько небольших и простых команд, которые работают заодно. Как таковой, этот подход позволяет осуществлять более тщательное управление, чем то предоставляет другими инструментами для debian/rules.

Имеется ряд небольших дополнительных пакетов `debhelper`, которые слишком мелки, чтобы описывать их здесь. Вы можете посмотреть список этих программ, выполнив `apt-cache search ^dh-`.

A.3.2 `dh-make`

Пакет `dh-make` содержит **dh_make**, программу, которая создаёт основу файлов, которые необходимы для сборки пакета Debian из дерева исходного кода. Как то предполагается в названии этой программы, **dh_make** является новой версией `debmake`, её файлы-шаблоны используют программы **dh_*** из `debhelper`.

Хотя файлы rules, порождаемые **dh_make**, вообще-то являются достаточной основой для создания рабочего пакета, они всё равно представляют собой лишь основу: на сопровождающем всё ещё лежит груз задачи по тонкой настройке порождённых файлов и сборке пакета, целиком соответствующего Политике и работающего.

A.3.3 `equivs`

`equivs` представляет собой ещё один пакет для создания пакетов. Часто он предлагается для локального использования, если вам нужно сделать пакет просто для удовлетворения зависимостей. Также он иногда используется при создании „метапакетов“, это пакеты, чья цель состоит лишь в том, чтобы зависеть от других пакетов.

A.4 Сборщики пакетов

Следующие пакеты помогают в процессе сборки пакета, по сути управляя **dpkg-buildpackage**, а также решая поддерживающие задачи.

A.4.1 `git-buildpackage`

`git-buildpackage` предоставляет возможность введения или импорта пакетов Debian с исходным кодом в репозиторий Git, сборки пакета Debian из репозитория Git, а также помогает в интеграции изменений из основной ветки разработки в этот репозиторий.

Эти утилиты предоставляют инфраструктуру для облегчения использования Git сопровождающими Debian. Это позволяет хранить отдельные ветки Git пакета для `стабильного`, `нестабильного` и возможно `экспериментальн` выпусков, а также иметь все другие преимущества системы контроля версий.

A.4.2 `debootstrap`

Пакет `debootstrap` и соответствующий сценарий позволяют вам произвести начальную установку базовой системы Debian в любую часть файловой системы. Под базовой системой мы подразумеваем минимальное число пакетов, необходимых для работы и установки остальной системы.

Иметь подобную систему весьма полезно. Например, вы можете сделать **chroot** в эту систему и проверить ваши сборочные зависимости. Либо вы можете проверить то, как ведут себя ваши пакеты при установке на базовую систему. Сборщики chroot используют этот пакет; см. об этом ниже.

A.4.3 `pbuilder`

`pbuilder` создаёт систему в chroot и собирает пакет внутри chroot. Это полезно для проверки того, то сборочные зависимости пакета указаны верно, а также для того, чтобы убедиться, что ненужных и неправильных сборочных зависимостей в получившемся пакете нет.

Связанный пакет, `cowbuilder`, ускоряет процесс сборки с помощью файловой системы COW на любой стандартной файловой системе Linux.

A.4.4 sbuild

sbuild представляет собой другой автоматизированный сборщик. Он также может использовать окружения chroot. Он может использоваться как отдельно, так и как часть сетевого, распределённого окружения. В последнем случае он является частью системы, используемой теми, кто занимается переносом, для сборки двоичных пакетов для всех доступных архитектур. Дополнительную информацию см. в разделе 5.10.3.3, «wanna-build», систему в действии можно посмотреть по адресу https://buildd.debian.org/.

A.5 ПО для загрузки пакетов

Следующие пакеты помогут автоматизировать или упростить процесс загрузки пакетов в официальный архив.

A.5.1 dupload

dupload представляет собой пакет и сценарий для автоматической загрузки пакетов Debian в архив Debian, для записи журнала загрузки, а также отправки сообщений электронной почты о загрузке пакета. Вы можете настроить его на использование новых мест или методов загрузки.

A.5.2 dput

Пакет и сценарий dput выполняет по большей части то же, что и dupload, но по-другому. Он имеет некоторые дополнительные возможности по сравнению с dupload, такие как способность проверять подпись GnuPG и контрольные суммы до загрузки, а также возможность запуска **dinstall** в холостом режиме после загрузки.

A.5.3 dcut

Сценарий **dcut** (часть пакета dput, раздел A.5.2, «dput») помогает удалять файлы из каталога загрузки на ftp.

A.6 Автоматизация сопровождения пакетов

Следующие инструменты помогают автоматизировать различные задачи по сопровождению пакетов от добавления записей в журнал изменений или строк подписи и поиска ошибок в Emacs до использования наиболее свежего и исключительно официального файла config.sub.

A.6.1 devscripts

devscripts представляет собой пакет, содержащий обёртки и инструменты, которые весьма полезны для сопровождения ваших пакетов Debian. К примеру, в нём имеются сценарии **debchange** и **dch**, которые управляют вашим файлом debian/changelog из командной строки, **debuild**, который представляет собой обёртку вокруг **dpkg-buildpackage**. Утилита **bts** также очень полезна для обновления статуса сообщений об ошибках из командной строки. **uscan** может использоваться для отслеживания новых версий основной ветки разработки для ваших пакетов. **debrsign** может использоваться для выполнения удалённого подписывания вашего пакета до его загрузки, что очень удобно если машина, на которой вы собрали ваш пакет, отличается от той, где находятся ваши ключи GPG.

Полный список доступных сценариев см. в руководстве devscripts(1).

A.6.2 autotools-dev

autotools-dev включает в себя лучшие практики для тех, кто сопровождает пакеты, использующие **autoconf** и/или **automake**. Также в этом пакете содержаться канонические файлы config.sub и config.guess, которые, как это известно, работают на всех переносах Debian.

A.6.3 dpkg-repack

dpkg-repack создаёт пакет Debian из пакета, который уже был установлен. Если в пакете были произведены какие-либо изменения в то время когда он был распакован (напр., были изменены файлы в `/etc`), новый пакет наследует все изменения.

Эта утилита позволяет легко копировать пакеты с одного компьютера на другой, либо заново создавать пакеты, которые были установлены в вашей системе, но более нигде не доступны, либо для сохранения текущего состояния пакета до того, как вы его обновите.

A.6.4 alien

alien преобразует пакеты между различными форматами пакетов, включая пакеты Debian, RPM (RedHat), LSB (Linux Standard Base), Solaris, и Slackware.

A.6.5 dpkg-dev-el

`dpkg-dev-el` является пакетом Emacs lisp, который предоставляет помощь в редактировании некоторых файлов в каталоге `debian` вашего пакета. Например, в этом пакете имеются полезные функции для вывода списка текущих ошибок пакета, а также для завершения последней записи в файле `debian/changelog`.

A.6.6 dpkg-depcheck

dpkg-depcheck (из пакета `devscripts`, раздел A.6.1, «devscripts») запускает команду в окружении **strace** для определения всех пакетов, которые используются вызванной командой.

Для пакетов Debian это весьма полезно, если вам необходимо сформировать строку `Build-Depends` для вашего нового пакета: запуск процесса сборки через **dpkg-depcheck** предоставит вам приблизительный список сборочных зависимостей. Например:

```
dpkg-depcheck -b debian/rules build
```

dpkg-depcheck также может использоваться для проверки зависимостей времени исполнения, особенно в том случае, если ваш пакет использует exec(2) для запуска других программ.

Дополнительную информацию см. в dpkg-depcheck(1).

A.7 Инструменты для переноса

Следующие инструменты полезны для занимающихся переносом и кросс-компиляцией.

A.7.1 dpkg-cross

`dpkg-cross` является инструментом для установки библиотек и заголовочных файлов для перекрёстной компиляции схожим с `dpkg` способом. Более того, функциональность **dpkg-buildpackage** и **dpkg-shlibdeps** была улучшена в плане поддержки перекрёстной компиляции.

A.8 Документация и информацию

Следующие пакеты предоставляют информацию для сопровождающих, либо помогают им в сборке документации.

A.8.1 docbook-xml

`docbook-xml` предоставляет DocBook XML DTD, который обычно используется для подготовки документации по Debian (как более старая документация debiandoc SGML DTD). Например, данное руководство написано в DocBook XML.

Пакет `docbook-xsl` предоставляет файлы XSL для сборки и форматирования стиля исходных файлов в различные форматы вывода. Для того, чтобы использовать стили XSL, вам потребуется процессор XSLT, такой как `xsltproc`. Документация для стилей может быть найдена в пакетах `docbook-xsl-doc-*`.

Чтобы создать PDF из FO, вам потребуется процессор FO, такой как `xmlroff` или `fop`. Другим инструментом для создания PDF из DocBook XML является `dblatex`.

A.8.2 debiandoc-sgml

debiandoc-sgml предоставляет DebianDoc SGML DTD, который широко используется для подготовки документации по Debian, но в настоящее время устарел (следует использовать пакет docbook-xml). Кроме того, он предоставляет сценарии для сборки и изменения стиля оформления исходного файла в других форматах вывода.

Документация для DTD может быть найдена в пакете debiandoc-sgml-doc.

A.8.3 debian-keyring

Содержит открытые ключи GPG разработчиков и сопровождающих Debian. Дополнительную информацию см. в разделе 3.2.2, «Сопровождение вашего открытого ключа» и документации по пакету.

A.8.4 debian-el

debian-el предоставляет режим Emacs для просмотра двоичных пакетов Debian. Это позволяет вам исследовать пакет без его распаковки.