

Добавление мультибиблиотечности в Slackware x86_64

Статья содержит инструкции по созданию мультибиблиотечной Slackware64.

Мультибиблиотечная 64-битная система Linux способна исполнять как 32-битные, так и 64-битные программы. [Стандарт иерархии файловой системы](#) документирует оптимальный метод достижения чистого разделения между 64-битным и 32-битным программами на одной системе. Начав разработку «Slackware64» (официального порта на архитектуру [x86_64](#)), мы решили придерживаться этого стандарта. Поэтому Slackware64 была настроена на поиск 64-битных библиотек в каталогах `/lib64` и `/usr/lib64`. Вот почему я называю Slackware64 готовой к мультибиблиотечности — хотя 32-битные библиотеки будут искаться в `/lib` и `/usr/lib`, в поставке Slackware64 нет 32-битных программ. Требуется сделать ещё один шаг (вам, пользователю), прежде чем Slackware64 можно будет назвать «мультибиблиотечной».

Это достигается следующим образом:

1. Сначала нужно переключиться на мультибиблиотечные версии
 - *glibc* (такой *glibc*, который поддерживает исполнение как 32-ных, так и 64-битных бинарных файлов), и
 - *gcc* (способный *компилировать* в 32-битные бинарные файлы наравне с 64-битными).
2. Затем взять из 32-битной Slackware библиотеки и вместе с их 64-битными версиями установить их в 64-битной Slackware, что завершит процесс создания 32-битного программного слоя совместимости.

Когда вышла Slackware64, она имела преимущество перед прочими существовавшими 64-битными «форками». Последние добавляли 32-битный слой совместимости путём перекомпиляции множества своих пакетов как 32-битных бинарников. С другой стороны, Slackware — это дистрибутив, состоящий из 32-битного и 64-битного выпуска, разрабатываемых параллельно. Это означает, что вам не нужно компилировать 32-битные пакеты с нуля для добавления мультибиблиотечности в 64-битную систему. Можно просто взять их из дерева пакетов 32-битной Slackware!

Это было одной из причин не добавлять в Slackware64 полную мультибиблиотечность — мы создали необходимые условия, но если пользователю нужна мультибиблиотечность, то ему нужно действовать самостоятельно.

В разделе [ниже](#) я объясню, как взять 32-битный пакет Slackware (скажем, пакет «mesa») и переупаковать его содержимое в пакет «mesa-compat32», который можно установить непосредственно в Slackware64.

Slackware для архитектуры x86_64 (или для краткости «Slackware64») является чистой 64-битной операционной системой, но легко обновляется до мультибиблиотечной. *Изначально в Slackware64 возможно компилировать и исполнять только 64-битные бинарные файлы.*

Преимущества мультибиблиотечной системы

Я приведу несколько примеров программ, которые требуют поддержки мультибиблиотечности на 64-битной Slackware, поскольку они не запускаются или не компилируются на Slackware64 без 32-битного слоя совместимости:

- [Wine](#)

Большинство программ для Windows по-прежнему 32-разрядные, для их запуска на Linux

в Wine требуется 32-битная версия Wine.

- [VirtualBox](#)

Популярное программное обеспечение для виртуальных машин. Несмотря на открытый (частично) исходный код, все ещё нуждается в 32-битных библиотеках на 64-битной Slackware.

- [Steam](#)

Очень популярная игровая платформа по-прежнему требует [32-разрядного клиента](#). Большинство доступных игр также 32-разрядные.

- [Skype](#), [Citrix client](#), ...

Это проприетарные программы с закрытыми исходными кодами. Мы вынуждены зависеть от их разработчиков в части доступности 64-битных бинарников. Для этих программ пока подобного не случилось.

К счастью, поддержка 64 бит становится всё более распространённой. Довольно долго больным местом был Adobe, но в конце концов они выпустили свой плагин Flash для браузера в 64-битном варианте. Sun (ныне поглощённый Oracle-ом) выпустила 64-битную версию своего плагина Java для браузера. Эти два события послужили спусковым крючком для начала работы над Slackware64.

Получение мультибиблиотечных пакетов

Можете загрузить набор мультибиблиотечных пакетов и сценариев с моего сайта:

<http://slackware.com/~alien/multilib/> .

Наряду с несколькими файлами README (эта статья вики — расширенная версия одного из тех файлов) ниже каталога верхнего уровня «*multilib*» находятся подкаталоги для каждого 64-битного выпуска Slackware. Там же расположен каталог «*source*». Он содержит исходные тексты и сценарии SlackBuild.

Что вам действительно нужно — бинарные пакеты — доступны в каталоге

<номер_выпуска_slackware> ниже каталога верхнего уровня. Каждый такой каталог содержит подкаталог «*slackware64-compat32*», где находится основной набор конвертированных 32-битных пакетов Slackware, готовых к установке на 64-битной Slackware.

Поддержка мультибиблиотечности в актуальном состоянии

Для поддержки актуальности советую обратить внимание на [ChangeLog](#) ([лента RSS](#)), которые я поддерживаю для моих мультибиблиотечных пакетов. Обычно обновляются пакеты *glibc* and *gcc* в течении дня после обновления их в Slackware.

Автоматизация:

1. Обратите внимание на [compat32pkg](#) от Sébastien Ballet, автоматизирующий процесс наподобие slackpkg.
2. Если для управления пакетами предпочитаете slackpkg, стоит обратить внимание на [slackpkg+](#), расширение для slackpkg, управляющее пакетами, установленными из сторонних хранилищ, включая мультибиблиотечные пакеты. При правильной настройке,

поддержка сводится к запуску:

```
# slackpkg update
# slackpkg upgrade multilib
# slackpkg install multilib
```

Последняя команда покажет новые «compat32» пакеты, например недавно добавленные llvm-compat32 и orc-compat32.

- Вот как выглядит типичная настройка для Slackware-current с использованием тестового хранилища KDE от Alien BOB. PKGS_PRIORITY задает приоритет пакетов gcc и glibc из мультибиблиотечного хранилища перед официальным деревом Slackware. Ключевое слово «multilib», которое задает имя хранилища, должно соответствовать используемому в приведенных выше командах «slackpkg». Слово «multilib» выбрано произвольно, с тем же успехом можно было везде использовать «compat32».

Ниже приведено примерное содержимое файла «/etc/slackpkg/slackpkgplus.conf»:

```
SLACKPKGPLUS=on
VERBOSE=1
ALLOW32BIT=off
USEBL=1
WGETOPTS="--timeout=5 --tries=1"
GREYLIST=on
PKGS_PRIORITY=( multilib restricted alienbob ktown )
REPOPLUS=( slackpkgplus multilib restricted alienbob ktown )
MIRRORPLUS['multilib']=http://bear.alienbase.nl/mirrors/people/alien/multilib/current/
MIRRORPLUS['alienbob']=http://bear.alienbase.nl/mirrors/people/alien/sbrepos/current/x86_64/
MIRRORPLUS['restricted']=http://bear.alienbase.nl/mirrors/people/alien/restricted_sbrepos/current/x86_64/
MIRRORPLUS['ktown']=http://bear.alienbase.nl/mirrors/alien-kde/current/latest/x86_64/
MIRRORPLUS['slackpkgplus']=http://slakfinder.org/slackpkg+/
```

Включение мультибиблиотечной поддержки в Slackware64

Примерные инструкции

Этот раздел содержит основные инструкции по добавлению полной поддержки мультибиблиотечности в вашу систему Slackware64. Если хотите разобраться в процессе более подробно или нужна информация о том, как компилировать 32-битные программы в Slackware64, прочтите и последующие разделы.

Обратите внимание, знак «#» в начале команд означает *приглашение суперпользователя root*.

- Загрузите пакеты с моего сайта (я давал ссылку в [предыдущем разделе](#)), в примере ниже

использована ссылка на зеркало. Допустим, используется Slackware 14.2. Выполните:

```
# SLACKVER=14.2
# mkdir multilib
# cd multilib
# lftp -c "open http://bear.alienbase.nl/mirrors/people/alien/multilib/
; mirror -c -e ${SLACKVER}"
# cd ${SLACKVER}
```

- Обновите пакеты 64-битной Slackware «*gcc*» и «*glibc*» моими мультибиблиотечными версиями.
Выполните команду

```
# upgradepkg --reinstall --install-new *.t?z
```

после перехода в каталог с загруженными пакетами.

Эта команда также установит дополнительный пакет «*compat32-tools*».

- Если загрузили и каталог *slackware64-compat32* (мой пример команды «*lftp*» делает это) — вам повезло, потому что я уже выполнил преобразование 32-битных пакетов! Всё, что нужно, это выполнить команду:

```
# upgradepkg --install-new slackware64-compat32/*-compat32/*.t?z
```

которая установит все 32-битные пакеты Slackware (или обновит их, если установлены более старые их версии, например, при обновлении до новой версии Slackware). Это всё!

- Если каталог *slackware64-compat32* отсутствует, то либо его не загрузили, либо зеркало его не содержало. В этом случае придётся выполнить конвертацию 32-битных пакетов самостоятельно. Несложно, но займёт какое-то время:
 - Быстрее всего, если у вас есть локальный каталог с оригинальными 32-битными пакетами Slackware (*локальное зеркало*). Купившие официальный DVD Slackware могут использовать его: он двусторонний, и 32-битная Slackware на одной из сторон. В этом примере будем считать, что локальное зеркало дерева 32-битной Slackware доступно в каталоге «*/home/ftp/pub/slackware/slackware-14.2/slackware/*». Непосредственно в нём должны быть подкаталоги «*a*», «*ap*», «*d*», «*l*», «*n*», «*x*». (Если смонтировали DVD Slackware, возможный каталог «*/media/SlackDVD/slackware/*», но в примерах команд ниже я его не буду использовать).
 - Создайте новый пустой каталог (назовём его «*slackware64-compat32*») и перейдите в него:

```
# mkdir slackware64-compat32 ; cd slackware64-compat32
```

- Выполните следующую команду для создания набора 32-битных пакетов совместимости, используя в качестве параметра каталог с официальными 32-битными пакетами Slackware:

```
# massconvert32.sh -i
/home/ftp/pub/slackware/slackware-14.2/slackware/
```

- Предыдущий шаг займёт какое-то время. По завершении установите 90Мб свежесконвертированных 32-битных пакетов Slackware, созданных в подкаталогах *текущего каталога*:

```
# upgradepkg --install-new *-compat32/*.t?z
```

- Закончили! Теперь можно загружать, устанавливать и запускать 32-битные программы. Не очень сложно, не так ли?

Если используете пакетный менеджер, как *slackpkg*, придётся добавить все пакеты *glibc* и *gcc* в его чёрный список. Если не принять подобных мер предосторожности, есть риск, что пакетный менеджер случайно заменит мультибиблиотечные версии версиями из оригинальной чисто 64-битной Slackware!

Если используете Slackware 13.37 или новее, тогда *slackpkg* поддерживает регулярные выражения в файле чёрного списка. В этом случае будет достаточно одной строки в */etc/slackpkg/blacklist* для внесения в чёрный список всех моих пакетов (включая мультибиблиотечные пакеты *gcc* и *glibc* и все пакеты *compat32*):

```
[0-9]+alien  
[0-9]+compat32
```

С другой стороны, если используете расширение *slackpkg* [slackpkg+](#), то **не** включайте эти пакеты в чёрный список, иначе *slackpkg+* не сможет ими управлять!

Если используете Slackware 13.1 или новее и загрузили пакет *compat32-tools* для этого выпуска, сценарий *massconvert32.sh* может загрузить 32-битные пакеты Slackware с удалённого веб-сервера, вместо локального зеркала Slackware или DVD. Используйте параметр «-u» для указания URL, как в примере ниже:

```
# massconvert32.sh -u http://someserver.org/path/to/slackware-14.2/slackware
```

Подробные инструкции

Обновление *glibc* и *gcc*

Следующие пакеты *glibc/gcc* являются заменой, *не добавлением*, к стандартным пакетам Slackware. Для обновления до моих мультибиблиотечных версий пакетов *gcc* и *glibc* используйте программу «*upgradepkg*». Они нужны для исполнения (*glibc*) и сборки (*gcc*) 32-битных программ на компьютере с 64-битной Slackware:

Slackware64 13.0

- Комплект компиляторов *gcc*:
 - *gcc-4.3.3_multilib-x86_64-4alien.txz*
 - *gcc-g++-4.3.3_multilib-x86_64-4alien.txz*
 - *gcc-gfortran-4.3.3_multilib-x86_64-4alien.txz*
 - *gcc-gnat-4.3.3_multilib-x86_64-4alien.txz*
 - *gcc-java-4.3.3_multilib-x86_64-4alien.txz*
 - *gcc-objc-4.3.3_multilib-x86_64-4alien.txz*

- Библиотеки GNU libc:
 - glibc-2.9_multilib-x86_64-3alien.txz
 - glibc-i18n-2.9_multilib-x86_64-3alien.txz
 - glibc-profile-2.9_multilib-x86_64-3alien.txz
 - glibc-solibs-2.9_multilib-x86_64-3alien.txz
 - glibc-zoneinfo-2.9_multilib-noarch-3alien.txz

Slackware64 13.1

- Комплект компиляторов gcc:
 - gcc-4.4.4_multilib-x86_64-1alien.txz
 - gcc-g++-4.4.4_multilib-x86_64-1alien.txz
 - gcc-gfortran-4.4.4_multilib-x86_64-1alien.txz
 - gcc-gnat-4.4.4_multilib-x86_64-1alien.txz
 - gcc-java-4.4.4_multilib-x86_64-1alien.txz
 - gcc-objc-4.4.4_multilib-x86_64-1alien.txz
- Библиотеки GNU libc:
 - glibc-2.11.1_multilib-x86_64-3alien.txz
 - glibc-i18n-2.11.1_multilib-x86_64-3alien.txz
 - glibc-profile-2.11.1_multilib-x86_64-3alien.txz
 - glibc-solibs-2.11.1_multilib-x86_64-3alien.txz
 - glibc-zoneinfo-2.11.1_multilib-noarch-3alien.txz

Slackware64 13.37

- Комплект компиляторов gcc:
 - gcc-4.5.2_multilib-x86_64-2alien.txz
 - gcc-g++-4.5.2_multilib-x86_64-2alien.txz
 - gcc-gfortran-4.5.2_multilib-x86_64-2alien.txz
 - gcc-gnat-4.5.2_multilib-x86_64-2alien.txz
 - gcc-java-4.5.2_multilib-x86_64-2alien.txz
 - gcc-objc-4.5.2_multilib-x86_64-2alien.txz
- Библиотеки GNU libc:
 - glibc-2.13_multilib-x86_64-7alien.txz
 - glibc-i18n-2.13_multilib-x86_64-7alien.txz
 - glibc-profile-2.13_multilib-x86_64-7alien.txz
 - glibc-solibs-2.13_multilib-x86_64-7alien.txz

Slackware64 14.0

- Комплект компиляторов gcc:
 - gcc-g++-4.7.1_multilib-x86_64-1alien.txz
 - gcc-gfortran-4.7.1_multilib-x86_64-1alien.txz
 - gcc-gnat-4.7.1_multilib-x86_64-1alien.txz
 - gcc-go-4.7.1_multilib-x86_64-1alien.txz
 - gcc-java-4.7.1_multilib-x86_64-1alien.txz
 - gcc-objc-4.7.1_multilib-x86_64-1alien.txz
- Библиотеки GNU libc:

- glibc-2.15_multilib-x86_64-9alien.txz
- glibc-i18n-2.15_multilib-x86_64-9alien.txz
- glibc-profile-2.15_multilib-x86_64-9alien.txz
- glibc-solibs-2.15_multilib-x86_64-9alien.txz

Slackware64 14.1

- До тех пор, пока отсутствует отдельный каталог «*current*», используйте файлы из каталога для наиболее свежего стабильного выпуска.
- Комплект компиляторов gcc:
 - gcc-4.8.2_multilib-x86_64-1alien.txz
 - gcc-g++-4.8.2_multilib-x86_64-1alien.txz
 - gcc-gfortran-4.8.2_multilib-x86_64-1alien.txz
 - gcc-gnat-4.8.2_multilib-x86_64-1alien.txz
 - gcc-go-4.8.2_multilib-x86_64-1alien.txz
 - gcc-java-4.8.2_multilib-x86_64-1alien.txz
 - gcc-objc-4.8.2_multilib-x86_64-1alien.txz
- Библиотеки GNU libc:
 - glibc-2.17_multilib-x86_64-10alien.txz
 - glibc-i18n-2.17_multilib-x86_64-10alien.txz
 - glibc-profile-2.17_multilib-x86_64-10alien.txz
 - glibc-solibs-2.17_multilib-x86_64-10alien.txz

Slackware64 14.2

- Комплект компиляторов gcc:
 - gcc-5.3.0_multilib-x86_64-3alien.txz
 - gcc-g++-5.3.0_multilib-x86_64-3alien.txz
 - gcc-gfortran-5.3.0_multilib-x86_64-3alien.txz
 - gcc-gnat-5.3.0_multilib-x86_64-3alien.txz
 - gcc-go-5.3.0_multilib-x86_64-3alien.txz
 - gcc-java-5.3.0_multilib-x86_64-3alien.txz
 - gcc-objc-5.3.0_multilib-x86_64-3alien.txz
- Библиотеки GNU libc:
 - glibc-2.23_multilib-x86_64-2alien.txz
 - glibc-i18n-2.23_multilib-x86_64-2alien.txz
 - glibc-profile-2.23_multilib-x86_64-2alien.txz
 - glibc-solibs-2.23_multilib-x86_64-2alien.txz

Slackware64 current

- Комплект компиляторов gcc:
 - gcc-7.1.0_multilib-x86_64-2alien.txz
 - gcc-brig-7.1.0_multilib-x86_64-2alien.txz
 - gcc-g++-7.1.0_multilib-x86_64-2alien.txz
 - gcc-gfortran-7.1.0_multilib-x86_64-2alien.txz
 - gcc-gnat-7.1.0_multilib-x86_64-2alien.txz
 - gcc-go-7.1.0_multilib-x86_64-2alien.txz

- gcc-objc-7.1.0_multilib-x86_64-2alien.txz
- Библиотеки GNU libc:
 - glibc-2.25_multilib-x86_64-3alien.txz
 - glibc-i18n-2.25_multilib-x86_64-3alien.txz
 - glibc-profile-2.25_multilib-x86_64-3alien.txz
 - glibc-solibs-2.25_multilib-x86_64-3alien.txz

После обновления до gcc 7 пакета gcc - java больше не существует, так как его разработка прекращена.

Пакет `glibc-zoneinfo` не содержит кода и поэтому не входит в мультилиб. Этот пакет устанавливается из 64-битной Slackware.

Все версии Slackware

Есть один дополнительный пакет для установки программой «installpkg». Актуальная версия для каждого из выпусков Slackware может отличаться, но пакет находится в том же каталоге, что и мультибиблиотечные версии `gcc` и `glibc`:

- «32-битный инструментарий» (сценарий, облегчающий создание 32-битных пакетов)
 - compat32-tools-3.7-noarch-1alien.tgz

Добавление 32-битных библиотек Slackware

Обновление `glibc` и `gcc`, описанное в предыдущем разделе, делает из «готовой к мультибиблиотечности» действительно «мультибиблиотечную» систему.

Теперь всё, что осталось — это установить 32-битные версии системных компонент Slackware, чтобы установленные и/или компилируемые в последующем программы смогли найти все необходимые им 32-битные библиотеки.

Это несколько сложнее, чем взять пакеты из 32-битной Slackware и установить их в Slackware64:

- Сперва нужно разобраться с одинаковыми именами пакетов (два пакета «mesa», два пакета «zlib» и т. д.), которые создадут путаницу для вас и для пакетного менеджера `slackpkg`.
- Кроме того, если 32-битный пакет содержит бинарные файлы в (вроде `/usr/lib/foo`), одноимённые 64-разрядные файлы будут перезаписаны при установке поверх 32-битного пакета и наоборот. Если такое случится, система будет серьёзно попорчена.

Потребуется дополнительные меры предосторожности для удаления ненужных/нежелательных файлов из 32-битных пакетов перед их установкой. Нужны 32-битные пакеты, которые не конфликтуют с тем, что уже есть в 64-битной Slackware. Отсюда и название «32-битный пакет совместимости».

Я решил, что если 32-битные пакеты совместимости для Slackware буду делать я сам, это будет пустой тратой загрузочного канала. В конце концов, вероятно, у вас уже есть купленный DVD Slackware 14.2, то есть уже имеется в наличии и 64-битная, и 32-битная версии Slackware... в

противном случае дерево пакетов 32-битной Slackware, безусловно, доступно для свободной загрузки 😊

Вместо этого я написал несколько сценариев (часть кода написана Fred Emmott для неизвестной [Slamd64](#)) и обернул их в пакет «*compat32-tools*». Их назначение — позволить вам самостоятельно извлечь необходимое из произвольного 32-битного пакета Slackware и использовать его для создания нового пакета, который можно безопасно устанавливать на 64-битной Slackware.

Пакет «*compat32-tools*» требует некоторых пояснений.

Прочтите подробный файл «*README*» в каталоге `/usr/doc/compat32-tools-*/`, он станет хорошим подспорьем. Пакет устанавливает три полезных сценария:

- `/etc/profile.d/32dev.sh`

Это тот же сценарий, что и в *Slamd64*. Он перенастраивает переменные окружения для облегчения компиляции 32-битных программ (путем установки предпочтений 32-битному компилятору и библиотекам перед 64-битными версиями)

- `convertpkg-compat32`

Этот сценарий принимает 32-битный пакет Slackware и собирает на его основе пакет «*compat32*», который можно безопасно установить (при помощи «*installpkg*») на Slackware64, параллельно с 64-битной версией того же самого пакета. Например, допустим, потребовались 32-битные библиотеки из пакета «*mesa*». Берём пакет «*mesa*» из 32-битной Slackware (`x/mesa-7.5-i486-1.txz`) и выполняем

```
# convertpkg-compat32 -i /path/to/mesa-7.5-i486-1.txz
```

который создаст новый пакет `mesa-compat32-7.5-x86_64-1compat32.txz`. Этот новый пакет (созданный в каталоге `/tmp`, если не указали другой каталог) основан на прежнем 32-битном пакете, но не содержит лишнего. Смена базового имени пакета (*mesa* на *mesa-compat32*) позволяет установить этот новый пакет в Slackware64, где он сможет сосуществовать с 64-битным пакетом *mesa* не перезаписывая его файлов.

Временные файлы, оставленные сценарием в каталоге «`/tmp/package-<prgnam>-compat32`», можно удалить.

- `massconvert32.sh`

Этот сценарий содержит внутренний список пакетов, которые я считаю основой 32-битной Slackware. Он использует упомянутый выше сценарий «*convertpkg-compat32*» для конвертации пакетов по внутреннему списку в «*compat32*».

Этот сценарий достаточно запустить один раз, например, как показано ниже (в примере считается, что 32-битный DVD Slackware смонтирован в `/mnt/dvd`):

```
# massconvert32.sh -i /mnt/dvd/slackware -d ~/compat32
```

В результате примерно 150 МБ новых пакетов будут записаны во вновь созданный каталог `~/compat32` (имя каталога выбрано для примера и, конечно, может быть произвольным). Эти пакеты содержат 32-битный костяк мультибиблиотечной системы Slackware64.

Установите их при помощи «*installpkg*», и они обеспечат вам достаточно полный 32-битный слой совместимости поверх Slackware64:

```
# installpkg ~/compat32/*/*.t?z
```

Конечно, при обновлении с предыдущих версий этих пакетов (например, при обновлении 64-битной Slackware до новой версии) нужно использовать «`upgradepkg --install-new`» вместо «`installpkg`»:

```
# upgradepkg --install-new ~/compat32/*/*.t?z
```

Параметр «`-install-new`» нужен, чтобы добавленные в новую версию Slackware новые пакеты *compat32* были установлены.

При установке пакетов *compat32* можно заметить сообщения об отсутствующих файлах в `/etc`. Так задумано, игнорируйте эти ошибки. Эти сообщения вызваны тем фактом, что файлы в `/etc` удаляются из пакетов «`compat32`» при конвертации (за исключением *pango* и *gtk+2*). Я полагаю, что файлы в `/etc` уже установлены оригинальными 64-битными пакетами. Вот пример подобных «ошибок» для пакета `cups-compat32`:

```
Executing install script for cups-compat32-1.3.11-x86_64-1.txz.
install/doinst.sh: line 5: [: too many arguments
cat: etc/cups/interfaces: Is a directory
cat: etc/cups/ppd: Is a directory
cat: etc/cups/ssl: Is a directory
cat: etc/cups/*.new: No such file or directory
cat: etc/dbus-1/system.d/cups.conf.new: No such file or directory
chmod: cannot access `etc/rc.d/rc.cups.new': No such file or directory
cat: etc/rc.d/rc.cups.new: No such file or directory
Package cups-compat32-1.3.11-x86_64-1.txz installed.
```

Если собрались использовать сценарий `convertpkg-compat32` для конвертации в пакет *compat32* пакета **не из Slackware**, настоятельно не рекомендую этого делать. Сценарий написан для единственной цели — сделать 32-битные версии официальных бинарных файлов/библиотек Slackware доступными для мультибиблиотечной установки. Поэтому сценарий оставляет в 32-битном пакете только необходимое для слоя совместимости, полагая, что всё остальное установлено в составе 64-битного пакета.

Если вы загрузили 32-битный пакет не-для-Slackware и хотите использовать его в Slackware64, в подавляющем большинстве случаев наилучшим вариантом будет загрузка исходного кода и сборка 64-битной версии пакета. Либо *устанавливайте оригинальный 32-битный пакет* вместо того, чтобы «конвертировать его» и затем запускать из командной строки для выяснения недостающих 32-битных библиотек, которые, возможно, придётся извлекать из официальных пакетов Slackware.

Запуск 32-битных программ

Когда подготовка системы согласно приведённым выше инструкциям завершена, запуск скомпилированных 32-битных программ несложен. Просто загрузите, установите и запускайте!

Время от времени могут попадаться программы, требующие определённых 32-битных библиотек, которых нет в системе. В этом случае найдите, какой пакет в 32-битной Slackware содержит отсутствующую библиотеку. Используйте сценарий «`convertpkg-compat32`» для

конвертации исходного 32-битного пакета и установите полученный 32-битный «*compat32*» пакет в Slackware64.

Компилирование 32-битных программ

Если нужно скомпилировать 32-битную программу (wine и grub — пара примеров исключительно 32-битных программ с открытым исходным кодом), сперва настройте окружение суперпользователя root запуском следующей команды:

```
# . /etc/profile.d/32dev.sh
```

Обратите внимание на точку в начале строки — это тоже часть команды. Использование точки эквивалентно команде source.

Запуск команды изменяет или создаёт несколько переменных окружения. В результате при компиляции 32-битные версии бинарных файлов получают приоритет над 64-битными — будет запускаться 32-битная компиляция. Изменения действуют до выхода (logout) из шелла суперпользователя root.

В этом изменённом окружении возможно использовать стандартные SlackBuild-ы для сборки 32-битных пакетов для Slackware64. Несколько важных замечаний:

1. Необходимо устанавливать переменную ARCH в значение «i486», поскольку компилируется 32-битная программа хотя бы и на 64-битной системе! Это относится к *тройке* «\$ARCH-slackware-linux», обычно используемой в команде «configure».
2. В качестве исключения, компилировать пакет «wine» необходимо с «ARCH=x86_64», поскольку он будет установлен непосредственно на мультибиблиотечный компьютер без преобразования в пакет «compat32».
3. Для установки этого 32-битного пакета на мультибиблиотечную Slackware64, его нужно преобразовать в совместимый пакет «compat32»:

```
# convertpkg-compat32 -i /path/to/your/fresh/foo-VERSION-i486-BUILD.txz  
# upgradepkg --install-new /tmp/foo-compat32-VERSION-x86_64-BUILDcompat32.txz
```

Предостережения

- После установки пакетов «-compat32» возможно потребуется переустановить бинарные драйверы видеокарт Nvidia или Ati для X.Org. Эти пакеты драйверов содержат одновременно 64-битные и 32-битные библиотеки для максимальной пользы на 64-битной мультибиблиотечной ОС. Если установили файлы драйвера для обеих архитектур, пакет «mesa-compat32» перезапишет некоторые из 32-битных библиотечных файлов.

С другой стороны, если были установлены *только* 64-битные библиотеки драйвера для видеокарты Nvidia/Ati, рекомендуется после установки *мультибиблиотечных* пакетов переустановить пакет бинарного драйвера. На этот раз выберите также и установку 32-битных файлов.

Графическим 32-битным приложениям, запускаемым на мультибиблиотечной системе, потребуются 32-битные библиотеки драйвера. Если не установить нужные файлы, возможны сбои.

- Если соберётесь компилировать собственное 64-битное ядро, удостоверьтесь, что будет скомпилирована возможность 32-битной эмуляции, иначе мультибиблиотечность волшебным образом отключится. Требуемый параметр настройки ядра:

CONFIG_IA32_EMULATION

Пакеты, преобразованные massconvert32.sh

Вот список пакетов, преобразованных в «-compat32» версии сценарием massconvert32.sh. Отметьте, что некоторые из этих пакетов не входят в Slackware 13.0 или 13.1, они были добавлены в более поздних версиях Slackware, и поэтому станут причиной сообщения «* **FAIL: package 'package_name' was not found!**» («* СБОЙ: пакет 'имя_пакета' не найден!») при запуске сценария на ранних версиях Slackware.

```
# Набор A/ :  
  
aaa_elflibs  
attr  
bzip2  
cups  
cxxlibs  
dbus  
e2fsprogs  
eudev  
libgudev  
openssl-solibs  
udev  
util-linux  
xz  
  
# Набор AP/ :  
  
cups  
cups-filters  
flac  
mariadb  
mpg123  
mysql  
sqlite  
  
# Набор D/ :  
  
libtool  
llvm  
opencl-headers  
  
# Набор L/ :
```

SDL2
alsa-lib
alsa-oss
alsa-plugins
atk
audiofile
cairo
dbus-glib
elfutils
esound
expat
ffmpeg
fftw
freetype
fribidi
gamin
gc
gdk-pixbuf2
giflib
glib2
gmp
gnome-keyring
gtk+2
gst-plugins-base
gst-plugins-base0
gst-plugins-good
gst-plugins-good0
gst-plugins-libav
gstreamer
gstreamer0
hal
harfbuzz
icu4c
jasper
json-c
lame
lcms
lcms2
libaio
libart_lgpl
libasyncns
libclc
libedit
libelf
libexif
libffi
libglade
libgphoto2
libidn
libieee1284

```
libjpeg  
libjpeg-turbo  
libmng  
libmpc  
libnl3  
libnotify  
libogg  
libpcap  
libpng  
libsamplerate  
libsndfile  
libtasn1  
libtermcap  
libtiff  
libunistring  
libusb  
libvorbis  
libxml2  
libxslt  
lzo  
ncurses  
ocl-icd  
openjpeg  
orc  
pango  
popt  
pulseaudio  
python-six  
qt  
readline  
sbc  
sdl  
seamoney-solibs  
speexdsp  
startup-notification  
svgalib  
v4l-utils  
zlib
```

Набор N/:

```
curl  
cyrus-sasl  
gnutls  
libgcrypt  
libgpg-error  
libtirpc  
nettle  
openldap-client  
openssl  
p11-kit
```

samba

Набор X/:

fontconfig
freelut
glew
glu
libFS
libICE
libSM
libX11
libXScrnSaver
libXTrap
libXau
libXaw
libXcomposite
libXcursor
libXdamage
libXdmcp
libXevie
libXext
libXfixes
libXfont
libXfont2
libXfontcache
libXft
libXi
libXinerama
libXmu
libXp
libXpm
libXprintUtil
libXrandr
libXrender
libXres
libXt
libXtst
libXv
libXvMC
libXxf86dga
libXxf86misc
libXxf86vm
libdmx
libdrm
libepoxy
libfontenc
libinput
libpciaccess
libva
libva-intel-driver

```
libvdpau
libxcb
libxshmfence
mesa
pixman
vulkan-sdk
xcb-util
```

```
# Набор XAP/:
```

```
sane
```

Зеркала мультибиблиотек

Загрузить мультибиблиотечные пакеты можно (как минимум) по этим адресам:

- <http://slackware.com/~alien/multilib/>
- <http://bear.alienbase.nl/mirrors/people/alien/multilib/>
- <http://slackware.uk/people/alien/multilib/>
- <http://alien.slackbook.org/slackware/multilib/>
- <http://slackbuilds.org/mirror/alien/multilib/>

Сторонние инструменты поддержки

- Sébastien Ballet написал инструмент *compat32pkg*. На [его сайте](#) *compat32pkg* доступен для загрузки как и обширная документация по его использованию в Slackware64. Цитата с сайта:
«*Compat32pkg* — это автоматический инструмент, предоставляющий всё необходимое для управления (конвертации, установки, обновления, удаления) 32-битной части мультибиблиотечной *slackware-64* от AlienBob-a, а также всех 32-битных пакетов из *Slackware-32*, в которых у пользователей 64-битного окружения может возникнуть потребность, таких как *firefox*, *seamonkey*, *jre*, ...»
- Написанный Matteo Rossini (ник *zerouno*) с правками (в частности) от Sébastien Ballet [slackpkg+](#). Это плагин для [slackpkg](#) из Slackware, добавляющий возможность установки пакетов из внешних (сторонних) неофициальных репозиториев Slackware. Он хорошо подходит для добавления мультибиблиотечности в вашу 64-битную Slackware и поддержания её в актуальном состоянии.

Переводы

- Bruno Russo перевёл эту статью на португальский (бразильский):
http://www.brunorusso.eti.br/slackware/doku.php?id=multilib_para_o_slackware_x86_64
- Mehdi Esmaeelpour перевёл статью на персидский:
<http://www.slack-world.com/index.php/articles/43-general-system/85-multilib-slackware64>
- Patrick FONIO и Sebastien BALLETT перевели статью на французский:
<http://wiki.slackware-fr.org/avance:articles:slackware64-multilib>

Благодарности

- Большое спасибо Fred Emmott, создателю Slamd64 — оригинального неофициального 64-битного форка Slackware. Хотя Slackware64 и не основана на его работе, большая часть моих знаний про настройку 32-битной части мультибиблиотечного Linux из его записок в Slamd64.
Заметьте, что в Slamd64 имеются отдельные мультибиблиотечные 64-битные и 32-битные пакеты gcc/glibc. Тем не менее, я считаю, что правильнее сохранять эти основные мультибиблиотечные пакеты неразделёнными. Я последовал концепции, используемой в пакете *binutils* из самой Slackware64, который сочетает 64-битные и 32-битные мультибиблиотечные возможности в едином пакете.
- Cross Linux From Scratch.
Вики CLFS (<http://trac.cross-lfs.org/wiki/read#ReadtheCrossLinuxFromScratchBookOnline>) из разряда «обязательно к прочтению», если хотите разобраться с портированием Linux на новую архитектуру. Там я позаимствовал несколько идей, концепций и патчей при создании Slackware64 с нуля, и опять, когда я создавал с нуля свои мультибиблиотечные пакеты gcc/glibc (мой README по мультибиблиотечности-с-нуля доступен в каталоге `./source`).

Удачи!

Эрик

Источники

- Оригинальная статья Eric Hameleers:
<http://alien.slackbook.org/dokuwiki/doku.php?id=slackware:multilib>
- Перевод на русский: [Serg Bormant](#)

[slackware](#), [multilib](#), [author alienbob](#), [translator bormant](#)

From:
<http://docs.slackware.com/> - **SlackDocs**

Permanent link:
<http://docs.slackware.com/ru:slackware:multilib>

Last update: **2017/07/07 20:40 (BST)**

