Додавання Multilib в Slackware з архітектурою x86_64

Це стаття містить інструкцію по тому, як створити *справжню multilib* Slackware64. Multilib 64-розрядний Linux, спроможний запускати як 64-х так і 32-х розрядні програми. Документ Стандарт ієрархії файлової системи описує метод, який є оптимальним для отримання чіткого розділення між 32-х та 64-х бітним програмним забезпеченням в одній системі. Коли була розпочата розробка порту (адаптованої версії) “Slackware64” (офіційний порт під архітектуру x86_64) ми вирішили адаптувати цей стандарт. Тому Slackware64 була налаштована шукати 64-х розрядні бібліотеки у директоріях `/lib64` та `/usr/lib64`. Ось чому я називаю її Slackware64 “multilib-ready” - незважаючи на те, що 32-х розрядні бібліотеки вона буде шукати у директоріях `/lib` та `/usr/lib`, 32-х розрядних програм чи бібліотек, з системою Slackware64 не постачається. Користувачу потрібно зробити декілька рухів, перед тим як систему можна буде називати Slackware64 “multilib-enabled”.

Це досягається наступним чином:

1. Спочатку нам потрібно перейти на multilib версії:
   - *glibc* (тобто glibc який підтримує *запуск* як 32-х так і 64-х розрядних програм), а також
   - *gcc* (так як він спроможний *компілювати* як 32-х так і 64-х розрядні програми).
2. Після цього треба взяти системні бібліотеки з 32-х розрядної Slackware та встановити в 64-х розрядну, це завершить процес створення 32-х розрядного програмного шару.

Slackware для архітектури x86_64 (чи скорочено “*Slackware64*”) це чиста 64-х розрядна операційна система, але вона дуже просто може бути оновлена до multilib. *З коробки Slackware64 може виконувати та збирати тільки 64-х розрядне програмне забезпечення.*

Коли вийшла Skackware64, вона мала перевагу над 64-розрядними “форками” які були на той час доступні. Ці форки додавали 32-розрядний шар, завдяки перекомпіляції великої кількості своїх пакетів під 32-х розрядну архітектуру. З іншого боку, Slackware, дистрибутив який виходить як 32-х так і в 64-х розрядному вигляді, обидва варіанти розробляються паралельно. Це означає, що вам не доведеться знову збирати 32-х розрядні пакунки з нуля, для того щоб додати сумісність multilib до системи з архітектурою 64. Ви просто берете пакунки з 32-х розрядного дистрибутиву Slackware!
Ось чому ми не додаємо відразу multilib до Slackware64 - натомість ми створюємо передумову, завдяки якій у вас є вибір, та і інструкція по тому як зробити multilib систему.
В розділі нижче, я покажу як взяти пакунки 32-х розрядної Slackware (назвемо їх “mesa”) та переупакуємо їх вміст в пакунки “mesa-compat32”, які можна буде встановити напряму в Slackware64.

Переваги multilib системи

Ось декілька прикладів програмного забезпечення, яке потребує наявності multilib в системі Slackware з архітектурою 64, тому що воно не може бути запущено чи зібрано без сумісного 32-х розрядного шару:

- Wine
  Більшість програм для платформи Windows все ще 32-х розрядні, і для того щоб запустити таке ПЗ на Linux, вам необхідна 32-х розрядна версія Wine.

- VirtualBox
  Популярна програмна, віртуальна машина. Так-як ця програма (частково) з відкритим кодом, для неї досі треба 32-х розрядний шар сумісності в системі Slackware 64-х розрядної архітектури.
- Steam
  Дуже популярна ігрова платформа, досі потребує 32-розрядний клієнт. Більшість доступних ігор, також 32-розрядні.
- Skype, Citrix клієнт, ...
  Ці програми пропріетарні, та їх код закрито. Ми залежимо від розробника, який не виробляє 64-х розрядні бінарні файли. В цьому випадку, саме така ситуація.

На радість, 64-розрядні програми набувають все більшої популярності. Adobe довгий час не випускав, але з якоїсь миті, нарешті випустив Flash плагін для 64-х розрядних операційних систем. Sun (поглинутий компанією Oracle) випустили 64-х розрядну версію плагіна Java. Ці дві великі події подали сигнал, що треба розробляти 64-х розрядну версію - Slackware64.

Отримання мульти-бібліотечних пакунків

Ви можете завантажити пакет multilib-enabled та скрипти з мого веб-сайту: http://slackware.com/~alien/multilib/ .

Крім декількох README файлів (ця стаття з Wiki є поліпшеною версією тих README файлів), ви знайдете по одній під-дерикторії на кожен 64-розрядний випуск Slackware, відносно кореневої папки "*multilib*". Також існує директорія під назвою "source". В папці "*source*" знаходиться сирцевий код пакунків, та скрипти для збірки SlackBuild.
Але дійсно цікаве - бінарні пакунки, які доступні в під-директорії <*slackware_номер_релізу*> яка знаходиться під кореневою директорією. Кожна така директорія містить під-директорію "*slackware64-compat32*" де ви знайдете перезібрані 32-розрядні пакунки, готові для встановлення на 64-х розрядну Slackware.

Підтримання пакунків муьлти-бібліотечності в актуальному стані

Для того, щоб бути в курсі оновлень, Я наполягаю на тому, щоб ви слідкували за журналом змін (RSS новинами) які я обслуговую для моїх мульти-бібліотечних пакунків. За звичай, я *оновлюю пакунки glibc та gcc* наступного дня, після оновлення цих пакунків у Slackware.

Автоматизація:

1. Перевірте інструмент compat32pkg Себастєна Балета, який автоматизує цей процес, на зразок як це робить slackpkg.
2. Якщо ви віддаєте перевагу slackpkg, для керування пакетами, то варто, перевірити slackpkg+, це додаток до slackpkg, який керує пакунками, які ви встановили зі стороннього репозіторія - включаючи мільти-бібліотечність. Коли все правильно налаштовано, то підтримка оновленої версії мільти-бібліотечності дуже просте:

```
# slackpkg update
```

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

Остання команда покаже якщо будь-який новий пакунок було додано до колекції пакунків “compat32”, як наприклад поточні llvm-compat32 та orc-compat32.

- Як за звичай виглядає налаштування - для комп'ютера з запущеною Slackware-current, та використовуючи тестовий репозіторій Alien BOB-а для KDE. PKGS_PRIORITY Забезпечує, що мільти-бібліотечні пакунки для gcc та glibc має перевагу, над оригінальним від Slackware. Ключове слово “multilib” яке означає ім'я репозіторія, має бути таким самим ім'ям, яке використовувалось вище в команді “slackpkg”. Обране ім'я “multilib” довільне, воно могло б, точно так же бути “compat32”, до тих пір, поки ви використовуєте його послідовно.
Зміст файлу для прикладу “/etc/slackpkg/slackpkgplus.conf” буде:

```
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+/
```

Включення підтримки мульти-бібліотечності в Slackware64

Інструкція по швидкому методу

Цей розділ містить основні інструкції щодо додавання повної мульти-бібліотечної сумісності в систему Slackware64. Якщо ви хочете зрозуміти більш детальніше цей процес, чи вам потрібна інформація по збірці 32-х розрядного ПЗ на Slackware64, вам треба ознайомитись з наступними розділами.
Увага, символ “*#*” означає що це *командний рядок користувача root*.

- Завантажте пакунки з мого веб сайту (я дав посилання в попередньому розділі, але в цьому прикладі я використовую дзеркало). Припустимо що у вас запущена Slackware 14.2. Виконайте команди:

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

- Оновіть в 64-х розрядній Slackware пакунки *gcc*“ та ”*glibc*“ до мульти-бібліотечних версій. Виконайте команду:

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

  після того як ви перейшли до директорії в яку завантажили ці пакунки.
  Ця команда встановить допоміжні пакунки під назвою ”*compat32-tools*“.

- Якщо ви також завантажили директорію *slackware64-compat32* (в моєму прикладі команда ”`lftp`“ робить це автоматично) ви будете раді пізнати, що необхідні 32-х розрядні пакунки вже перетворенні як треба! Все що від вас потрібно, це виконати команду:

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

  яка встановить відконвертовані 32-х розрядні пакунки (чи оновить їх, якщо вони вже були встановлені зі старої версії мульти-бібліотечності). Це все!

- Якщо ви не знаходите директорію *slackware64-compat32* це означає, що в її не завантажили, або дзеркало з пакунками не містить необхідну директорію. В цьому випадку, ви можете самі переконвертувати 32-х розрядні пакунки для встановлення. Це не важко, та займе декілька хвилин:
- Найшвидше буде, якщо у вас є локальна копія з оригінальними 32-х розрядними пакунками від Slackware, (в цьому випадку воно має назву *локальне дзеркало*). В тих, у кого є офіційний DVD носій, може використати його: він двосторонній, та 32-х розрядна версія Slackware знаходиться на одній із сторін. Для цього прикладу Я очікую, що у вас є локальна копія 32-х розрядної Slackware у директорії ”`/home/ftp/pub/slackware/slackware-14.1/slackware/`“.
  Де розміщені під-директорії 'a', 'ap', 'd', 'l', 'n' та 'x'. (Якщо у вас змонтовано DVD диск, то директорія може знаходитись тут ”`/media/SlackDVD/slackware/`“ але цей варіант я тут не використовую).
- Створіть нову пусту директорію (давайте назвемо її 'slackware64-compat32') та перейдіть до неї:

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

  - Запустіть наступну команду, для створення низки 32-х розрядних пакетів, використовуючи офіційні 32-х розрядні пакети Slacwkare на вході команди:

    ```
    # massconvert32.sh -i
    /home/ftp/pub/slackware/slackware-14.1/slackware/
    ```

  - Попередній крок, займе деякий час. Коли він завершиться, продовжіть встановленням 90 Мб свіжих перероблених 32-х розрядних пакунків, які були

створені у під-директоріях *відносно поточної*:

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

- Готово! Тепер ви можете завантажувати, встановлювати та запускати 32-х розрядні програми. Це було і не також важко?

Якщо ви використовуєте пакетні менеджери, такі як *slackpkg*, ви повинні додати усі пакети з назвою `glibc` та `gcc` до чорного списку цього менеджеру. Якщо ви цього не зробите, ви ризикуєте тим, що цей менеджер може замінити, мульти-бібліотечні пакунки на оригінальні з Slackware64!
Якщо у вас запущено Slackware 13.37 чи вище, в цих версіях slackpkg, є підтримка регулярних виражень у файлі чорного списку. В цьому випадку, однієї стрічки у файлі `/etc/slackpkg/blacklist` буде достатньо для внесення усіх моїх пакунків до чорного списку (включаючи мульти-бібліотечні пакунки gcc, glibc та усіх `compat32`):

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

Якщо ви працюєте на Slackware 13.1 чи новіше, та ви завантажили пакунок з інструментами compat32-tools для цієї версії Slackware, скрипт *massconvert32.sh* може використовувати віддалений веб сервер, для завантаження 32-х розрядних пакунків Slackware, замість завантаження їх до локального дзеркала чи з DVD. Вкажіть параметр ”-u“ як адресу URL віддаленого сервера:

```
# massconvert32.sh -u http://someserver.org/path/to/slackware-14.1/slackware
```

Детальніша інструкція

Оновлення glibc та gcc

Наступні пакунки glibc/gcc замінять - *не додадуть* - стандартні пакунки Slackware. Використайте команду ”`upgradepkg`“ для оновлення до моїх мільти-бібліотечних версій gcc та glibc. Вони потрібні для запуску (glibc), та збірки (gcc) 32-х розрядного програмного забезпечення на 64-х розрядному комп'ютері зі Slackware:

Slackware64 13.0

- Набір компілятора 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
- Бібліотеки 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

- Набір компілятора 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
- Бібліотеки 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

- Набір компілятора 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
- Бібліотеки 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
    - glibc-zoneinfo-2013d_multilib-noarch-7alien.txz

Slackware64 14.0

- Набір компілятора 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
- Бібліотеки GNU libc:
    - glibc-2.15_multilib-x86_64-8alien.txz
    - glibc-i18n-2.15_multilib-x86_64-8alien.txz
    - glibc-profile-2.15_multilib-x86_64-8alien.txz

    - glibc-solibs-2.15_multilib-x86_64-8alien.txz
    - glibc-zoneinfo-2013d_2013d_multilib-noarch-8alien.txz

Slackware64 14.1

- Набір компілятора gcc:
    - gcc-4.8.2_multilib-x86_64-1alien.txz
    - gcc-g++-4.8.@_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
- Бібліотеки GNU libc:
    - glibc-2.17_multilib-x86_64-7alien.txz
    - glibc-i18n-2.17_multilib-x86_64-7alien.txz
    - glibc-profile-2.17_multilib-x86_64-7alien.txz
    - glibc-solibs-2.17_multilib-x86_64-7alien.txz
    - glibc-zoneinfo-2013d_multilib-noarch-7alien.txz

Slackware64 current

- Доки ви не бачите окрему директорію з назвою "*current*" ви можете використовувати файли в ній, для стабільних версій Slackware.

- Набір компілятора gcc:
    - gcc-4.8.3_multilib-x86_64-1alien.txz
    - gcc-g++-4.8.3_multilib-x86_64-1alien.txz
    - gcc-gfortran-4.8.3_multilib-x86_64-1alien.txz
    - gcc-gnat-4.8.3_multilib-x86_64-1alien.txz
    - gcc-go-4.8.3_multilib-x86_64-1alien.txz
    - gcc-java-4.8.3_multilib-x86_64-1alien.txz
    - gcc-objc-4.8.3_multilib-x86_64-1alien.txz
- Бібліотеки GNU libc:
    - glibc-2.19_multilib-x86_64-1alien.txz
    - glibc-i18n-2.19_multilib-x86_64-1alien.txz
    - glibc-profile-2.19_multilib-x86_64-1alien.txz
    - glibc-solibs-2.19_multilib-x86_64-1alien.txz
    - glibc-zoneinfo-2014b_multilib-noarch-1alien.txz

Ось один додатковий пакунок, який ви встановлюєте програмою "installpkg":

- Це "32-х розрядний набір інструментів" (скрипти які вміють створювати 32-х розрядні пакунки)
    - compat32-tools-3.2-noarch-2alien.tgz

Slamd64 має відокремлені версії 64-х та 32-х розрядні мульти-бібліотечні версії пакунків gcc/glibc.
Але, я вірю, в те, що найкраще не розділяти ці пакунки. Я слідую за принципом з Slakcware64 там свій пакунок *binutils*, який використовує 64-х та 32-розрядний мульти-бібліотечний сумісний

шар в одному пакунку.

Додавання 32-х розрядних бібліотек Slackware

Оновлення glibc та gcc яке було описано вище, перемикає режим системи з "*multilib-ready*" на "*multilib-enabled*".
Все що вам треба, це встановити 32-х розрядні версії програмного забезпечення для Slackware , щоб в майбутньому ці 32-х розрядні програми знаходили 32-х розрядні бібліотеки які їм потрібні для запуску та роботи.

Цей процес не просто, встановлювання 32-х розрядного ПЗ в Slackware64:

- По перше, в цьому випадку, з'явиться пакунки з однаковим ім'ям (два пакунка 'mesa', 'zlib' та інші...) це буде вводити в оману користувача, та пакетні менеджери як наприклад *slackpkg*.
- А ще, якщо 32-х розрядний пакунок містить бінарні файли (наприклад `/usr/bin/foo`), вони перезапишуть 64-х розрядні оригінали, коли ви будете встановлювати 32-х розрядний пакунок поверх існуючого 64-х розрядного. Це буде дуже фатально, якщо таке трапиться.

Тут треба бути уважнішим, щоб не трапилась плутанина з файлами та їх розрядністю під час встановлення 32-х розрядних пакунків. Але вам треба 32-х розрядний пакунок, який не конфліктує з вже встановленим 64-х розрядним в системі. Його назва "32-х розрядний сумісний пакунок".

Я вирішив, буде марно витрачено трафік, якщо я створю сумісні 32-х розрядні пакунки для Slackware. Та і велика вірогідність що ви придбали диск з Slackware 14.1 DVD, і у вас вже є обидві 64-х та 32-х розрядні версії цієї системи... чи є доступ до дерева пакунків Slackware яке доступно для безкоштовного завантаження 

Напроти, я написав скрипт (частина нього написана Фредом Емотом для Slamd64) та включено до пакунка "*compat32-tools*". Цей скрипт запропонує, розпакувати вміст 32-х розрядного пакунку, та створення з нього нового пакунка, який можна буде встановлювати на 64-х розрядну Slackware.

Пакунок "*compat32-tools*" вимагає деякого розуміння.
Тому протчіть файл '*README*' в директорії `/usr/doc/compat32-tools-*/`, це допоможе розібратись що до чого. В ньому є низька корисних скриптів, які встановлюються в систему:

- */etc/profile.d/32dev.sh*
  Це той ж самий скрипт який йде з Slamd64. Він переналаштовує оточення оболонки, для того щоб було зручно збирати 32-х розрядне програмне забезпечення (перемикаючи пріоритет на 32-х розрядний компілятор та бібліотеки в 64-х розрядній Slacwkare)
- *convertpkg-compat32*
  Цей скрипт бере 32-х розрядний пакунок та конвертує його в пакунок '-compat32', який потім можна встановити (використовуючи "installpkg") в Slackware64, біля 64-х розрядної версії того ж самого пакунка. Наприклад. припустимо що вам потрібні 32-х розрядні бібліотеки які містяться у пакунку mesa. Ви берете пакунок mesa з 32-х розрядної Slackware (`x/mesa-7.5-i486-1.txz`) та виконуєте команду

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

яка створить новий пакунок під назвою `mesa-compat32-7.5-x86_64-1.txz`. Новий пакунок (який було створено у директорії '/tmp' у випадку якщо ви не вказали іншу директорію для зберігання) це старий 32-х розрядний пакунок, але з нього вилучено не потрібні частини. Змінено ім'я (*mesa* стало *mesa-compat32*) що дозволяє встановити цей пакунок в Slackware64 де вже є 64-х розрядна *mesa* не перезаписав вже встановлений пакунок.
Скрипт залишає тимчасові файли у директорії "`/tmp/package-<prgnam>-compat32`" які можна безпечно видалити.

- *massconvert32.sh*
  В цьому скрипті знаходиться список 32-х розрядних пакунків, які я вважаю необхідні. Він використовує скрипт "*convertpkg-compat32*" для стягування кожного пакунка із з списку, та їх конвертація в пакунки '-compat32'.
  Цей скрипт треба запускати лише один раз, на приклад (в цьому прикладі вважається, що 32-х розрядна Slackware на DVD змонтована в `/mnt/dvd`):

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

  Після цього, у вас з'явиться пакунки близько 60Мб в новій директорії `~/compat32` (ім'я цієї директорії можна налаштовувати, я вибрав таке ім'я лише для прикладу). Ці пакунки включають 32-х розрядні компоненти для мульти-бібліотечної Slackware64.
  Їх треба встновити використовуючи "`installpkg`", зо надасть шар сумісності, поверх Slackware64:

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

  Якщо ви виконуєте оновлення з попередньої версії цих пакунків (на приклад ви оновлюєте 64-х розрядну Slackware до нової версії) це означає, що використовувати "`installpkg`" не можна, а замість цієї команди, треба виконати "`upgradepkg –install-new`": code>

upgradepkg –install-new ~/compat32/*/*.t?z </code> параметр "*–install-new*" необхідний для, того щоб додати нові пакунки *compat32* які були додані між версіями.

Коли встановлюються пакунки *compat32*, можна побачити помилки про відсутні файли в `/etc`. Це "так заплановано", ці помилки можна проігнорувати. Ці повідомлення, говорять нам про той факт, що файли в `/etc` були видалені з пакунків "-compat32" в процесі конвертації (окрім *pango* та *gtk+2*). Мається на увазі, що файли в `/etc`, вже встановлені від оригінальних 64-х розрядних пакунків.
Приклад цієї "помилки" в пакунку `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.
```

Якщо ви плануєте використати скрипт `convertpkg-compat32` для конвертації **не-Slackware** пакунка в *-compat32*, Я повинен вас попередити. Цей інструмент було створено лише для однієї задачі, це створення з 32-х розрядних пакунків в набір мільти-бібліотечності для Slackware64. Та такі дії як, вилучення деяких компонентів з оригінальних 32-х розрядних пакунків - компоненти які були встановлені як частина оригінальних 64-х розрядних пакунків.
В багатьох випадках, коли ви завантажуєте не-Slackware 32-х розрядний пакунок, та хочете щоб він запрацював на Slackware64, найліпше буде знайти сирцевий код цієї програми, та зібрати 64-х розрядну версію цього пакунка. Альтернативно, ви можете просто *встановити оригінальній* 32-х розрядний пакунок, замість “його конвертації”, та спробувати його запустити з командного рядка, щоб побачити помилки про відсутні 32х розрядні бібліотеки, які потрібно дістати з офіційних пакунків Slackware.

Запуск 32-х розрядних програм

Запуск вже зібраної 32-х розрядної програми, стане можливим, як тільки ви закінчите підготовку системи як показано вище. Залишиться тільки завантажити, встановити та запустити її!

Інколи, буде траплятись, що ви запустите програму яка потребує конкретні 32-х розрядні бібліотеки, яких у вас немає. В цьому випадку, знайдіть ці бібліотеки в 32-х розрядних пакунках Slackware. Використайте скрипт ”*convertpkg-compat32*“ для конвертації знайденого 32-х розрядного пакунку з необхідними бібліотеками у ”*сумісний*“ пакунок який можна буде встановити в Slackware64.

Компіляція 32-х розрядних програм

В тому випадку, якщо вам необхідно скомпілювати 32-х розрядне програмне забезпечення (wine та grub це яскраві приклади програм, які існують тільки в 32-х розрядному вигляді) знадобиться налаштувати оточення оболонки користувача root, запустивши команду:

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

Звернуть увагу на “крапку” в початку - це актуальна частина команди! Використання крапки еквівалентно використання команди 'source'.
Виконання цієї команди змінює декілька змінних оточення. Ефект від цього, це зміна пріоритету, 32-х розрядних версій бінарних файлів над 64-х розрядними, на час компіляції 32-х розрядного ПЗ з сирцевого коду. Ефект буде працювати доки ви не вийдете з оболонки користувача root.

В цій модифікованої оболонці, змінних оточення, ви можете використовувати стандартні SlackBuild для створення 32-х розрядних пакунків для Slackware64. Є декілька речей які треба пам’ятати:

1. Ви повинні встановити значення змінної ARCH в 'i486', тому що, навіть на комп’ютері з архітектурою 'x86_64', ви збираєте 32-х розрядне ПЗ!
   Це пов’язано з *triplet* змінною “$ARCH-slackware-linux” який за звичай використовується в

команді “configure”.
2. Як виняток, ви повинні компілювати пакунок “wine” з змінною 'ARCH=x86_64', так як цей пакунок буде встановлюватись напряму в мульти-бібліотечну систему, без переконвертації в сумісний 'compat32' пакунок.
3. Якщо ви хочете встановити цей 32-х розрядний пакунок в систему Slackware64-multilib, ви повинні спочатку конвертувати його в сумісний пакунок 'compat32':

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

Застереження

- Після встановлення пакунків ”`-compat32`“, ви повинні пере-встановити бінарні драйвери X.Org для *Nvidia* чи *Ati*. Ці пакунки містять обидва варіанти 32-х та 64-х розрядних бібліотек, для максимальної користі на 64-х розрядній ОС. Якщо ви встановили драйвер з файлами для декількох архітектур, пакунок ”`mesa-compat32`“ перезапише деякі 32-х розрядні бібліотеки.

  Іншою мовою, якщо ви встановили 64-х розрядний драйвер з бібліотеками для вашої відеокарти Nvidia/Ati, рекомендовано після встановлення мульти-бібліотечних пакунків, пере-встановити бінарні драйвери. В цей раз це також 32-х розрядний драйвер.

  Графічні 32-х розрядні програми, запущені на мульти-бібліотечній системі, потребують 32-х розрядні графічні драйвери та бібліотеки. Збої такого ПЗ, свідчать про некоректне встановлення необхідних файлів.
- Якщо вам знадобиться скомпілювати своє 64-х розрядне ядро, перевірте що збираєте ядро з можливістю 32-х розрядної емуляції, бо мульти-бібліотечність таємничо не буде працювати. Для ядра необхідні деякі параметри: **CONFIG_IA32_EMULATION**

Пакунки пере-конвертовані скриптом massconvert32.sh

Це список пакунків, які пере-конвертовано в сумісні пакунки ”*-compat32*“ скриптом `massconvert32.sh`. Увага, деякі з цих пакунків, не є частиною Slackware 13.0 чи 13.1, вони біли добавлені, з більш нових версій Slackware, вони можуть спричинити деякі сповіщення про помилки ”**** FAIL: package 'package_name' was not found!****“ під час запуску цього скрипту на старих версіях Slackware. Навпаки вірно - деякі пакунки були *вилучені* в нових версіях Slackware які спричиняють сповіщення ”**** FAIL: package 'package_name' was not found!****“. Не піклуйтесь про це.

```
# Серія A/:

aaa_elflibs
attr
bzip2
cups
cxxlibs
```

```
dbus
e2fsprogs
openssl-solibs
udev
util-linux

# Серія AP/:

flac
mariadb
mpg123
mysql
sqlite

# Серія D/:

libtool

# Серія L/:

alsa-lib
alsa-oss
atk
audiofile
cairo
dbus-glib
esound
expat
freetype
fribidi
gamin
gdk-pixbuf2
giflib
glib2
gmp
gnome-keyring
gtk+2
gst-plugins-base
gst-plugins-good
gstreamer
hal
harfbuzz
icu4c
jasper
lcms
lcms2
libart_lgpl
libelf
libexif
libffi
libglade
```

```
libgphoto2
libidn
libieee1284
libjpeg
libmng
libmpc
libogg
libpcap
libpng
libsamplerate
libsndfile
libtasn1
libtermcap
libtiff
libusb
libvorbis
libxml2
libxslt
ncurses
pango
popt
qt
readline
sdl
seamonkey-solibs
startup-notification
svgalib
v4l-utils
zlib

# Серія N/:

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

# Серія X/:

fontconfig
freeglut
glew
glu
libFS
libICE
libSM
```

```
libX11
libXScrnSaver
libXTrap
libXau
libXaw
libXcomposite
libXcursor
libXdamage
libXdmcp
libXevie
libXext
libXfixes
libXfont
libXfontcache
libXft
libXi
libXinerama
libXmu
libXp
libXpm
libXprintUtil
libXrandr
libXrender
libXres
libXt
libXtst
libXv
libXvMC
libXxf86dga
libXxf86misc
libXxf86vm
libdmx
libdrm
libfontenc
libpciaccess
libxcb
mesa
pixman
xcb-util

# Серія XAP/:

sane
xsane
```

Дзеркала для завантаження пакунків мульти-бібліотечності

Ви можете завантажити пакунки мульти-бібліотечності з цих місць:

- http://slackware.com/~alien/multilib/
- http://taper.alienbase.nl/mirrors/people/alien/multilib/
- http://slackware.org.uk/people/alien/multilib/
- http://alien.slackbook.org/slackware/multilib/

Сторонні інструменти

- Себастієн Балет написав інструмент під назвою *compat32pkg*. На цьому сайті *compat32pkg* доступний для завантаження, також там є документація по використанню цього інструменту в Slackware64.
  Процитую цей сайт:
  “Compat32pkg це інструмент автоматизації, який надає усе необхідне для керування (конвертація, встановлення, оновлення та видалення) 32-х розрядною частиною мульти-бібліотечності AlienBob'а для slackware-64, та усі 32-х розрядні пакунки з Slackware-32 які можуть знадобитись для 64-х розрядного оточення, як наприклад firefox, seamonkey, jre,…”
- Також існує slackpkg+, написаний Матео Розіні (прізвисько zerouno) з вкладником (зокрема) Себастієн Балет. Це плагін для slackpkg який додає можливість встановлювати додаткові пакунки з зовнішніх не офіційних Slackware репозіторіїв. Дуже зручна підтримка мульти-бібліотечних пакунків, дозволяє відслідковувати новини, та використовувати завжди найсвіжішу версію.

Переклади

- Бруно Руссо переклав цю статтю на португальську (бразильська): http://www.brunorusso.eti.br/slackware/doku.php?id=multilib_para_o_slackware_x86_64
- Мехді Есмаелпур переклав на перську: http://www.slack-world.com/index.php/articles/43-general-system/85-multilib-slackware64
- Патрік Фоніо та Себастієн Балет переклали статтю на французьку: http://wiki.slackware-fr.org/avance:articles:slackware64-multilib

Подяки

- Велика подяка Фред Емоту, він створив Slamd64, оригінальній не офіційний порт Slackware для 64-х розрядної архітектури. Хоча Slackware64 не базується на цьому форку. Я досі пізнаю, багато чого про налаштування 32-х розрядної частини мільти-бібліочечності в Linux, яке було описано для Slamd64.
- Linux From Scratch.
  CLFS вікі (http://trac.cross-lfs.org/wiki/read#ReadtheCrossLinuxFromScratchBookOnline) яку 'необхідно-прочитати' якщо ви хочете зрозуміти як портувати Linux на інші архітектури. Я взяв з відси деякі ідеї, концепти та патчі для створення Slackware64 з нуля, та знову коли створював пакунки мульти-бібліотечності gcc/glibc (файл README можна знайти в директорії ./source)

Удачі!

Ерік

Джерела

- Оригінальну статтю написав Ерік Хамелірс, на http://alien.slackbook.org/dokuwiki/doku.php?id=slackware:multilib

slackware, multilib, author alienbob