

Лабораторная работа №10.

Управление ПО в Astra Linux

Введение

Сегодня мы посмотрим на приложения одновременно с двух позиций: как программисты и как системные администраторы. Из этой лабораторной вы узнаете, что такое deb-пакеты, зачем они нужны, из чего состоят, как работают пакетные менеджеры и зачем их так много.

Исполняемые файлы ELF

Для того чтобы эффективно управлять программным обеспечением, нужно понимать, как оно работает и из чего состоит.

В предыдущей лабораторной мы уже познакомились с процессами и знаем, что операционная система может выполнять инструкции из специальных бинарных файлов, которые называются исполняемыми файлами. В Windows такие файлы имеют расширение *.exe (от англ. executable files - исполняемые файлы), а в Linux это ELF (от англ. executable and linkable format — формат исполняемых и связываемых файлов).

С помощью утилиты `hexdump` мы можем легко заглянуть в начало этих файлов и найти специфичные для них сигнатуры, но если с ELF все понятно, то MZ в exe-файлах – это не что иное, как инициалы ведущего разработчика MS-DOS Марка Збиковски.

```
sa@astra:~$ hexdump -C /usr/bin/cat | head -3
00000000 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 |.ELF.....|
00000010 02 00 3e 00 01 00 00 00 50 2f 40 00 00 00 00 |..>.....P/@....|
00000020 40 00 00 00 00 00 00 00 90 95 00 00 00 00 00 |@.....|

sa@astra:~$ file /usr/bin/cat
/usr/bin/cat: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynami...
```

В качестве подопытного возьмем WinBox от MikroTik:

```
sa@astra:~$ wget https://download.mikrotik.com/routeros/winbox/3.41/winbox64.exe
sa@astra:~$ hexdump -C winbox64.exe | head -3
00000000 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 |MZ.....|
00000010 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 |.....@.....|
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|

sa@astra:~$ file winbox64.exe
winbox64.exe: PE32+ executable (GUI) x86-64 (stripped to external PDB), for MS Windows
```

Обратите внимание, что утилита `file` без проблем распознает сигнатуру exe-файлов даже без расширения.

Файлы EXE и ELF, конечно, отличаются по формату, но куда важнее то, что их программный код по-разному взаимодействует с операционной системой для получения доступа к ресурсам. В Windows это взаимодействие осуществляется через API, а в Unix-подобных системах через прямые системные вызовы к ядру, поэтому просто так вы не запустите exe-файл на Linux и ELF-файл на Windows.

Для возможности использования одного и того же приложения на разных операционных системах разработчики выполняют его портирование, т.е. вносят необходимые изменения для возможности компиляции приложения под другую целевую операционную систему. Существуют также языки программирования, библиотеки и фреймворки, которые позволяют изначально писать полностью кроссплатформенный код.

Примечание

Можно отметить еще, что исполняемые файлы компилируются под разные архитектуры процессоров, отличающиеся набором инструкций на аппаратном уровне, поэтому у одной и той же версии приложения могут быть релизы для архитектуры x86-64 (она же AMD64, Intel64, /EM64T), ARM, Эльбрус и др. Компиляторы современных языков высокого уровня позволяют генерировать исполняемый код под разные архитектуры даже без внесения каких-либо изменений в программный код, поэтому этот нюанс не стоит большого внимания.

Для еще большего стирания границы между Windows и Linux был создан продукт Wine, который умеет перехватывать вызовы exe-приложений к библиотекам операционной системы Windows и подменять их своими, обращаясь уже к системным вызовам системы Linux. Но следует понимать, что этот проект сталкивается с большими трудностями вследствие отсутствия документации по многим элементам Win32 API. Однако, если у вас на предприятии используются какие-то древние клиентские приложения, на переписывание которых нет ресурсов, то Wine вполне может стать для вас спасательным кругом, и такую возможность нужно обязательно проверить.

Примечание

Для установки Wine выполните команду `sudo apt install wine ia32-libs`, после чего приложение WinBox, например, можно будет запустить командой `wine winbox64.exe` или двойным кликом из файлового менеджера.

А что насчет совместимости систем, основанных на Linux? Ядра этих систем используют общую кодовую базу и компилируются с небольшими отличиями, поэтому один и тот же ELF-файл может вполне успешно запускаться и на RHEL, и на Debian, и на Astra Linux. Собственно, именно поэтому мы можем запускать Linux-контейнеры разных систем на одном и том же ядре хостовой системы и portable файлы AppImage, которым для работы достаточно указать флаг запуска +x.

Пакеты приложений и менеджер dpkg

При создании сложных приложений разработчики редко могут обойтись одним исполняемым файлом, т.к. некоторые функции приложения бывает целесообразно вынести в отдельные утилиты, а общий для них программный код оформить в виде динамических библиотек. И не забываем еще про файлы настроек, файлы справочной системы и т.д.

Если же мы вспомним еще о том, что Linux придерживаются концепции единой иерархии файловой системы FHS, то окажется, что все файлы приложения при установке нужно еще и разложить по правильным папкам. Учитывая все вышесказанное, в целях упрощения установки приложений они распространяются в виде архивов, которые называют пакетами программ, а их установку в системе поручают пакетным менеджерам.

Разные операционные системы используют разные пакеты и пакетные менеджеры, например, в Debian это deb-пакеты и Debian package management system (DPKG), в RedHat — RedHat Package Manager (RPM), в Arch Linux — PAMCAN, в OpenSUSE — Zypper. По тому, какие пакеты используются в системе, разные дистрибутивы Linux иногда даже объединяют в группы deb-based, rpm-based и другие.

Есть свой пакетный менеджер и в Windows, который называется [Windows Package Manager](#) (winget). Он работает с MSI-пакетами и EXE программами установки. Winget имеет свой репозиторий ПО, из которого он устанавливает пакеты на целевую систему, но можно также добавить сторонние репозитории или создать свой.

Примечание

Причина, по которой мы имеем дело с таким большим разнообразием пакетных менеджеров, очень проста: управление пакетами является крайне сложной задачей, к решению которой подступались уже неоднократно, но ни один из созданных инструментов нельзя считать идеальным, поэтому будут пробовать еще раз и еще.

Система управления пакетами DPKG предоставляет набор динамических библиотек, реализующих программные интерфейсы, и одноименную утилиту командной строки dpkg, которая, используя эти интерфейсы, обеспечивает администраторам доступ ко всем возможностям системы. Приведем наиболее часто используемые команды:

- `sudo dpkg -i <пакет>.deb` — ключ `-i` (`--install`) означает, что нужно установить указанный пакет. Указывается имя deb-файла, включая абсолютный или относительный путь к нему.

- `sudo dpkg -r <имя пакета>` — ключ `-r` (`--remove`) указывает, что требуется удалить пакет, сохраняя конфигурационные файлы в системе. Указывается имя пакета, установленного ранее в системе.
- `sudo dpkg -P <имя пакета>` — ключ `-P` (`--purge`) указывает, что нужно удалить пакет полностью вместе с конфигурационными файлами.
- `dpkg -l` — ключ `-l` (строчная L, `--list`) позволяет получить список всех установленных в системе пакетов. Для получения этой информации привилегии суперпользователя не требуются.
- `dpkg -I <пакет>.deb` — ключ `-I` (заглавная i, `--info`) позволяет получить подробную информацию о пакете из deb-файла, включая краткое описание и список зависимостей.
- `dpkg -s <имя пакета>` — ключ `-s` (`--status`) позволяет получить подробную информацию о пакете, ранее установленном в системе, включая краткое описание.
- `dpkg -L <имя пакета>` — ключ `-L` (`--listfiles`) позволяет получить полный список всех файлов, которые были установлены в системе из этого пакета, включая справочные страницы.
- `dpkg -S <имя файла>` — ключ `-S` (`--search`) позволяет найти пакет, которому принадлежит указанный файл. Для поиска вы можете указать полный путь к файлу, но учитывайте, что это может быть ссылка, которая не зарегистрирована в dpkg. Поэтому, если вы получите ошибку «не найден путь, подходящий под шаблон», попробуйте указать только имя файла и проанализировать все полученные результаты.

Рассмотрим несколько практических примеров:

С помощью ключа `-l` (строчная L, `--list`) получим список установленных пакетов и отфильтруем его:

```
sa@astra:~$ dpkg -l | grep sudo
ii  sudo  1.8.27-1+deb10u5+ci202309191157+astra1  amd64  Provide ...
```

Запросим детализированную информацию об установленном пакете:

```
sa@astra:~$ dpkg -s sudo
Package: sudo
Status: install ok installed
Priority: optional
Section: admin
Installed-Size: 3792
Maintainer: Bdale Garbee <bdale@gag.com>
Architecture: amd64
Version: 1.8.27-1+deb10u5+ci202303222055+astra1
Replaces: sudo-ldap
Depends: libaudit1 (>= 1:2.2.1), libc6 (>= 2.27), libpam0g (>= 0.99.7.1), libselinux1 (>= 1.32),
libpam-modules, lsb-
base
Conflicts: sudo-ldap
Conffiles:
/etc/init.d/sudo 1153f6e6fa7c0e2166779df6ad43f1a8
/etc/pam.d/sudo 4af0e4d91e635bebb96ca0eabdea1182
/etc/sudoers 45437b4e86fba2ab890ac81db2ec3606
/etc/sudoers.d/README 8d3cf36d1713f40a0ddc38e1b21a51b6
```

Description: Provide limited super user privileges to specific users
Sudo is a program designed to allow a sysadmin to give limited root privileges to users and log root activity. The basic philosophy is to give as few privileges as possible but still allow people to get their work done. This version is built with minimal shared library dependencies, use the sudo-ldap package instead if you need LDAP support for sudoers.
Homepage: <http://www.sudo.ws/>

Отообразим список всех файлов, установленных из пакета acl:

```
sa@astra:~$ dpkg -L acl
./
/bin
/bin/chacl
/bin/getfacl
/bin/setfacl
. . .
/usr/share/man/man5
/usr/share/man/man5/acl.5.gz
```

Скачаем deb файл Яндекс диска командой `wget`:

```
sa@astra:~$ wget https://repo.yandex.ru/yandex-disk/yandex-disk_latest_amd64.deb
--2025-03-28 12:25:02-- https://repo.yandex.ru/yandex-disk/yandex-disk_latest_amd64.deb
Распознаётся repo.yandex.ru (repo.yandex.ru)... 213.180.204.183, 2a02:6b8::183
Подключение к repo.yandex.ru (repo.yandex.ru)[213.180.204.183]:443... соединение установлено.
HTTP-запрос отправлен. Ожидание ответа... 200 OK
Длина: 4796514 (4,6M) [application/octet-stream]
Сохранение в: «yandex-disk_latest_amd64.deb»
yandex-disk_latest_amd 100%[=====] 4,57M 10,7MB/s
за 0,4с
2025-03-28 12:25:02 (10,7 MB/s) - «yandex-disk_latest_amd64.deb» сохранён [4796514/4796514]
```

Выведем информацию об этом deb-пакете:

```
sa@astra:~$ dpkg -I yandex-disk_latest_amd64.deb
new Debian package, version 2.0.
size 4796514 bytes: control archive=1740 bytes.
   51 байт(a),      1 строка      conffiles
  540 байт(a),     15 строка      control
   729 байт(a),     10 строка      md5sums
 1663 байт(a),     66 строка      * postinst      #!/bin/sh
   371 байт(a),     24 строка      * postrm        #!/bin/sh
    41 байт(a),      3 строка      * prerm         #!/bin/sh
Package: yandex-disk
Version: 0.1.6.1080
Maintainer: Yandex LLC <support@disk.yandex.ru>
Architecture: amd64
Installed-Size: 12404
Depends: libc6, libgcc1, libstdc++6, zlib1g
Section: misc
Priority: optional
Description: Yandex.Disk keeps your files with you at all times.
 You can access photos, videos, and documents on Disk
 from any where in the world where there's internet.
 The Yandex.Disk client console lets you:
 * synchronize files and folders with your Disk
 * get public links to files and folders
 * customize folder syncing
```

Выполним установку Яндекс диска из deb-пакета утилитой `dpkg`:

```
sa@astra:~$ sudo dpkg -i yandex-disk_latest_amd64.deb
Выбор ранее не выбранного пакета yandex-disk.
```

(Чтение базы данных ... на данный момент установлено 191708 файлов и каталогов.)

```
Подготовка к распаковке yandex-disk_latest_amd64.deb ...
Распаковывается yandex-disk (0.1.6.1080) ...
Настраивается пакет yandex-disk (0.1.6.1080) ...
Using /usr/bin/wget
Обрабатываются триггеры для man-db (2.8.5-2) ...
Обрабатываются триггеры для xserver-xorg-core (2:1.20.14-1ubuntu1astra.se31) ...
update exec ids due to /usr/bin changed
```

Найдем, какому пакету принадлежит утилита `zcat`:

```
sa@astra:~$ dpkg -S zcat
gzip: /usr/share/man/man1/zcat.1.gz
xz-utils: /usr/share/man/man1/xzcat.1.gz
bzip2: /usr/share/man/man1/bzcat.1.gz
bzip2: /bin/bzcat
klibc-utils: /usr/lib/klibc/bin/zcat
gzip: /bin/zcat
xz-utils: /usr/bin/xzcat
```

Не забываем, что `/bin` является символической ссылкой на `/usr/bin`:

```
sa@astra:~$ ls -l /bin
lrwxrwxrwx 1 root root 7 map 22 2023 /bin -> usr/bin
```

При установке и удалении приложений пакетный менеджер `dpkg` обновляет информацию в своей локальной базе, которая находится в папке `/var/lib/dpkg/` и представляет собой обычные текстовые файлы. Например, когда мы запускаем команду `dpkg -S zcat`, она выполняет поиск всех файлов `*.list` в каталоге `/var/lib/dpkg/info/`, которые содержат имя `zcat`, что можно имитировать с помощью обычной команды `find`:

```
sa@astra:~$ find /var/lib/dpkg/info/ -name '*.list' -exec grep -iH 'zcat' {} \;
/var/lib/dpkg/info/klibc-utils.list:/usr/lib/klibc/bin/zcat
/var/lib/dpkg/info/xz-utils.list:/usr/bin/xzcat
/var/lib/dpkg/info/xz-utils.list:/usr/share/man/man1/xzcat.1.gz
/var/lib/dpkg/info/bzip2.list:/bin/bzcat
/var/lib/dpkg/info/bzip2.list:/usr/share/man/man1/bzcat.1.gz
/var/lib/dpkg/info/gzip.list:/bin/zcat
/var/lib/dpkg/info/gzip.list:/usr/share/man/man1/zcat.1.gz
```

Управление зависимостями и приложения apt, aptitude, synaptic

Разработчики приложений могут писать код с нуля, а могут воспользоваться уже готовыми библиотеками, утилитами или даже целыми приложениями, что может существенно повысить скорость разработки и качество программного кода.

Методология разработки приложений с использованием частей других программных продуктов получила название «повторного использования кода» (англ. *code reuse*). Эта идея лежит в основе абсолютно всех языков программирования высокого уровня, платформ для

разработки СУБД и всевозможных фреймворков – все они облегчают разработку за счет того, что предлагают повторное использование их кода.

Например, при разработке Системы управления ресурсами (СУР, устаревшее название ERP) мы можем написать собственную систему хранения данных на языке Си, воспользоваться библиотекой для создания встроенных баз данных Berkeley DB, системой управления базами данных PostgreSQL или даже технологической платформой для разработки систем учета «1С:Предприятие». И чем более подходящий инструмент будет выбран, тем успешнее окажется продукт в будущем.

В системе Windows администраторы не так часто сталкиваются с ситуациями, когда для работы одного приложения нужно установить что-то еще. Ну, максимум потребуется обновить библиотеки VC++ Runtime и поставить .net framework, виртуальную машину Java (JVM) или кодеки. А в Linux идея повторного использования кода была возведена в абсолют, что стало одновременно ее благословением и проклятьем.

Для работы отдельных Linux-приложений требуется установить десятки или даже сотни других пакетов, и когда-то давно это приходилось делать исключительно вручную, на что уходило катастрофически много времени. Поэтому для упрощения процедуры управления пакетами были созданы продвинутые пакетные менеджеры, такие как APT (Advanced Packaging Tool), которые умеют автоматически разрешать зависимости и загружать все необходимые пакеты из сетевых репозиториях. Первым оказался Debian со своим APT, что обеспечило ему технологическое лидерство на несколько лет, а затем подтянулся и RedHat со своим Yum и DNF.

Например, если скачать с сайта Яндекса пакет `yandex-browser-stable_24.4.1.951-1_amd64.deb`, то при попытке установить его обычной командой `dpkg -i` вы получите ошибку «проблемы зависимостей». Но стоит в `/etc/apt/source.list` раскомментировать репозитории Astra Linux и выполнить команду `apt update`, как установка Яндекс браузера и множества других прикладных приложений станет доступна всего одной командой типа `apt install yandex-browser-stable`. Причем загружаться эти пакеты будут из проверенных репозиториях вендора, что намного безопаснее.

Проверьте, что в файле источников у вас раскомментированы репозитории **base** и **extended**

```
sa@astra:~$ cat /etc/apt/sources.list
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-base/ 1.7_x86-64 main
contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended/ 1.7_x86-64
main contrib non-free
...
```


Обновите индекс доступных пакетов:

```
sa@astra:~$ sudo apt-get update
Пол:1 http://repo.yandex.ru/yandex-disk/deb stable InRelease [3 900 B]
Игн:2 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main 1.7_x86-64
InRelease
...
Получено 7 994 kB за 10с (829 kB/s)
Чтение списков пакетов... Готово
...
```

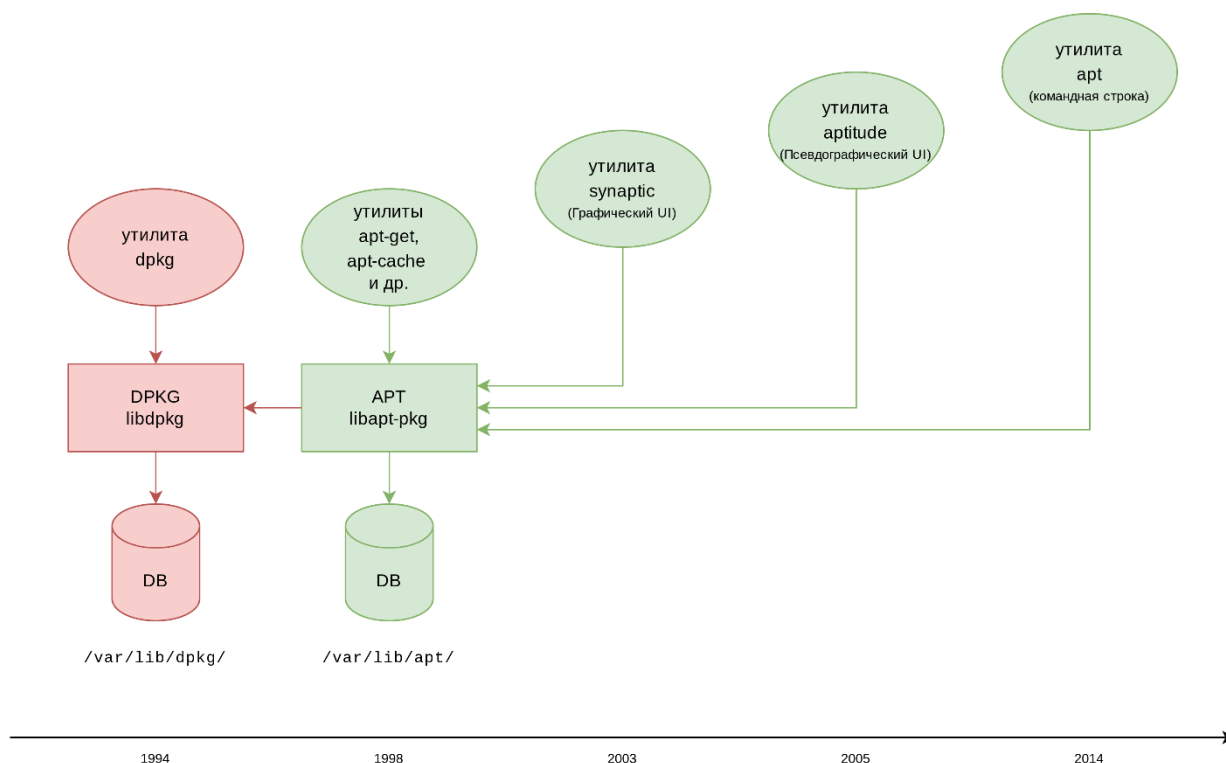
Установите Яндекс браузер из репозитория Astralinux:

```
sa@astra:~$ sudo apt-get install yandex-browser-stable
[sudo] пароль для sa:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
...
Следующие НОВЫЕ пакеты будут установлены:
  ffmpeg-plugin-browser gstreamer1.0-libav gstreamer1.0-plugins-bad ... mesa-vulkan-drivers
yandex-browser-stable
Обновлено 0 пакетов, установлено 40 новых пакетов ...
Необходимо скачать 190 MB архивов.
После данной операции объём занятого дискового
пространства возрастёт на 448 MB.
Хотите продолжить? [Д/н]
...
```

Следует понимать, что DPKG и APT — это технологии, доступ к которым осуществляется через API динамических библиотек, а утилиты dpkg, apt-get и др. — это всего лишь инструменты командной строки, которые к ним обращаются. Утилиты apt-* вышли сразу вместе с APT в 1998 году. Чуть позднее в 2003 году появилась графическая оснастка Synaptic, а затем в 2005 году увидела свет утилита с псевдографическим интерфейсом Aptitude. Но значительным результатом переосмысления опыта работы с зависимостями стала утилита командной строки apt, первый релиз которой состоялся в 2014 году.

Утилита apt предлагает комплексное разрешение зависимостей и разработана как интерфейс конечного пользователя, поэтому выводит более подробные комментарии о результатах выполнения каждой команды `apt` и использует красивую подсветку вывода. В скриптах автоматизации предпочтение следует отдавать более простым специализированным инструментам APT, таким как apt-get, apt-cache и др. Именно по этой причине apt выдает предупреждение в командах типа `apt list | grep sssd`, когда вывод утилиты перенаправляется дальше по конвейеру: «ВНИМАНИЕ: apt не имеет стабильного интерфейса командной строки. Используйте его в скриптах автоматизации с осторожностью».

Архитектура системы управления пакетами представлена на рисунке ниже.



Архитектура системы управления пакетами Debian-систем

Примечание

Имя утилиты `apt` совпадает с именем пакетного менеджера АРТ в целом, поэтому вам крайне важно внимательно следить за контекстом, в котором используются эти названия. Например, когда говорят, что Synaptic или Aptitude работают через АРТ, то подразумевают, что они обращаются к API из `libapt-pkg`, а не просто вызывают утилиту `apt` в командной строке. Утилита `apt` появилась значительно позже и предназначена для работы конечного пользователя в интерактивном режиме.

Управление списком источников

Менеджер АРТ загружает пакеты из одного или нескольких репозиториев (источников) программного обеспечения и устанавливает их на ваш компьютер. В роли репозитория может выступать CD-ROM, папка на диске (локальная или смонтированная по сети), удаленный HTTP или FTP сервер.

Подключение репозиториев Astra Linux

Управление общим списком источников осуществляется через редактирование файла `/etc/apt/sources.list*`. Приложения могут также создавать свои собственные списки источников в папке `/etc/apt/sources.list.d*`, используя произвольное имя файла с суффиксом `«.list»`. Редактировать файлы можно как обычным текстовым редактором, так и с помощью утилит.

Например, команда `sudo apt edit-sources` откроет файл `/etc/apt/sources.list` в редакторе по умолчанию и при его сохранении выполнит базовую проверку синтаксиса. Еще для редактирования источников вы можете воспользоваться утилитами Synaptic и Aptitude соответственно.

Ниже представлен пример правильно сконфигурированного файла `/etc/apt/sources.list` для ALSE 1.7.x.



```
# Astra Linux repository description https://wiki.astralinux.ru/x/0oLiC
#deb cdrom:[OS Astra Linux 1.7.4 1.7_x86-64 DVD ]/ 1.7_x86-64 contrib main non-free

#deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main/ 1.7_x86-64 main contrib non-free
#deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-update/ 1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-base/ 1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended/ 1.7_x86-64 main contrib non-free
```

Правильно настроенный список репозиториев sources.list

Для использования репозиториев Astra Linux достаточно подключить источник base (который уже включает main и update) и источник extended (который содержит дополнительное ПО, такое как Яндекс Браузер). Строки, указывающие на CD-ROM, и лишние репозитории можно не удалять, а просто закомментировать. После редактирования списка источников требуется обновить индекс пакетов командой `apt update`.

Разберем синтаксис строки источника:



Синтаксис строки репозитория

Где:

- **deb** — указывает на то, что репозиторий соответствует репозиторию бинарных файлов с предварительно скомпилированными пакетами. Для репозиториев с исходными кодами используют «deb-src».
- https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main/ — задает адрес репозитория. У интернет-репозиториев адрес начинается с «http(s)://», адреса локальных репозиториев начинаются с «file:». При добавлении репозитория с диска командой `apt-cdrom add` в файле появится строка «cdrom:[]/». В адресе вы видите, что используется поддерево репозиториев stable — это основная ветка для установки самых крайних оперативных обновлений, содержащая фиксированные ссылки, публикуемые в официальных бюллетенях безопасности. Установку оперативных обновлений следует

выполнять по инструкциям, опубликованным в бюллетенях. Есть еще ветка frozen, которая позволяет зафиксировать пакетную базу на определенную версию операционной системы, которая используется для установки комплексных приложений, которые начинают поддержку новых релизов операционной системы с задержкой.

- **1.7_x86-64** - кодовое имя дистрибутива, которое дополняет адрес, уточняя необходимый релиз продукта. В одном репозитории могут находиться пакеты сразу для нескольких релизов.

- **main contrib non-free** — это группа пакетов, объединенная по условиям использования:

- **non-free** — группа содержит пакеты, которые не соответствуют принципам свободного ПО, имеют патенты или другие юридические ограничения;

- **contrib** — группа содержит пакеты, которые сами по себе соответствуют принципам свободного ПО, но зависят от пакетов из группы «non-free» (т. е. не могут без них работать);

- **main** — группа содержит пакеты свободного ПО, которые не зависят от пакетов из групп «contrib» и «non-free».

Информацию о пакетах для обновления индекса менеджер возьмет из файла Release или InRelease, ссылки на которые формируются по следующей схеме, которая отражена на рисунке. Файлы InRelease отличаются от файлов Release только тем, что они содержат PGP-подписи. Содержание Packages-файлов после выполнения «apt-get update» кэшируется на локальном диске в папке `/var/lib/apt/lists`.

The diagram illustrates how the URL for the Release file is constructed from a repository line. The repository line is: `deb https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.1/repository-base 1.7_x86-64 main contrib non-free`. Arrows point from the base URL `https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.1/repository-base` and the architecture `1.7_x86-64` to the final URL: `https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.1/repository-base/dists/1.7_x86-64/Release`.

Способ получения адреса на файл Release

Примечание

Обратите внимание, что сервера репозитория Astra Linux не показывают вам все содержимое, если вы будете переходить по индексным страницам. Полная информация доступна только через Release/InRelease файлы.

Подключение сторонних репозиторий

Сторонние репозитории можно вписать в файл `sources.list`, но правильнее создать отдельный файл `имя_репозитория.list` в каталоге `/etc/apt/sources.list.d/`, а также добавить в систему его открытый ключ, чтобы доверять этому репозиторию. Разберем этот процесс на примере добавления репозитория yandex-disk.

Создадим файл `/etc/apt/sources.list.d/yandex-disk.list`, добавим в него строку с репозиторием yandex-disk и обновим список пакетов в локальном кэше.

```
sa@astra:~$ echo "deb http://repo.yandex.ru/yandex-disk/deb/ stable main" | sudo tee
/etc/apt/sources.list.d/yandex-disk.list
deb http://repo.yandex.ru/yandex-disk/deb/ stable main
```

Обновим индекс пакетов `sudo apt update`. Скачаем файл с открытым ключом репозитория и добавим его в систему.

```
sa@astra:~$ sudo wget http://repo.yandex.ru/yandex-disk/YANDEX-DISK-KEY.GPG
--2025-03-28 13:25:02-- http://repo.yandex.ru/yandex-disk/YANDEX-DISK-KEY.GPG
Распознаётся repo.yandex.ru (repo.yandex.ru)... 213.180.204.183, 2a02:6b8::183
Подключение к repo.yandex.ru (repo.yandex.ru) 213.180.204.183 :80... соединение
восстановлено.
HTTP-запрос отправлен. Ожидание ответа... 200 OK
Длина: 1760 (1,7K) [application/octet-stream]
Сохранение в: «YANDEX-DISK-KEY.GPG»
YANDEX-DISK-KEY.GPG
100% [=====>] 1,72K --.- KB/s
2025-03-28 13:25:03 (206 MB/s) - «YANDEX-DISK-KEY.GPG» сохранён [1760/1760]
sa@astra:~$ sudo apt-key add YANDEX-DISK-KEY.GPG
OK
```

Теперь обновим локальный кэш пакетов и убедимся, что репозиторий был добавлен без ошибок и что пакет `yandex-disk` теперь доступен нам.

```
sa@astra:~$ sudo apt update
Пол:1 http://repo.yandex.ru/yandex-disk/deb stable InRelease [3 900 B]
Игн:2 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main 1._x86-64
InRelease
Сущ:3 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-update 1._x86-64
InRelease
Сущ:4 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-base 1._x86-64
InRelease
Сущ:5 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended 1._x86-64
InRelease
Сущ:6 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main 1._x86-64 Release
Пол:7 http://repo.yandex.ru/yandex-disk/deb stable/main amd64 Packages [561 B]
Пол:8 http://repo.yandex.ru/yandex-disk/deb stable amd64 Contents (deb) [244 B]
Получено 4 705 B за 1с (3 220 B/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
sa@astra:~$ apt search yandex-disk
Сортировка... Готово
Полнотекстовый поиск... Готово
yandex-disk/now 0.1.6.1080 amd64 [установлен, локальный]
Yandex.Disk keeps your files with you at all times.
```

Отключим репозиторий `yandex-disk`, закомментировав его в файле `/etc/apt/sources.list.d/yandex-disk.list`, обновим кэш и убедимся, что пакет `yandex-disk` больше не может быть найден.

```
sa@astra:~$ sudo sed -i 's/deb/#deb/g' /etc/apt/sources.list.d/yandex-disk.list
sa@astra:~$ cat /etc/apt/sources.list.d/yandex-disk.list
#deb http://repo.yandex.ru/yandex-disk/deb/stable main
sa@astra:~$ sudo apt update
Игн:1 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main 1.7_x86-64
InRelease
Сущ:2 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-update 1.7_x86-64
InRelease
Сущ:3 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-base 1.7_x86-64
InRelease
Сущ:4 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended 1.7_x86-64
```

```
InRelease
Сущ:5 https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main 1.7_x86-64
Release
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
sa@astra:~$ apt search yandex-disk
Сортировка... Готово
Полнотекстовый поиск... Готово
```

Примечание

Если добавить строку `[trusted=yes]` между словом `deb` и путем до репозитория, например, «`deb [trusted=yes] https://...`», то АРТ будет использовать файл `Release`, доверяя этому источнику без проверки подписей. Однако это подходит только в том случае, если компьютеры не покидают пределов локальной сети. Отключение проверки подписей для источников на переносимых устройствах создает большую угрозу безопасности, т.к. злоумышленники смогут перехватить запрос компьютера и отдать ему пакет с каким-нибудь зловредом вместо запрашиваемого приложения.

Управление ПО с помощью АРТ

Как мы уже знаем, для управления в качестве интерфейсов для взаимодействия с АРТ можно использовать утилиты `apt`, `apt-get`, `aptitude` и `synaptic`. Рассмотрим работу с каждой из них подробнее.

Утилиты `apt` и `apt-*`

Для базового управления пакетами используются следующие команды утилиты `apt`:

- **update** — загружает информацию о пакетах из репозитория и обновляет так называемый индекс пакетов, т.е. встроенную базу данных менеджера АРТ, в которой хранятся записи о доступных пакетах из репозитория, включенных в систему.

- **install** — устанавливает один и более пакетов, перечисленные через пробел. Например, `sudo apt install package1 package2`. Стоит отметить, что нужно указывать не полное имя `deb`-файла, а его внутреннее имя (свойство `Package`), например, `sudo apt install htop`.

С помощью знака равно можно выбрать конкретную версию пакета для установки, например, «`sudo apt install package1=2.15`», а с помощью суффикса «`-`» выполнить удаление пакета вместо установки, например: `sudo apt install package1 package2-`.

- **remove** — удаляет пакеты, сохраняя конфигурационные файлы. Если необходимо не удалить пакет, а установить, то можно воспользоваться суффиксом «`+`», например: `sudo apt remove package1 package2+`.

- **purge** — действует аналогично команде `remove`, но удаляет в том числе файлы конфигурации, относящиеся к этому пакету.

- **reinstall** — если по какой-то причине вы удалили один из файлов пакета или вам требуется откатить настройки приложения до заводских, то этой командой вы сможете выполнить повторную установку пакета.

- **search** — выполняет поиск пакетов, в том числе по кратким описаниям пакетов.

Например, с помощью команды `apt search 'file manager'` вы сможете найти все пакеты, которые имеют какое-то отношение к файловым менеджерам. Строка поиска позволяет использовать в том числе регулярные выражения.

- **show** — показывает информацию о пакете из файла `control`.

- **list** — выводит список пакетов по заданным критериям.

Для управления обновлениями используются команды:

- **upgrade** — устанавливает доступные обновления для всех пакетов в системе из всех доступных источников (`sources.list`).

Внимание

Использовать команду «`apt upgrade`» категорически запрещается, т. к. она не удаляет устаревшие пакеты, даже если это требуется для обновления приложений, что может нарушить работу операционной системы.

Вместо `upgrade` следует использовать хотя бы `apt dist-upgrade`, а лучше специальную утилиту «`astra-update -a -r`», где ключ `-a` включает автоматическую установку, а ключ `-r` ограничивает обновление списком репозитория из файла `/etc/apt/sources.list`.

- **dist-upgrade** — действует аналогично `upgrade`, но при необходимости устанавливает новые зависимости или удаляет старые.

- **full-upgrade** — действует аналогично `dist-upgrade`, но может удалять уже установленные пакеты, если они больше не нужны или заменены другими пакетами.

- **autoremove** — используется для очистки системы от пакетов, которые были установлены для разрешения зависимостей и в настоящий момент более не используются, т.к. пакеты, зависящие от них, ранее были удалены из системы.

- **edit-sources** — позволяет редактировать файлы `sources.list` в редакторе по умолчанию и производит базовую проверку синтаксиса при его редактировании.

Рассмотрим работу с этими опциями на практике. Обновим список доступных пакетов командой `sudo apt update`.

```
sa@astra:~$ sudo apt update
Игн:1 https://download.astralinux.ru/astra/stable/1.7_x86-64/ repository-main 1.7_x86-64
InRelease
Сущ:2 https://download.astralinux.ru/astra/stable/1.7_x86-64/ repository-update 1.7_x86-64
InRelease
. . .
```

```
Пол:16 https://download.astralinux.ru/astra/stable/1.7_x86-64/ repository-main 1.7_x86-64  
amd64 Contents (deb) [1 621 kB]  
Получено 19,2 МВ за 10с (2 017 kB/s)
```

```
Чтение списков пакетов... Готово  
Построение дерева зависимостей  
Чтение информации о состоянии... Готово  
Может быть обновлён 681 пакет. Запустите «apt list --upgradable» для показа.
```

Как видите, утилита запросила информацию из всех перечисленных в `sources.list` репозиториях и обновила список доступных пакетов. Кроме того, она предлагает нам посмотреть список доступных пакетов для обновления командой `apt list --upgradable`.

Опция `search` позволяет производить поиск пакетов по полям в информации о пакете (файле `control`), но для поиска пакетов по входящим в него файлам необходимо воспользоваться утилитой `apt-file`, которая не входит в состав пакета `apt`:

```
sa@astra:~$ sudo apt install apt-file  
Чтение списков пакетов... Готово  
Построение дерева зависимостей  
Чтение информации о состоянии... Готово  
Будут установлены следующие дополнительные пакеты:  
  libapt-pkg-perl libexporter-tiny-perl liblist-moreutils-perl libregexp-assemble-perl  
Следующие НОВЫЕ пакеты будут установлены:  
  apt-file libapt-pkg-perl libexporter-tiny-perl liblist-moreutils-perl libregexp-assemble-perl
```

При этом утилита построила дерево зависимостей, сформировала список дополнительных пакетов для установки, рассчитала объем необходимого места для их загрузки и установки и в интерактивном режиме запросила подтверждение на установку этих пакетов. После получения подтверждения она скачала все необходимые пакеты, распаковала их и настроила. На последнем шаге были автоматически обновлены страницы в `man-db`.

Воспользуемся этой утилитой, чтобы найти пакет, в состав которого входит утилита `pstree`. Для этого выполним команду `apt-file search pstree`.

```
sa@astra:~$ apt-file search pstree  
psmisc: /usr/bin/pstree  
psmisc: /usr/bin/pstree.x11  
psmisc: /usr/share/man/man1/pstree.1.gz  
psmisc: /usr/share/man/man1/pstree.x11.1.gz  
psmisc: /usr/share/pixmaps/pstree16.xpm  
psmisc: /usr/share/pixmaps/pstree32.xpm  
python-psutil-doc: /usr/share/doc/python-psutil-doc/examples/pstree.py
```

Как видим, этот пакет называется `psmisc`, и командой `apt show psmisc` мы можем вывести о нем дополнительную информацию:

```
sa@astra:~$ apt show psmisc  
Package: psmisc  
Version: 23.2-1+b1  
Priority: optional  
Section: admin
```



```
Maintainer: Craig Small <csmall@debian.org>
Installed-Size: 643 kB
Depends: libc6 (>= 2.14), libselinux1 (>= 1.32), libtinfo6 (>= 6)
Replaces: procps (< 1:1.2)
Homepage: http://psmisc.sf.net/
Download-Size: 121 kB
APT-Manual-Installed: no
APT-Sources: https://download.astralinux.ru/astra/stable/1.7_x86-64/repository- update 1.
7_x86-64/main amd64 Packages
...
```

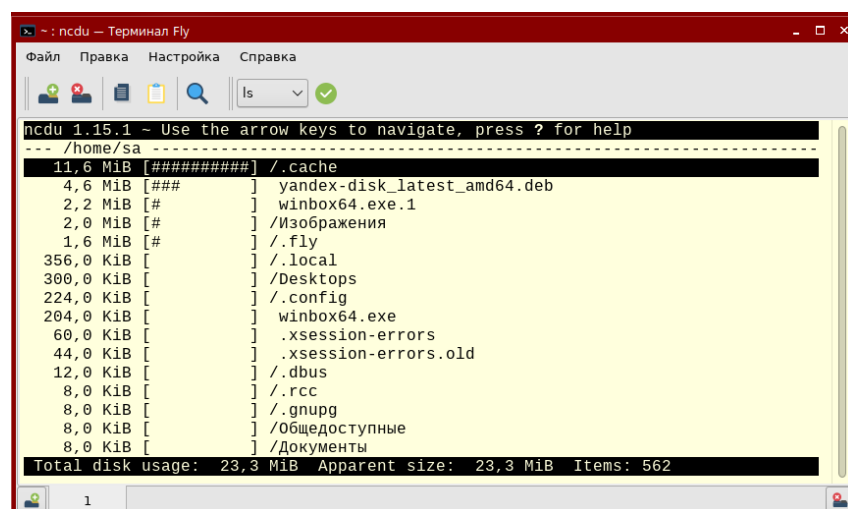
В опции `search` утилиты `apt` можно применять регулярные выражения, например, для поиска пакета веб-сервера `nginx` необходимо ввести команду `apt search ^nginx$`. Сравните с выводом команды без метасимвола `$` `apt search "nginx"`.

```
sa@astra:~$ apt search "^nginx$"
Сортировка... Готово
Полнотекстовый поиск... Готово
nginx/stable 1.26.0-1 amd64
small, powerful, scalable web/proxy server
```

Теперь установим какой-нибудь пакет в систему, например, анализатор диска, указывающий на наиболее заполненные каталоги - утилиту `ncdu`, командой `sudo apt install ncdu`.

```
sa@astra:~$ sudo apt install ncdu
[sudo] пароль для sa:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Следующие НОВЫЕ пакеты будут установлены:
 ncdu
...
```

Теперь мы можем воспользоваться ей, чтобы оценить размер каталогов в домашней директории командой `ncdu ~`.



```
ncdu 1.15.1 ~ Use the arrow keys to navigate, press ? for help
--- /home/sa -----
11,6 MiB [#####] /.cache
4,6 MiB [###] ] yandex-disk_latest_amd64.deb
2,2 MiB [#] ] winbox64.exe.1
2,0 MiB [#] ] /Изображения
1,6 MiB [#] ] /.fly
356,0 KiB [ ] ] /.local
300,0 KiB [ ] ] /Desktops
224,0 KiB [ ] ] /.config
204,0 KiB [ ] ] winbox64.exe
60,0 KiB [ ] ] .xsession-errors
44,0 KiB [ ] ] .xsession-errors.old
12,0 KiB [ ] ] /.dbus
8,0 KiB [ ] ] /.rcc
8,0 KiB [ ] ] /.gnupg
8,0 KiB [ ] ] /Общедоступные
8,0 KiB [ ] ] /Документы
Total disk usage: 23,3 MiB Apparent size: 23,3 MiB Items: 562
```

Размер каталогов в домашней директории в ncdu

Переустановить эту утилиту (например, при повреждении её файлов) можно командой `sudo apt reinstall ncdu`.

```
sa@astra:~$ sudo apt reinstall ncdu
Чтение списков пакетов... Готово
Построение дерева зависимостей
. . .
Распаковывается ncdu (1.15.1-1) на замену (1.15.1-1) ...
Настраивается пакет ncdu (1.15.1-1) ...
Обрабатываются триггеры для man-db (2.8.5-2) ...
Обрабатываются триггеры для xserver-xorg-core (2:1.20.14-1ubuntu1astra.se17) ...
update exec ids due to /usr/bin changed
```

Решив, что эта утилита нам больше не потребуется, её можно удалить командой `sudo apt remove ncdu`.

```
sa@astra:~$ sudo apt remove ncdu
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Следующие пакеты будут УДАЛЕНЫ:
 ncdu
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 1 пакетов, и 681
пакетов
не обновлено.
После данной операции объём занятого дискового пространства уменьшится на 102 kB.
Хотите продолжить? [Д/н] у
(Чтение базы данных ... на данный момент установлено 211293 файла и каталога.)
Удаляется ncdu (1.15.1-1) ...
Обрабатываются триггеры для man-db (2.8.5-2) ...
Обрабатываются триггеры для xserver-xorg-core (2:1.20.14-1ubuntu1astra.se17) ...
update exec ids due to /usr/bin changed
```

Утилита сообщит объем пространства, которое будет высвобождено после удаления пакета, и запросит подтверждение на удаление пакета.

Теперь установим пакет Visual Studio Code и расширение `rogalmic.bash-debug` из репозитория на локальном диске. `Rogalmic.bash-debug` - удобный инструмент отладки bash-скриптов, который выполняет скрипт по шагам, позволяет устанавливать точки остановки и отображает значения переменных.

Загрузим deb-пакет VS Code с официального сайта: <https://code.visualstudio.com/>. По умолчанию файл сохранится в «Загрузки». В терминале измените текущую директорию на `/home/sa/Загрузки`, далее выполните команду:

```
sa@astra:~/Загрузки$ sudo apt install ./code_1.98.2-1741788907_amd64.deb
[sudo] пароль для sa:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Заметьте, вместо «./code_1.98.2-1741788907_amd64.deb» выбирается «code»
Некоторые пакеты не могут быть установлены. Возможно, то, что вы просите,
неосуществимо, или же вы используете нестабильную версию дистрибутива, где
запрошенные вами пакеты ещё не созданы или были удалены из Incoming.
Следующая информация, возможно, вам поможет:

Следующие пакеты имеют неудовлетворённые зависимости:
code : Зависит: libxkbfile1 (>= 1:1.1.0) но 1:1.0.9-2+b1 должен быть установлен
Е: Невозможно исправить ошибки: у вас зафиксированы сломанные пакеты.
```

Процесс установки прерывается с ошибкой. Для ее исправления нужно скачать недостающие пакеты. Для этого переходим по ссылке <https://packages.debian.org/>. В поле «Поиск по каталогу пакетов» вводим имя недостающего пакета `libxkbfile1`, нажимает «Поиск». Нас интересуют точные совпадения. Поэтому переходим по ссылке:

Точные совпадения

Пакет `libxkbfile1`

- [bookworm \(stable\)](#) (libs): библиотека для работы с файлами описаний клавиатур в X11
1:1.1.0-1: amd64 arm64 armel armhf i386 mips64el mipsel ppc64el s390x

Листаем страницу до конца и выбираем пакет в зависимости от архитектуры (в нашем случае это `amd64`).

Загрузка `libxkbfile1`

Архитектура	Размер пакета	В установленном виде	Файлы
amd64	73,4 Кб	180,0 Кб	[список файлов]
arm64	69,8 Кб	179,0 Кб	[список файлов]
armel	64,3 Кб	166,0 Кб	[список файлов]
armhf	65,6 Кб	130,0 Кб	[список файлов]
i386	77,1 Кб	194,0 Кб	[список файлов]
mips64el	67,0 Кб	197,0 Кб	[список файлов]
mipsel	66,2 Кб	183,0 Кб	[список файлов]
ppc64el	79,4 Кб	223,0 Кб	[список файлов]
s390x	68,2 Кб	187,0 Кб	[список файлов]

Выбираем зеркало для загрузки файла. Разрешаем загрузку, в случае предупреждения о вероятной угрозе безопасности.

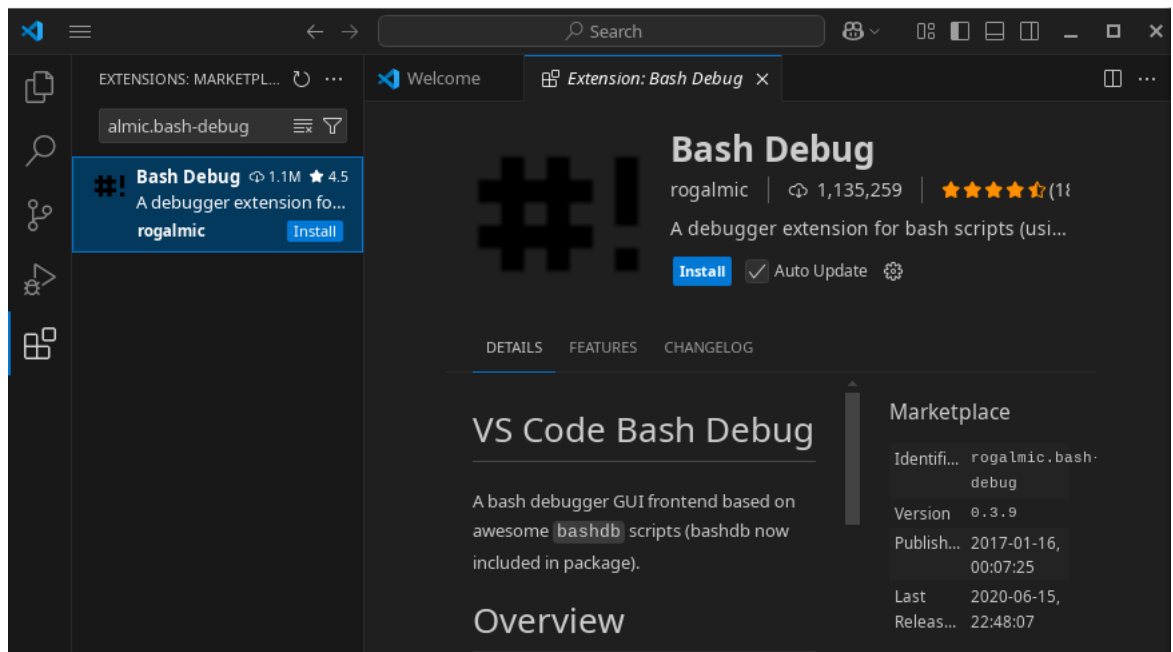
После этого снова переходим в терминал и запускаем установку недостающего пакета.

```
sa@astra:~/Загрузки$ sudo apt install ./libxkbfile1_1.1.0-1_amd64.deb
[sudo] пароль для sa:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Заметьте, вместо «./libxkbfile1_1.1.0-1_amd64.deb» выбирается «libxkbfile1»
Следующие пакеты будут обновлены:
  libxkbfile1
Обновлено 1 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов,
и 0 пакетов не обновлено.
Необходимо скачать 0 В/75,2 кВ архивов.
После данной операции объём занятого дискового пространства уменьшится на 3 072 В.
Пол:1 /home/sa/Загрузки/libxkbfile1_1.1.0-1_amd64.deb libxkbfile1 amd64 1:1.1.0-1
[75,2 кВ]
(Чтение базы данных ... на данный момент установлено 287283 файла и каталога.)
Подготовка к распаковке .../libxkbfile1_1.1.0-1_amd64.deb ...
Распаковывается libxkbfile1:amd64 (1:1.1.0-1) на замену (1:1.0.9-2+b1) ...
Настраивается пакет libxkbfile1:amd64 (1:1.1.0-1) ...
```

Далее необходимо снова запустить команду установки пакета VS Code:

```
sa@astra:~/Загрузки$ sudo apt install ./code_1.98.2-1741788907_amd64.deb
[sudo] пароль для sa:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Заметьте, вместо «./code_1.98.2-1741788907_amd64.deb» выбирается «code»
Следующие НОВЫЕ пакеты будут установлены:
code
...
```

После успешного завершения установки запустите VS Code: **Пуск ▸ Разработка ▸ Visual Studio Code**. Для установки расширения `rogalmic.bash-debug` введите соответствующее название в строке поиска. Нажмите кнопку «Install». Если возникло предупреждение о том, что источник является недоверенным, нажмите «Trust Publisher & Install».



Выберите нужный скрипт-файл, чтобы открыть его на редактирование. Нажмите горячее сочетание клавиш `Ctrl + Shift + G` для замены левой вкладки на «Run and Debug». Перейдите по ссылке `create a launch.json file` и в открывшемся меню выберите «Bash Debug».

Для вывода пакетов, которые можно обновить, можно воспользоваться командой `apt list --upgradable` (а командой `apt list --installed` можно вывести список всех установленных пакетов в системе):

```
sa@astra:~$ apt list --upgradable
Вывод списка... Готово
afick/stable,stable 3.8.0-astra1.7+ci7 amd64 [может быть обновлён с: 2.11.1-0astra.se2]
alsa-utils/stable,stable 1.2.8-1+b1 amd64 [может быть обновлён с: 1.1.8-2]
ark/stable,stable 4:22.12.3-1astra3 amd64 [может быть обновлён с: 4:22.08.1-1astra2]
...
xserver-xorg-video-vmware/stable,stable 1:13.3.0-3.1astra1 amd64 [может быть обновлён с: 1:13.3.0-2astra2]
```

Команда `sudo astra-update -A -r` обновит все пакеты в системе автоматически. Если по каким-либо причинам вам необходимо обновить не всю систему, а только отдельный пакет, то можно воспользоваться командой `sudo apt-get install --only-upgrade astra-admin-events`.

```
sa@astra:~$ sudo apt-get install --only-upgrade astra-admin-events
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
...
```

Теперь обновим систему командой `sudo astra-update -A -r` (этот процесс может занять некоторое время, и вы пока можете перейти к прочтению следующего менеджера пакетов `aptitude`).

```
sa@astra:~$ sudo astra-update -A -r
Параметры запуска: -A -r

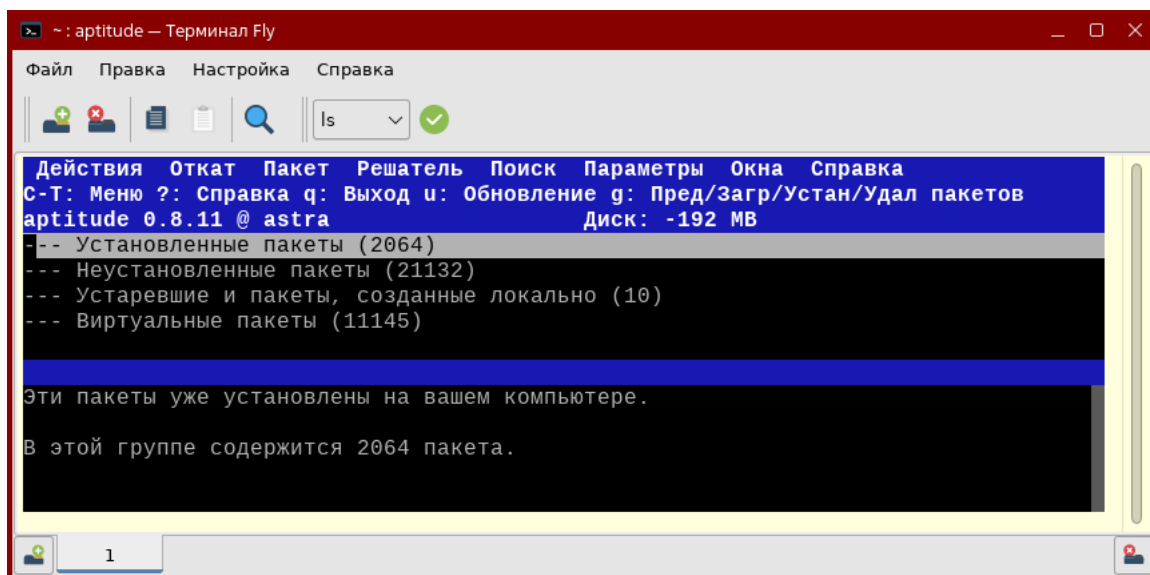
Источники из /etc/apt/sources.list:
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-main/ 1.7_x86-64 main
contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-update/ 1.7_x86-64 main
contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-base/ 1.7_x86-64 main
contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended/ 1.7_x86-64
main
contrib non-free

Обнаруженные компоненты: contrib main non-free
Отключение функций безопасности не требуется
...
Обновление успешно установлено
Для применения изменений в ядре или модулях нужна перезагрузка
Лог сохранен в /var/log/astra_update_202308151047.log
```

Менеджер пакетов `aptitude`

Менеджер пакетов `aptitude` позволяет выполнять те же операции с пакетами, что и `apt`, но обладает псевдографическим интерфейсом и поддерживает работу с мышью. В то же время вы можете использовать и командную строку, причем синтаксис команд такой же, как у `apt`.

Запускается приложение командой `aptitude`, а если необходимо не только просматривать информацию, но и устанавливать/обновлять/удалять пакеты, то `sudo aptitude`.



Псевдо-графический менеджер пакетов aptitude

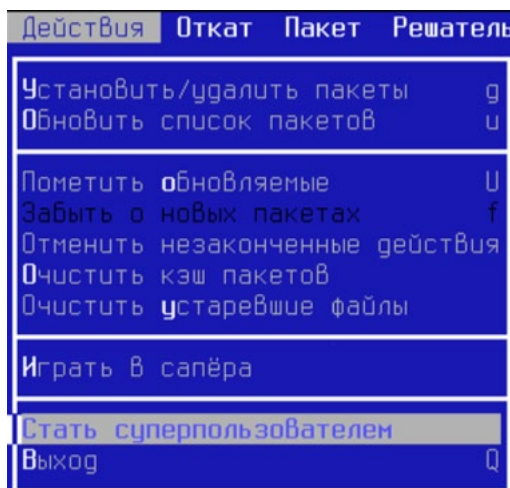
В главном окне приложения представлено дерево пакетов, свернутое до первого уровня. Для навигации по дереву используйте клавиши стрелок и мышь.

В верхней части окна расположено меню. Для его вызова с клавиатуры необходимо нажать сочетание `Ctrl + <Пробел>` или `F10`.

Для установки, обновления или удаления пакета, его необходимо пометить. Для этого необходимо воспользоваться меню «Пакет» или ввести:

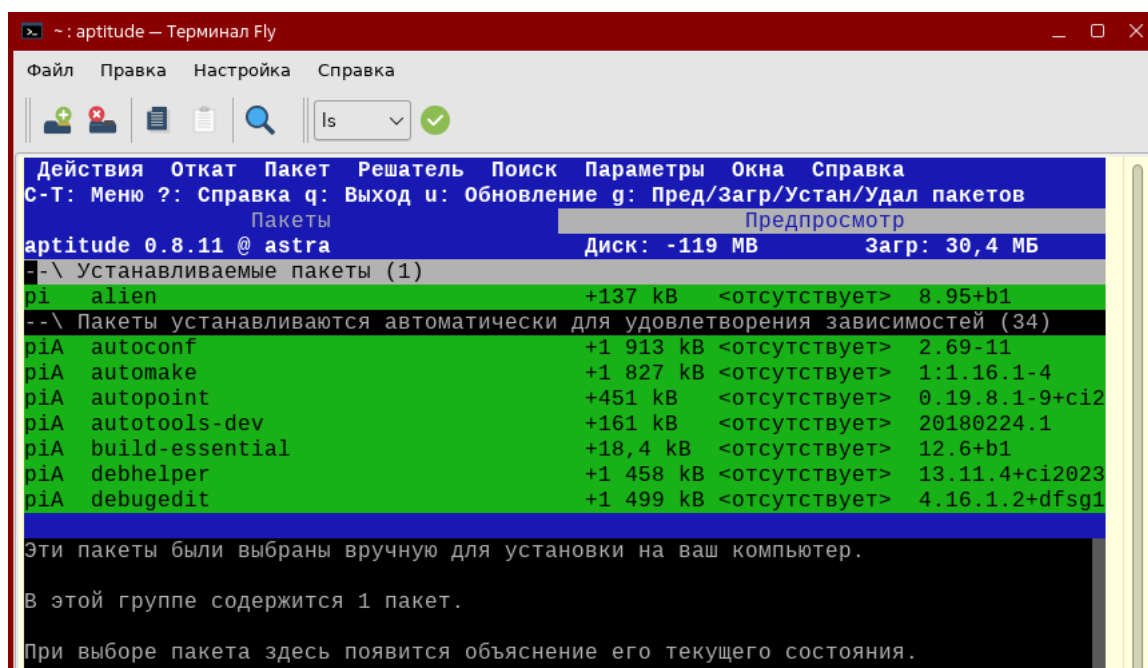
- `+` (плюс) — для установки или обновления;
- `-` (минус) — для удаления (аналог `aptitude remove`);
- `_` (подчеркивание) — для полного удаления (аналог `aptitude purge`).

Если утилита была запущена без прав суперпользователя, а они потребовались в процессе работы, то в меню «Действия» можно выбрать пункт «Стать суперпользователем», и утилита будет перезапущена с повышенными привилегиями.



Меню действия утилиты aptitude

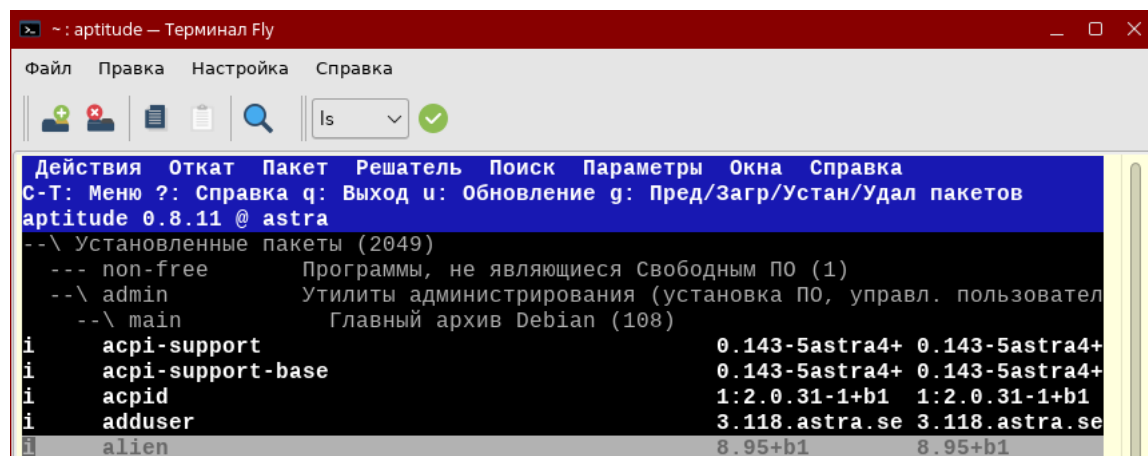
После выделения пакет подсвечивается зеленым (для удаления красным).



Построение дерева зависимостей и предпросмотр

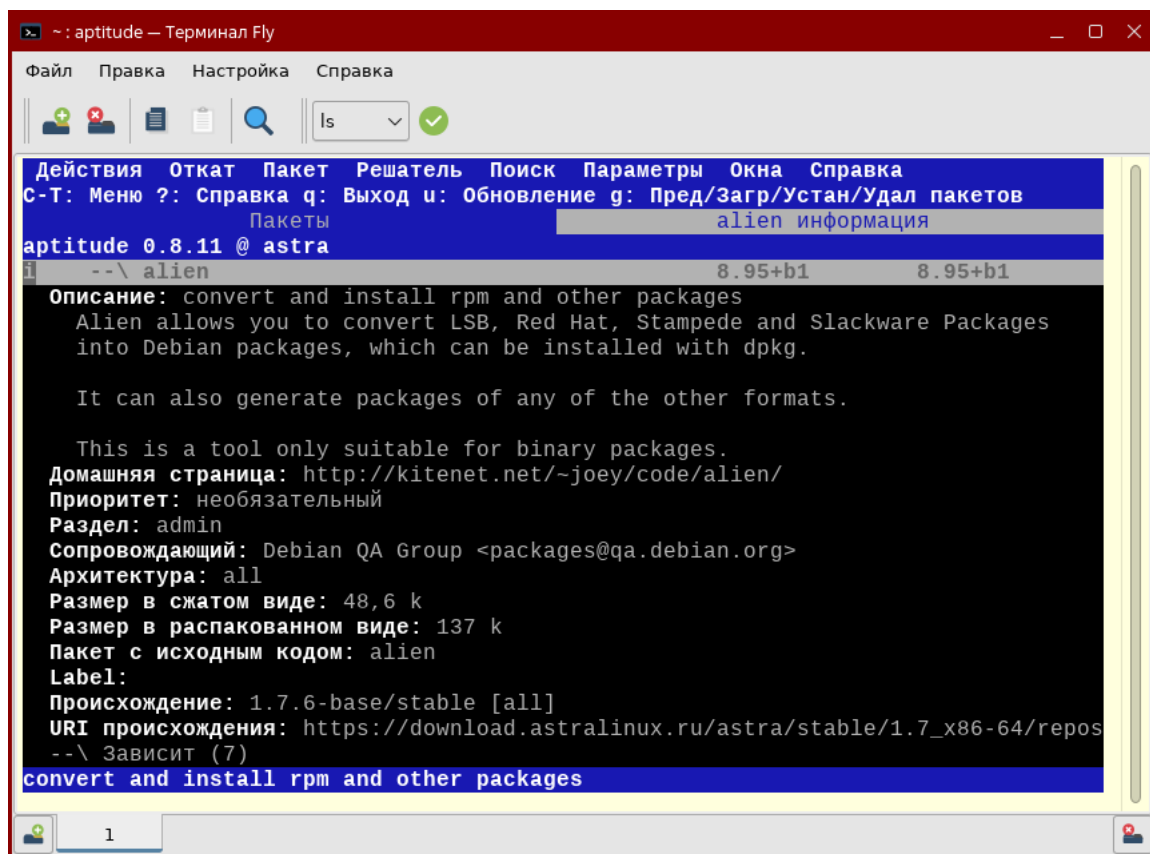
При этом произойдёт построение дерева зависимостей, и для подтверждения установки необходимо будет нажать **g** еще раз. В консоль будет выведен лог производимых операций.

Найдем пакет alien через поиск и убедимся, что он был установлен в системе.



Найден пакет alien в ветке установленные пакеты

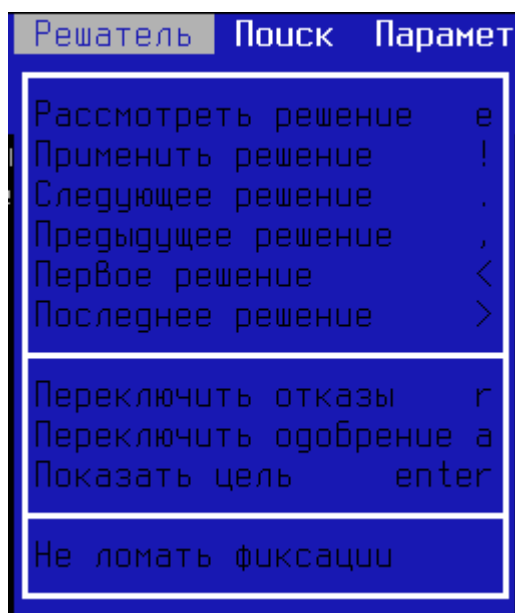
Для просмотра информации необходимо открыть пакет двойным щелчком мыши или клавишей **Enter**, при этом информация будет выведена в новом окне. Для закрытия окна необходимо нажать **q** или воспользоваться помощью меню.



Окно просмотра информации

Для перехода между окнами используйте клавиши **F7** и **F8**.

Меню «Решатель» позволяет удобно разрешать конфликты зависимостей в интерактивном режиме.

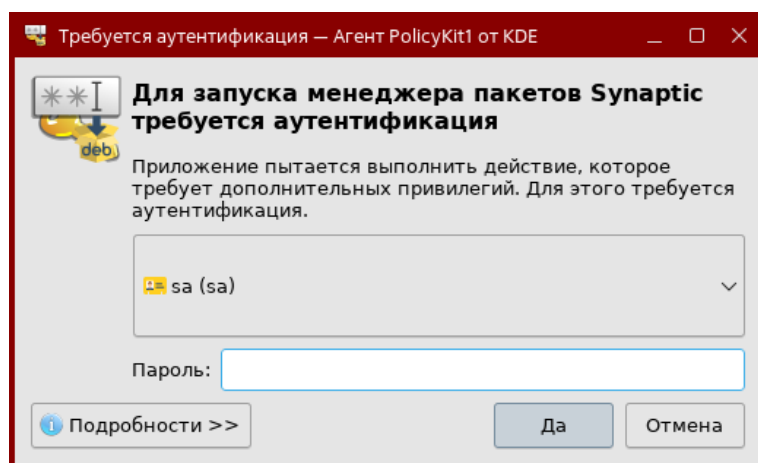


Меню решатель конфликтов aptitude

Для закрытия программы необходимо нажать **Q** или воспользоваться командой из меню **Действия** **Выход**.

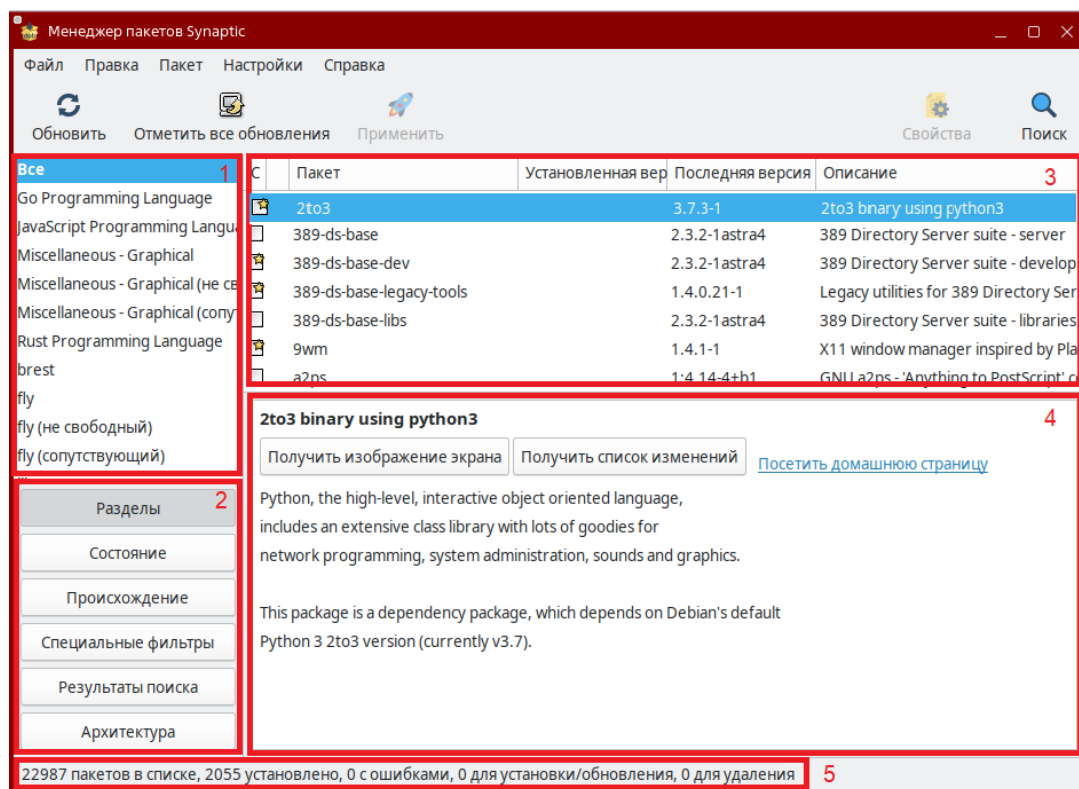
Менеджер пакетов Synaptic

Менеджер пакетов Synaptic позволяет управлять ПО в Astra Linux через графический интерфейс. Запустить его можно из меню **Пуск ▸ Системные ▸ Менеджер пакетов Synaptic**. Для работы с ним нужны будут права суперпользователя.



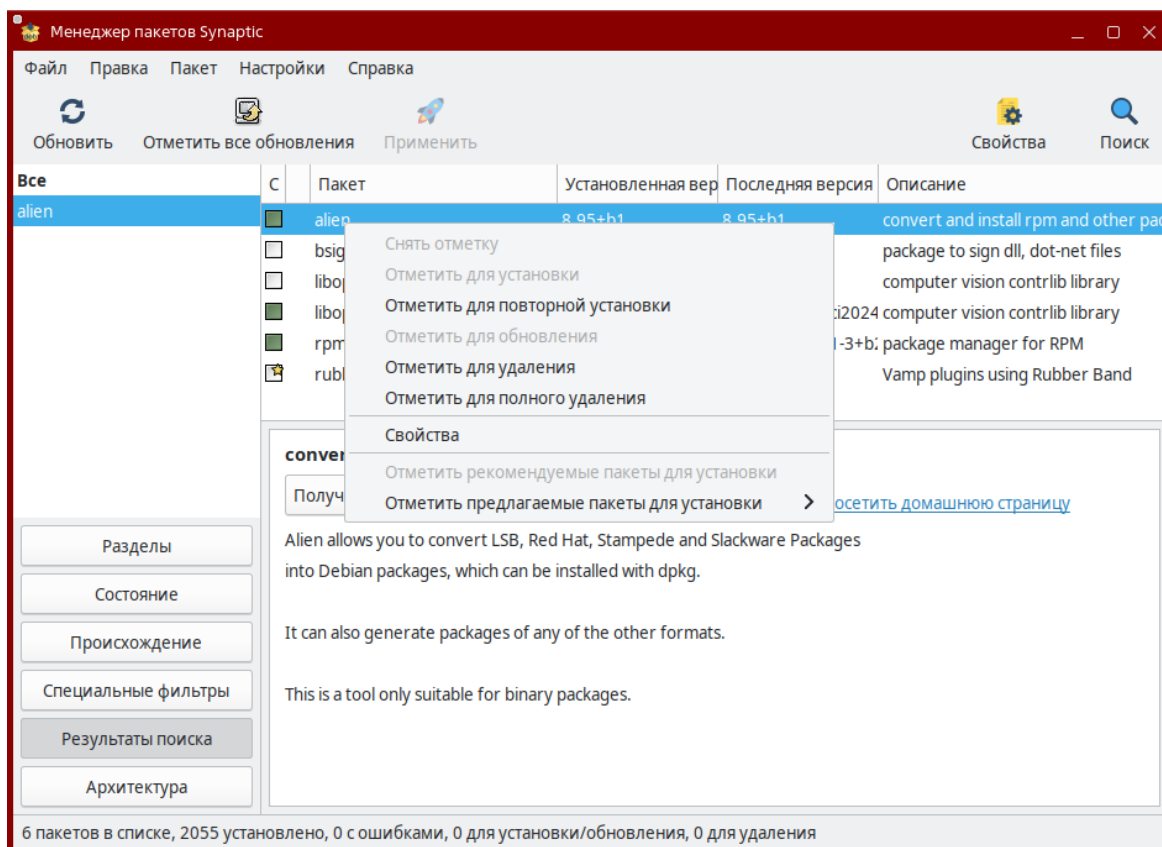
Подтверждение прав суперпользователя

Synaptic имеет интуитивно понятный интерфейс работы. В левой части окна представлены категории пакетов (1) в зависимости от выбранного способа разделения (2). В правой части окна представлены список пакетов (3) и описание выделенного пакета (4). В нижней части окна расположена строка состояния.

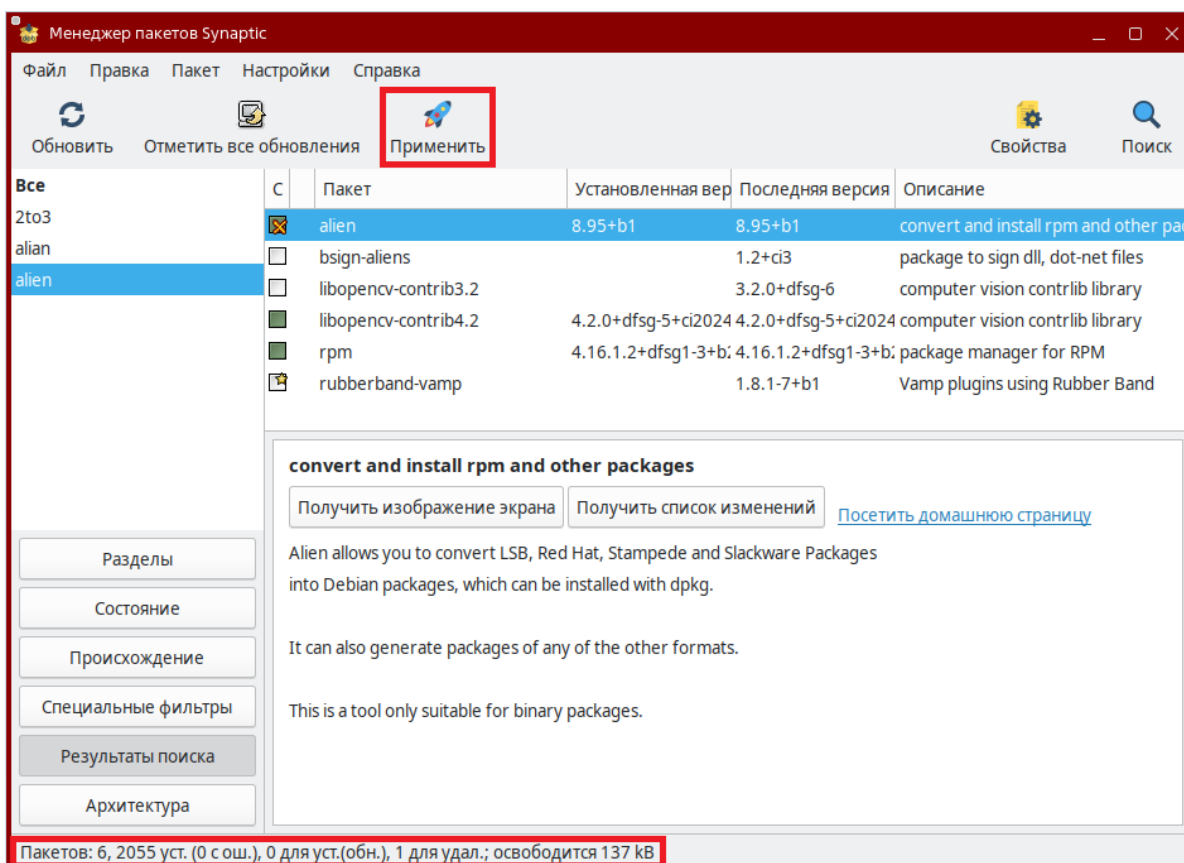


Графический интерфейс Synaptic

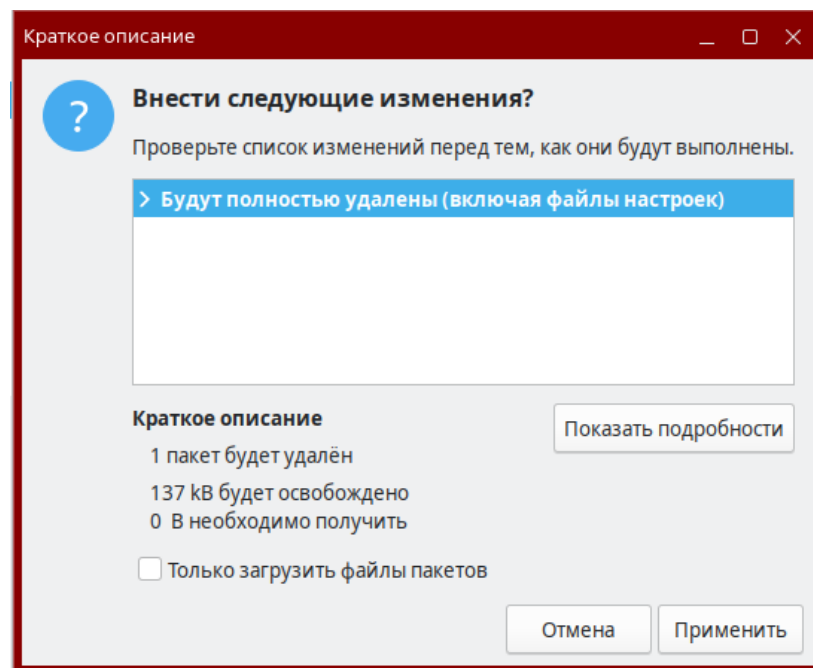
Удалим установленный нами через aptitude пакет alien с помощью Synaptic.



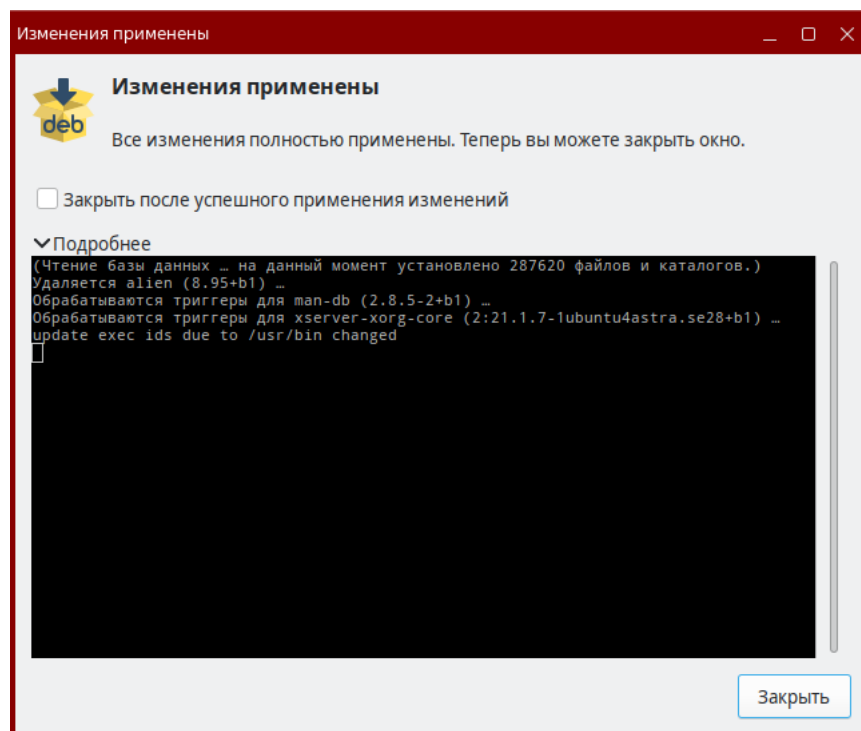
Контекстное меню по выбранному пакету alien



Для применения удаления/установки нажимаем «Применить»

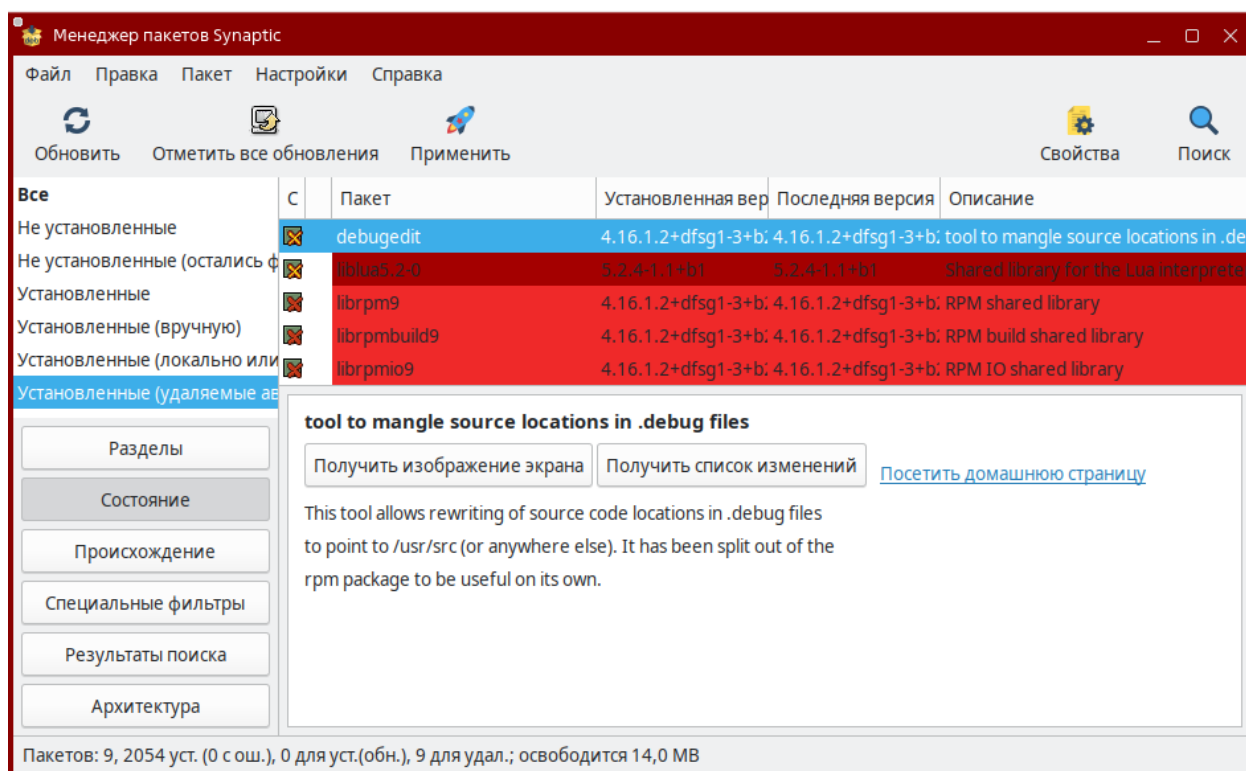


Описание того, что будет сделано после подтверждения

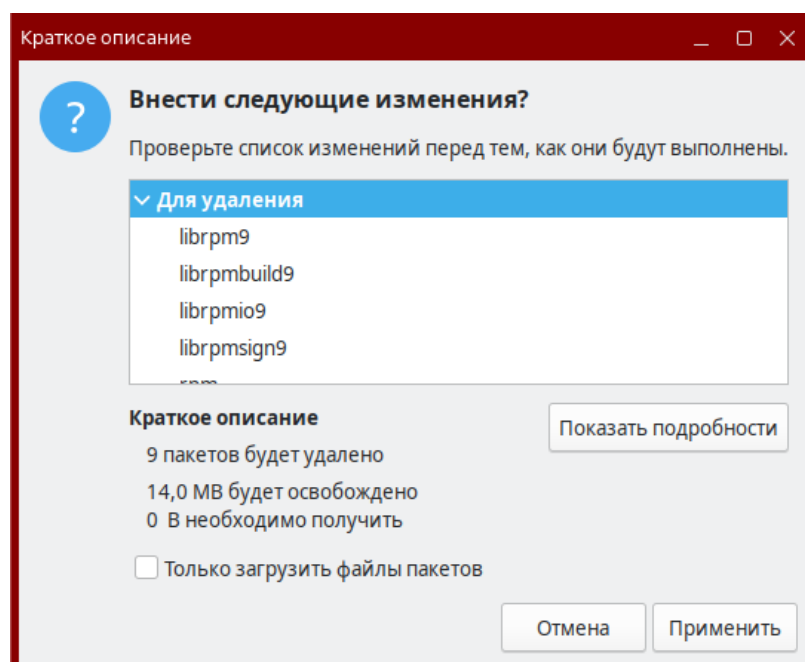


Запуск API dpkg в терминале с просмотром о ходе выполнения

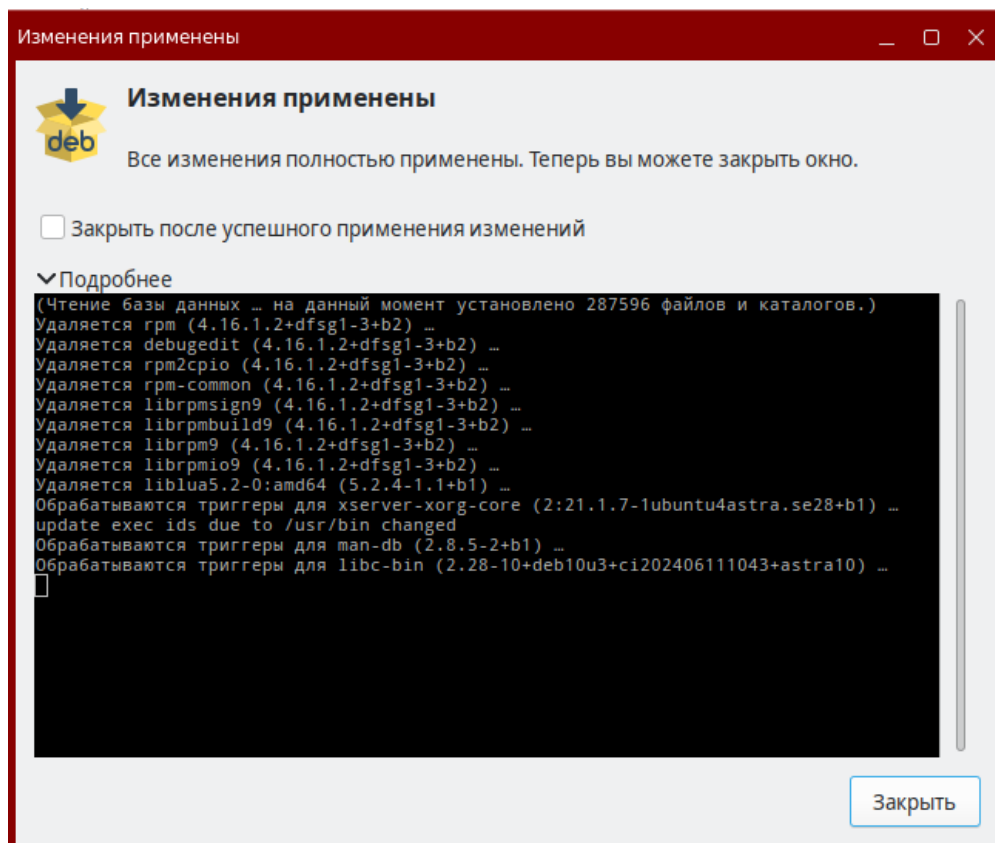
И удалим все пакеты из категории «Установленные (удаляемые автоматически)».



Выбор пакетов для удаления

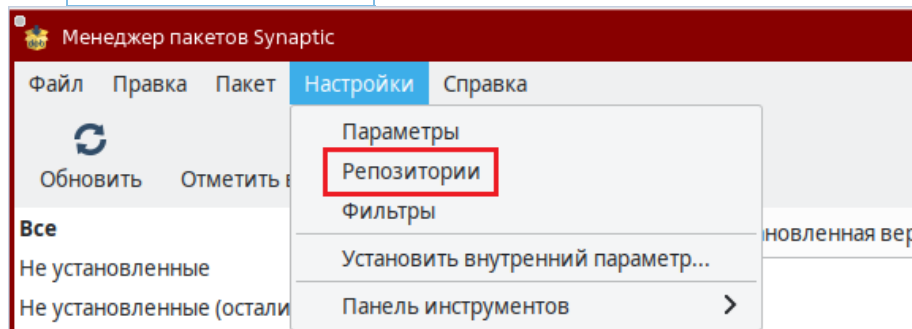


Окно подтверждение с краткой информацией

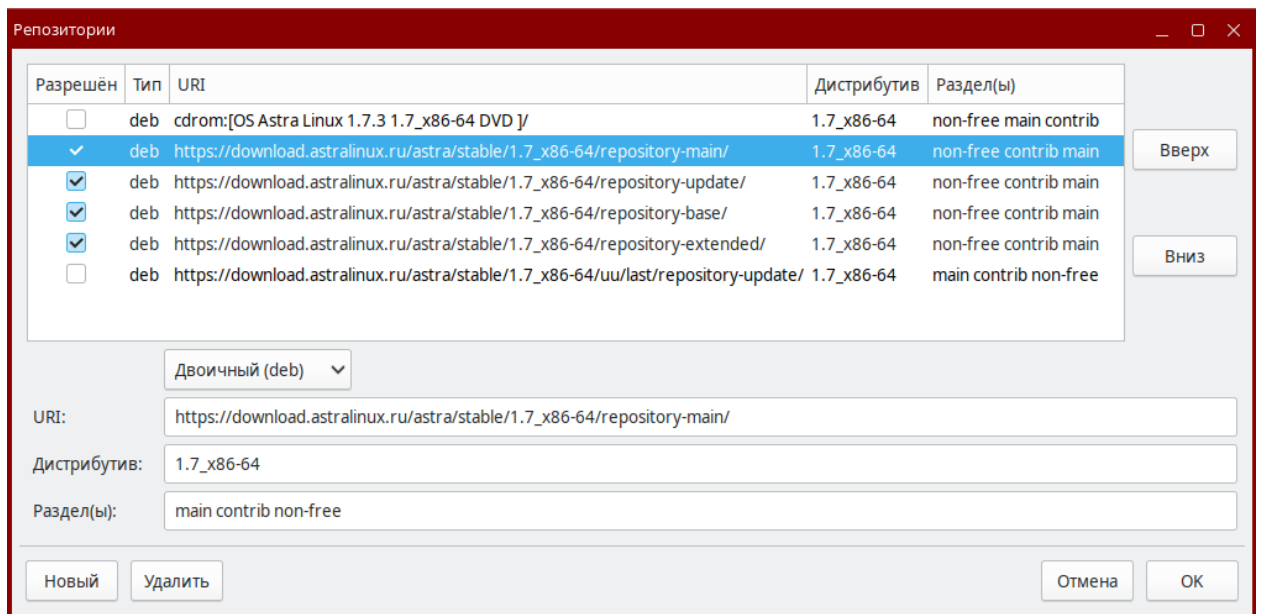


Ход выполнения удаления в терминале

Источниками пакетов можно управлять в окне «Репозитории», вызвать которое можно из меню **Настройки > Репозитории**.



Меню настроек Synaptic



Окно настроек репозиториев Synaptic

Внутреннее устройство deb-пакета

Обычные пакеты

Deb-пакет — это архив, созданный с помощью архиватора `ar`, внутри которого находятся не только файлы приложения, но и еще несколько служебных файлов, которые нужны для того, чтобы добавить гибкости.

Пакеты используют строгие правила именования, подчиняющиеся следующему синтаксису:

имя_пакета_версия_архитектура.deb

В качестве примера можно вспомнить пакет Яндекс диска, который мы уже загружали ранее `wget http://repo.yandex.ru/yandex-disk/yandex-disk_latest_amd64.deb`

С помощью утилиты `file` можно увидеть, что это Debian binary package:

```
sa@astra:~$ file yandex-disk_latest_amd64.deb
yandex-disk_latest_amd64.deb: Debian binary package (format 2.0)
```

Подробную информацию, включая зависимости, можно получить с помощью команды `apt info`:

```
sa@astra:~$ dpkg --info yandex-disk_latest_amd64.deb
new Debian package, version 2.0.
size 4796514 bytes: control archive=1740 bytes.
  51 байт(a),      1 строк      conffiles
  540 байт(a),    15 строк      control
```

```
729 байт(a), 10 строк md5sums
1663 байт(a), 66 строк * postinst      #!/bin/sh
371 байт(a), 24 строк * postrm       #!/bin/sh
41 байт(a), 3 строк * prerm          #!/bin/sh
Package: yandex-disk
Version: 0.1.6.1080
...
```

Специальные типы deb-пакетов

Кроме обычных deb-пакетов существуют метапакеты и виртуальные пакеты.

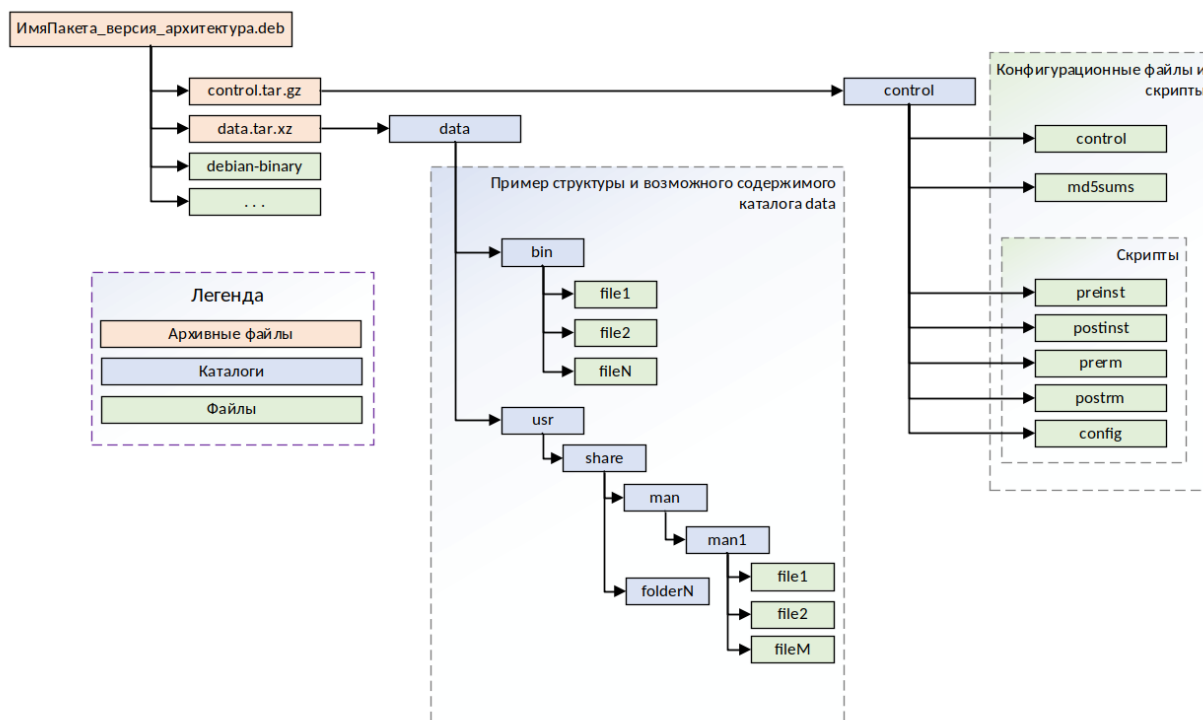
Метапакеты — это пустые пакеты, которые только описывают зависимости. Они облегчают установку совместной группы программ, предварительно выбранных сопровождающим метапакета. Команда `apt install <метапакет>` автоматически установит все эти программы, используя зависимости метапакета. Пример такого пакета libreoffice.

Виртуальный пакет — это средство идентификации (своего рода группировки) реальных пакетов на основе общих логических критериев (например, предоставляемых услуг или совместимости со стандартной программой или уже существующим пакетом). Таких пакетов физически не существует. Для обозначения виртуального пакета используется поле Provides файла control, входящего в пакет.

Внутреннее устройство deb-пакета

Содержимое deb-пакета включает:

- Архив `control.tar.gz` содержит скрипты, которые используются при установке/удалении пакета, а также другие служебные файлы.
- Архив `data.tar.gz` содержит файлы программы для установки в систему. Их расположение в архиве соответствует положению в целевой системе. Например, файл `usr/share/doc/package/copyright` в deb-архиве после установки будет помещен в `/usr/share/doc/package/copyright`.
- Файл `debian-binary` содержит версию программы.
- Другие файлы, например, электронную цифровую подпись.



Внутреннее устройство Deb-пакета

Командой `apt download acl` можно скачать текущую актуальную версию пакета `acl` и посмотреть его содержимое. Ниже представлен скриншот содержимого файла `control` в текстовом редакторе `kate`:

```

1 Package: acl
2 Version: 2.2.53-4
3 Architecture: amd64
4 Maintainer: Guillem Jover <guillem@debian.org>
5 Installed-Size: 206
6 Depends: libacl1 (= 2.2.53-4), libattr1 (>= 1:2.4.46-8), libc6 (>= 2.14)
7 Section: utils
8 Priority: optional
9 Multi-Arch: foreign
10 Homepage: https://savannah.nongnu.org/projects/acl/
11 Description: access control list - utilities
12 This package contains the getfacl and setfacl utilities needed for
13 manipulating access control lists. It also contains the chacl IRIX
14 compatible utility.
15

```

Содержимое файла `control` включает в себя следующие основные параметры:

- **Package** (обязательный) — имя пакета, под которым он будет зарегистрирован в системе.
- **Source** — имя и версия src-пакета, из которого собран данный deb-пакет (если есть таковой).
- **Version** — номер версии программы (пакета).

• **Section** (рекомендуемый) — имя раздела и секции, к которому принадлежит пакет, для его корректного размещения в репозитории в формате раздел/секция. Существуют 3 стандартных раздела: main, contrib и non-free. В каждом из этих разделов существует порядка 60 секций, например, admin, utils, devel, doc, base, net, mail и другие.

• **Priority** (рекомендуемый) - уровень важности пакета. Может принимать одно из 5 значений:

1. **Required** — пакет требуется в обязательном порядке для корректной работы любой Linux системы.

2. **Important** — пакет не влияет на работоспособность, но ожидается в любой Linux системе, например, man-db.

3. **Standard** — пакет входит в стандартный набор ПО для конкретного дистрибутива. Для разных дистрибутивов набор пакетов с уровнем важности Standard, как правило, разный.

4. **Optional** — пакеты, которые могут быть потенциально установлены в систему с данным дистрибутивом. Большинство пакетов относятся именно к этому уровню.

5. **Extra** — пакеты, которые конфликтуют с другими пакетами. Такие пакеты не устанавливаются по умолчанию пакетными менеджерами, но в случае необходимости такие пакеты может установить администратор, понимая при этом все связанные с этим риски.

• **Architecture (обязательный)** — архитектура, для которой данный пакет собран.

• **Depends** — перечень зависимостей между пакетами. Он включает в себя несколько видов зависимостей (могут быть использованы сразу несколько из них, каждая в новой строке):

1. **Depends** — перечень пакетов, перечисленных через запятую, от которых зависит данный пакет и без которых он не сможет нормально функционировать. Могут использоваться операторы сравнения для конкретизации необходимой версии пакета (<<, <=, =, >=, >>), например, **lib6 (>= 2.15)**, а также логическое «И» - запятая и логическое «ИЛИ» - вертикальная черта «|».

2. **Pre-Depends** — более строгая, чем depends, эти пакеты должны быть установлены еще до запуска скрипта установки данного пакета.

3. **Recommends** — перечень пакетов, которые значительно расширят функциональность, предлагаемую пакетом, но не являются обязательными для его работы.

4. **Suggests** — перечень пакетов, которые улучшают функциональность, но имеют второстепенное значение, как правило, отдельных утилит, или это может быть набор документации.

5. **Enhances** — перечень пакетов, работу которых может улучшить данный пакет, своего рода противоположность suggests.

6. **Conflicts** — перечень пакетов, которые не могут быть установлены одновременно с этим пакетом. Самой распространённой причиной является то, что оба пакета включают

файлы с одинаковыми именами, предоставляют один и тот же сервис на одном и том же порту протокола управления передачей TCP.

7. **Breaks** — схож с **conflicts**, но тут перечислены пакеты, работа которых (или их определенных версий) будет нарушена при установке данного пакета.

8. **Provides** — имя виртуального пакета.

- **Installed-Size** — размер пакета, поле установки в систему.

- **Maintainer (обязательный)** — имя и координаты мейнтейнера, создавшего пакет.

- **Description (обязательный)** — подробное описание пакета, может представлять собой несколько абзацев текста.

Можно вывести информацию об установленном пакете `acl` в системе командой `apt show acl` и убедиться, что выводимая информация совпадает с содержимым файла `control`:

```
sa@astra:~$ apt show acl
Package: acl
Version: 2.2.53-4+b1
Priority: optional
...
```

К необязательным файлам пакета относятся файл `md5sums` и скрипты для установки/удаления пакета. Файл `md5sums` содержит md5 хеши для всех файлов, кроме файлов, находящихся в архиве `control.tar.xz`. Данный файл необязателен для `deb`-пакета, однако программы верификации пакетов считают пакеты, не содержащие этот файл, ошибочными. Может использоваться некоторыми программами администрирования системы для верификации изменений в файловой системе.

Скрипты пакета могут вызываться при установке/удалении. Они расширяют возможности мейнтейнера по установке и настройке приложения. Например, этими скриптами могут создаваться/удаляться каталоги, в которых программа будет хранить свои временные файлы, может производиться добавление/удаление программы в меню вашего оконного менеджера, первичное конфигурирование программы и так далее.

Виды скриптов:

- **preinst** — скрипт, вызывающийся перед распаковкой пакета в файловую систему. Обычно при помощи этого скрипта пользователю задаются вопросы, позволяющие сконфигурировать программу прежде, чем начнется установка. Если скрипт завершится с кодом ошибки, то пакет не будет установлен в систему.

- **postinst** — скрипт, вызывающийся после распаковки пакета. При помощи этого скрипта производится конфигурирование уже установленной программы. Конфигурирование происходит с использованием информации, полученной на предыдущем этапе от пользователя. Задавать пользователю вопросы из скрипта `postinst` не принято.

- **prerm** — скрипт, вызываемый перед удалением пакета. Может задать пользователю некоторые вопросы, связанные с удалением. Например, вопрос о том, что делать с файлами журналов. Если скрипт завершится с кодом ошибки, пакет не будет удален.

- **postrm** — скрипт, вызываемый после удаления файлов пакета.

- **config** — скрипт, вызываемый на стадии конфигурирования и переконфигурирования пакета. Это дополнительный скрипт, который используется системой *Debconf*. Вызывается перед скриптом *postinst*, используется для того, чтобы задать пользователю вопросы в системе диалогов *debconf*.

Репозитории Astra Linux

Интернет-репозитории Astra Linux предназначены для упрощения процедуры обновления ОС и при наличии доступа в Интернет позволяют выполнять установку и обновление пакетов с помощью стандартных установщиков пакетов.

Для версии ALSE x.7 доступны 4 репозитория:

- Основной репозиторий (main)
- Оперативные обновления основного репозитория (update)
- Базовый репозиторий (base, включает main и update)
- Расширенный репозиторий (extended)

Рассмотрим назначение, состав и параметры этих репозиториях.

Основной репозиторий (репозиторий установочного диска, main):

- Содержит основной состав продукта, реализующий все функциональные возможности и функции безопасности.

- Проходит сертификационные испытания, соответствует сертифицированному изделию.

- Поставляется на установочном диске и в виде индивидуальной ссылки в личном кабинете пользователя.

Оперативные обновления основного репозитория (update):

- Содержит кумулятивные изменения основного репозитория (main), соответствующие выпуску оперативного обновления.

- Подлежит сертификации.

- Размещается в публичном доступе при выпуске бюллетеня безопасности.

Базовый репозиторий (base):

- Содержит все пакеты main и update.

- Пакеты со средствами разработки.

- Пакеты, необходимые для сборки репозитория main.

- Не подлежит сертификации (если рабочие места требуют аттестации, то во избежание установки лишних пакетов не используйте base вместо main+update).

- Поставляется в виде индивидуальной ссылки в личном кабинете пользователя.

Расширенный репозиторий (extended):

- Содержит дополнительное ПО, которое может функционировать в среде Astra Linux.
- Не подлежит сертификации и не дорабатывается для реализации функций безопасности. Пакеты из этого репозитория могут быть несовместимы с последними оперативными обновлениями основного/базового репозитория.

- Поставляется в виде индивидуальной ссылки в личном кабинете пользователя.

Примечание

Подробнее о составе, назначении и о работе с интернет-репозиториями Astra Linux, вы можете прочитать в статьях на wiki.astralinux.ru.

Создание собственных репозитория

Собственный репозиторий может понадобиться для уменьшения сетевого трафика, при массовом обновлении компьютеров или в случае, если по тем или иным причинам, интернет-канал не доступен на машине.

Создать собственный репозиторий (и управлять им) можно как полностью вручную (создать всю необходимую иерархию каталогов и метаданных), так и воспользовавшись специальным приложением **reprepro**, которое само создаст репозиторий согласно предоставленному конфигурационному файлу.

Практические задания

Задание 1.

Изучить утилиту dpkg и deb пакет.

1. Скачайте пакет Яндекс диска.
2. Проверьте его информацию.
3. Установите пакет.
4. Просмотрите список установленных пакетов.
5. Удалите пакет и все его конфигурации.
6. Получите список пакетов, установленных в системе, и отфильтруйте с помощью утилиты grep.
7. Получите детализированную информацию об установленном пакете sudo.
8. Выведите список файлов установленного пакета coreutils.
9. Найдите, какому пакету принадлежит указанный файл /bin/chacl.

Задание 2.

Изучить менеджер пакетов apt.

1. Настройте репозитории apt в файле source.list.
2. Обновите кэш apt.
3. Посмотрите список доступных пакетов для обновления командой.
4. Для поиска пакетов по входящим в него файлам необходимо воспользоваться утилитой apt-file.
5. Обновите кэш утилиты apt-file.
6. Воспользуйтесь этой утилитой, чтобы найти пакет, в состав которого входит утилита pstree.

Задание 3.

1. Установите VS code,
2. Установите расширение отладки VS code rogalmic.bash-debug.
3. Откройте в VS Code домашнюю папку со скриптами из лабораторной №6.
4. Настройте параметры запуска launch.json.
5. Откройте любой из файлов скрипта и поставьте точку останова на 5 строке (Клавиша <F9>).
6. Запустите отладку <F5> скрипта через расширение bash debug.

Задание 4.

Изучить менеджер пакетов aptitude.

1. Установите с помощью менеджера пакетов aptitude утилиты alien.
2. Удалите с помощью менеджера пакетов aptitude утилиты alien.

Задание 5.

Изучить менеджер пакетов Synaptic.

1. Найдите через поиск утилиту htop.
2. Установите с помощью менеджера пакетов Synaptic утилиту htop.
3. Удалите с помощью менеджера пакетов Synaptic утилиту htop.

4. Отключите репозиторий https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-extended/

Вопросы

1. Какое расширение у библиотек Astra Linux?
2. Какое расширение у инсталляционных пакетов Astra Linux?
3. Какой утилитой создается deb пакет?
4. Какие существуют репозитории в ALSE?
5. В каком файле прописываются репозитории?
6. Какой командой посмотреть информацию о пакете?
7. Какой менеджер пакетов устанавливает пакет и зависимые пакеты?
8. Какой менеджер пакетов имеет графический интерфейс?
9. Какая команда позволяет создать свой репозиторий?