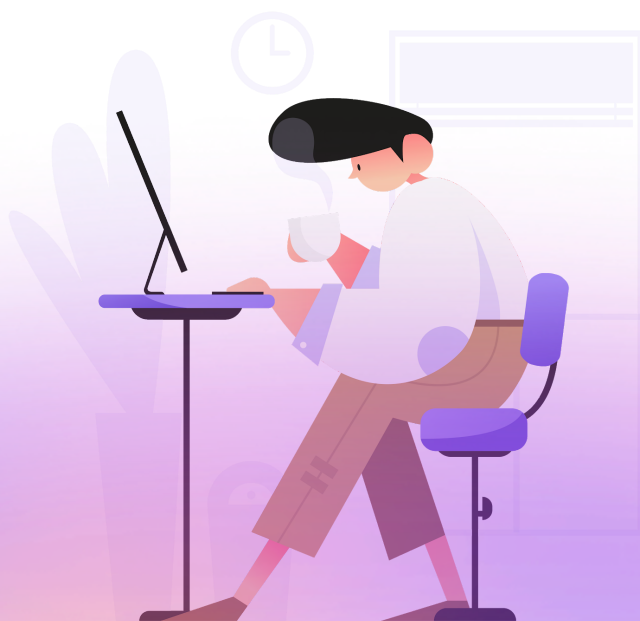


Общая версия Linux. Уровень 1

# Управление пакетами и репозиториями. Основы сетевой безопасности



# На этом уроке

1. Научимся подключать репозитории с программным обеспечением вручную, редактируя файл `/etc/apt/source.list` и используя утилиту `apt`.
2. Разберёмся, как устанавливать, обновлять и удалять пакеты.
3. Научимся использовать утилиту `dpkg` для управления пакетами, а также попробуем собрать файл из исходного кода.
4. Изучим основы сетевой безопасности на примере `netfilter` (`iptables`).

## Оглавление

[Глоссарий](#)

[Репозитории и управление репозиториями](#)

[Подключение репозитория](#)

[Добавление репозитория через редактирование файла `/etc/apt/source.list`](#)

[Добавление репозитория, используя команду `apt-add-repository`](#)

[Управление пакетами](#)

[Управление пакетами через утилиту `apt`](#)

[Управление пакетами через утилиту `dpkg`](#)

[Управление пакетами через утилиту `snap`](#)

[Основы сетевой безопасности и журналирование событий операционной системы](#)

[Практическое задание](#)

[Дополнительные материалы](#)

[Используемые источники](#)

## Глоссарий

**Репозиторий** — место, где хранятся и поддерживаются какие-либо данные. Чаще всего данные в репозитории хранятся в виде файлов, доступных для дальнейшего распространения по сети.

**Пакет** — под пакетами в Linux подразумевается программное обеспечение, которое можно установить, то есть набор файлов, объединённых для выполнения определённого функционала. Пакеты как правило хранятся в репозиториях.

**PPA** (сокр. от англ. Personal Packages Archive) — персональный архив пакетов. В отличие от других репозиториях Ubuntu, PPA-репозиторий содержит версии только какой-то одной программы.

[Apt](#) — программа для установки, обновления и удаления программных пакетов в операционных системах Debian и основанных на них. В Apt есть коровья суперсила.

[Dpkg](#) — это пакетный менеджер для Debian-систем. Он может устанавливать, удалять и создавать пакеты, но, в отличие от других систем управления пакетами, не может автоматически загружать и устанавливать пакеты или их зависимости.

[Snap](#) — это пакет, который, помимо готовой сборки самого приложения, включает в себя все необходимые зависимости и может работать почти в любом дистрибутиве Linux.

[Порт](#) — целое неотрицательное число, записываемое в заголовках протоколов транспортного уровня модели OSI (TCP, UDP, SCTP, DCCP). Используется для идентификации процесса-получателя, т.е. помимо адреса хоста необходимо идентифицировать приложения - это происходит с помощью номера порта. Номера портов уникальны в пределах одного хоста.

## Репозитории и управление репозиториями

Обновления и программное обеспечение в Linux-системах устанавливаются из репозитория.

[Репозиторий](#) — это хранилище пакетов, то есть файлов и библиотек, которые мы можем установить в ОС. Репозиторий может быть размещён локально, может находиться на носителе (флешке, DVD-диске), но чаще всего он размещён в интернете. Условно репозитории можно разделить на три группы:

1. **Стандартные репозитории** — это репозитории, поддерживаемые разработчиками операционных систем. Включают в себя стабильные версии программного обеспечения. Зачастую эти версии отстают на несколько шагов от последних версий пакетов.
2. **Дополнительные репозитории** — репозитории, поддерживаемые разработчиками программного обеспечения. Включают в себя последние стабильные версии ПО. Зачастую узкоспециализированы под конкретный пакет и библиотеки, необходимые для этого пакета.
3. **Неофициальные репозитории** — репозитории, созданные сообществом или одним человеком. Могут содержать в себе как последние стабильные, так и тестируемые версии программного обеспечения.

Программное обеспечение в Ubuntu делится на четыре вида по типу лицензирования и уровню поддержки:

1. **Main** — свободное ПО, официально поддерживаемое компанией Canonical.
2. **Restricted** — проприетарное ПО (в основном драйверы устройств), официально поддерживаемое компанией Canonical.
3. **Universe** — свободное ПО, официально не поддерживаемое компанией Canonical, но поддерживаемое сообществом пользователей.

4. **Multiverse** — проприетарное ПО, не поддерживаемое компанией Canonical.

Официальные репозитории Ubuntu делятся на следующие типы:

1. **\$release** — пакеты на момент выхода релиза.
2. **\$release-security** — пакеты критических обновлений безопасности.
3. **\$release-updates** — пакеты обновления системы, то есть более поздние версии ПО, вышедшие уже после релиза.
4. **\$release-backports** — пакеты более новых версий ПО, которое доступно только в нестабильных версиях Ubuntu.
5. **partner** — репозиторий, содержащий ПО компаний-партнёров Canonical.

Информация о подключённых репозиториях в Ubuntu хранится в каталоге `/etc/apt/`, в файле **sources.list**. Репозитории защищают от подмены при помощи сверки цифровых подписей репозитория и клиента. В репозитории хранится закрытая часть ключа, у клиента — открытая часть ключа.

## Подключение репозитория

В Ubuntu репозитории можно подключить тремя способами: используя графический интерфейс, путём редактирования файла `/etc/apt/sources.list` и используя утилиту `apt`. Рассмотрим два последних варианта.

### Добавление репозитория через редактирование файла `/etc/apt/sources.list`

В текстовом редакторе открываем файл `/etc/apt/sources.list` и в конце файла вводим строку вида:

```
deb http://адрес_репозитория версия_дистрибутива ветки
```

Например, добавим репозиторий `nginx`, для этого создадим в каталоге `/etc/apt/sources.list.d/` файл `nginx.list` следующего содержания:

```
deb http://nginx.org/packages/ubuntu bionic nginx
```

Здесь `bionic` — это версия Ubuntu, а `nginx` — название ветки, содержащей пакеты, необходимые для установки `nginx`. Следующий шаг — это установка публичного ключа репозитория, для этого нужна команда **apt-key**. Скачиваем при помощи `curl` наш ключ и передаём через pipe утилите **apt-key**:

```
curl -fsSL https://nginx.org/keys/nginx_signing.key | sudo apt-key add -
```

И последний шаг — это обновление информации о пакетах **sudo apt update** и установка пакета **sudo apt install nginx -y**.

Создание файлов с репозиториями и размещение их в каталоге `/etc/source.list.d/` удобно тем, что мы можем обновить информацию о пакетах из конкретного репозитория `apt update repo_name`.

## Добавление репозитория, используя команду `apt-add-repository`

Эта команда автоматически создаёт записи в файле `/etc/apt/sources.list` или создаёт файл репозитория в каталоге `/etc/apt/sources.list.d/`, а также может удалять информацию о репозиториях. Чаще всего эта утилита используется для добавления PPA-репозитория.

PPA-репозитории находятся [на сайте Launchpad.net](http://ppa.launchpad.net), который поддерживается компанией Canonical. Утилита автоматически находит строку для записи в файл репозитория, скачивает и импортирует ключи. Рассмотрим добавление репозитория `nginx` с использованием PPA-репозитория:

```
apt-add-repository ppa:nginx/stable
```

Здесь утилите `apt-add-repository` мы говорим, что подключаем PPA-репозиторий, поддерживаемый группой `nginx`, и подключаем стабильную версию. Утилита автоматически создаст файл `/etc/apt/sources.list.d/nginx-ubuntu-stable-bionic.list` с содержимым, которое мы можем просмотреть при помощи команды `cat`:

```
cat /etc/apt/sources.list.d/nginx-ubuntu-stable-bionic.list  
  
"deb http://ppa.launchpad.net/nginx/stable/ubuntu bionic main"
```

Утилита импортирует ключи и обновит список пакетов.

# Управление пакетами

В Ubuntu управление пакетами осуществляется тремя способами: с использованием утилиты `apt`, `dpkg` или `snap`.

## Управление пакетами через утилиту `apt`

Это пакетный менеджер, который включает в себя набор утилит для управления пакетами. Он позволяет осуществлять поиск, установку и удаление пакетов, обновлять операционную систему, подключать репозитории. Подключение репозитория с использованием `apt` было рассмотрено в предыдущей части. Рассмотрим параметры утилиты `apt` для управления пакетами:

- `apt search package_name` — поиск пакета;
- `apt show package_name` — посмотреть информацию о пакете;
- `apt install package_name -y` — установить пакет;

- `apt install package_name1 package_name2 -y` — установить два пакета;
- `apt remove package_name` — удалить пакет, при этом сохраняются файлы с настройками;
- `apt purge package_name` — полностью удалить пакет, включая конфигурационные файлы;
- `apt upgrade` — обновить все установленные пакеты;
- `apt update` — обновить информацию о пакетах в репозиториях, указанных в настройках.

## Управление пакетами через утилиту **dpkg**

**Dpkg** — пакетный менеджер в Debian-подобных системах. Главное отличие от утилиты **apt** состоит в том, что **dpkg** работает только с локальными пакетами, он не умеет искать и устанавливать пакеты с репозиториях. Основные параметры утилиты **dpkg**:

- `dpkg -l` — просмотр списка пакетов;
- `dpkg -i package_name` — установить пакет или группу пакетов;
- `dpkg -r package_name` — удалить пакет или группу пакетов.

## Управление пакетами через утилиту **snap**

**Snap** — это пакет, который, помимо готовой сборки самого приложения, включает в себя все необходимые зависимости и может работать почти в любом дистрибутиве Linux. В какой-то степени можно считать, что пакеты, установленные при помощи **snap**, — альтернатива самостоятельной сборке пакета. Пакет, установленный через **snap**, содержит все необходимые зависимости и может работать в любом окружении Linux. Snap состоит из двух частей: демона **snapd** и клиента для управления пакетами **snap**. Установка **snapd** производится командой `apt install snapd -y`. Параметры **snap**:

- `snap search package_name` — поиск пакета;
- `snap install package_name` — установка пакета;
- `snap refresh package_name` — обновление пакета;
- `snap remove package_name` — удаление пакета;
- `snap list` — просмотр установленных пакетов.

# Основы сетевой безопасности и журналирование событий операционной системы

Информационная безопасность и построение защищённых серверов — отдельное направление в мире информационных технологий. Мы рассмотрим базовые аспекты работы **netfilter** и **iptables** — инструмента управления **netfilter**. Они понадобятся нам для понимания, например, построения

сетевого взаимодействия между операционной системой и системой контейнерной виртуализации Docker.

Netfilter — встроенный в ядро Linux сетевой фильтр. Для управления netfilter служит утилита iptables. Основа iptables — таблицы, в которых содержатся цепочки с правилами. Основная работа происходит с двумя таблицами:

1. Таблица **filter**. В этой таблице происходит фильтрация входящего и исходящего трафика, а также транзитный трафик.
2. Таблица **nat**. Необходима для трансляции адресов и портов.

В каждой таблице есть **цепочки** — наборы правил, согласно которым обрабатывается тот или иной трафик. У каждой цепочки есть **политики по умолчанию**, согласно этим политикам трафик обрабатывается, если не попадает ни под одно из правил.

Существует 5 типов стандартных цепочек, встроенных в систему:

1. **PREROUTING** — для изначальной обработки входящих пакетов.
2. **INPUT** — для входящих пакетов, адресованных непосредственно локальному процессу (клиенту или серверу).
3. **FORWARD** — для входящих пакетов, перенаправленных на выход. Заметьте, что перенаправляемые пакеты проходят сначала цепь PREROUTING, затем FORWARD и POSTROUTING.
4. **OUTPUT** — для пакетов, генерируемых локальными процессами.
5. **POSTROUTING** — для окончательной обработки исходящих пакетов.

Политики для цепочки устанавливаются следующим образом:

```
iptables -P имя_цепочки Действие
```

Например, `iptables -P INPUT ACCEPT` разрешит весь входящий трафик, действует для таблицы filter. Если нам необходимо установить политику в конкретной таблице, то через параметр `-t` нужно передать имя таблицы, например: `iptables-t nat -P INPUT ACCEPT`. В Ubuntu ACCEPT — политика по умолчанию для всех таблиц и цепочек. Действует правило: разрешено всё, за исключением того, что запрещено.

Правила в цепочках создаются следующим образом:

```
iptables -A имя_цепочки -p протокол --dport порт -j действие
```

Например: `iptables -A INPUT -p tcp --dport 80 -j ACCEPT`. В данном правиле мы добавляем в iptables, в таблицу filter правило, разрешающее подключения по протоколу TCP на порт 80 нашего сервера.

Также можно разрешить или ограничить подключения для определённых источников, используя параметр `-s` — source, например: `iptables -A INPUT -p tcp -s 192.168.0.100 --dport 80 -j DROP`. Здесь мы запретим все подключения по протоколу TCP, исходящие соединения от хоста с IP-адресом 192.168.0.100 на порт 80 нашего сервера.

Используя таблицу nat, мы можем организовать проброс портов внутрь нашей сети. Для этого необходимо сделать следующее:

1. Включить переадресацию трафика на уровне ядра, изменив значение параметра `net.ipv4.ip_forward` на 1: `sudo sysctl -w net.ipv4.ip_forward=1`.
2. Убедиться, что политика по умолчанию для транзитного трафика: ACCEPT: `iptables -A FORWARD -j ACCEPT`.
3. Настроить модификацию адреса назначения (Destination) DNAT и модификацию адреса отправителя (Source) SNAT. Действие DNAT работает с цепочкой PREROUTING таблицы nat. В этой цепочке будет изменяться адрес назначения пакета, чтобы он достиг нужной нам цели. Действие SNAT работает с цепочкой POSTROUTING таблицы nat. В этой цепочке произойдёт модификация адреса источника и замена его на адрес маршрутизатора. Например:
  - `iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination 10.0.0.2` пробросит все входящие соединения на порт 80 нашего сервера, внутрь сети, на хост 10.0.0.2.
  - `iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080` организует проброс портов входящего трафика с 80-го порта нашего сервера на порт 8080 нашего сервера.
  - `iptables -t nat -A POSTROUTING -o eth1 -p tcp --dport 80 -d 10.0.0.2 -j SNAT --to-source 10.0.0.1` модифицирует IP-адрес источника из локальной сети на адрес маршрутизатора локальной сети.

Посмотреть текущие правила iptables можно следующим образом:

1. `iptables -t имя_таблицы -L` покажет правила, установленные в таблице. Если не указать имя таблицы, будут показаны правила из цепочки filter. Например, `iptables -L` покажет правила в цепочках таблицы filter, а `iptables -t nat -L` покажет правила в цепочках таблицы nat.
2. `iptables-save` сделает дамп правил и выведет его на экран. Также команду `iptables-save` можно использовать для сохранения правил в файл: `iptables-save > iptables.bk`. Восстановить правила можно, используя команду `iptables-restore: iptables-restore<iptables.bk`.



# Практическое задание

1. Подключить репозиторий с nginx любым удобным способом, установить nginx и потом удалить nginx, используя утилиту dpkg.
2. Установить пакет на свой выбор, используя snap.
3. Настроить iptables: разрешить подключения только на 22-й и 80-й порты.
4. Настроить проброс портов локально с порта 80 на порт 8080.

## Дополнительные материалы

[Основы управления пакетами Ubuntu](#)

[Установка snap-пакетов](#)

[Основы iptables](#)

## Используемые источники

[Управление пакетами в Ubuntu](#)

[snap](#)

[iptables](#)