

Введение в Linux

Юрий Литвинов

y.litvinov@spbu.ru

1. Что такое Linux

Linux — это семейство Unix-подобных операционных систем, объединённых более-менее общим ядром, набором стандартов и соглашений. Ядро Linux имеет открытый исходный код, лежит в репозиториях, поддерживаемых сообществом, но имеет зеркало на GitHub (<https://github.com/torvalds/linux> (дата обращения: 24.02.2024)). Вообще, Linux строится вокруг идеи свободного программного обеспечения (хотя Linux, Free Software Foundation и GNU — это разные вещи, их не надо путать), поэтому сообщество очень щепетильно относится к защите от vendor lock-in, поэтому на GitHub и подобные хостинги целиком никогда не переедет.

Linux имеет огромное сообщество разработчиков, управляемых по иерархической структуре, в корне которой находится Линус Торвальдс — собственно, первый разработчик ядра, который начинал писать Linux как студенческую работу на основе идей учебной ОС Minix от Эндрю Танненбаума (который сам по себе очень известный товарищ). Открытость и вовлечённость кучи людей в процесс имеет свои плюсы и свои минусы:

- всё, о чём вы можете подумать, скорее всего, кто-то уже сделал;
- пользователи имеют доступ к исходному коду, замечают проблемы, предлагают правки, поэтому популярное ПО (тем более — ядро), скорее всего, очень стабильно и «вылизано» до блеска;
- никто никому ничего не должен, поэтому не очень популярное ПО может быть откровенно плохим;
 - более того, часто разрабатывается красноглазыми школьниками, которые без идей, как программировать, но попасть в какой-нибудь дистрибутив Linux для них мечта — в ядро их код не берут, но прикладные программы вполне.
- нет единых стандартов, архитектуры, процесса, не только для прикладного ПО, но и для важных частей ОС.

Применяется Linux сейчас в основном как серверная ОС, и в этом плане она с большим отрывом лидирует (разве что другие UNIX-подобные системы типа FreeBSD, которые неопытный пользователь никогда не отличит от Linux, могут составить ей конкуренцию). Также Linux, в силу открытости и модульности, применяется для встроенных устройств

и разного специализированного оборудования, где нет требований реального времени¹. Linux также несколько удобнее для профессиональной разработки, особенно для низкоуровневого программирования, под него более зрелый и удобный инструментарий. Однако и Enterprise-, и веб-, облачная и т.п. разработка тоже часто ведётся на Linux, отчасти потому, что это достаточно удобно, отчасти чтобы быть ближе к окружению, в котором разработанная система будет работать.

Почему не весь мир работает на Linux — у Linux не очень хороши дела для поддержки конечных пользователей. Современные дистрибутивы очень дружелюбны (в былые времена иногда приходилось программировать на Си для повседневного пользования компьютером, что не очень), но гораздо хуже обстоят дела с поддержкой аппаратного обеспечения и с играми. С аппаратным обеспечением дела обычно поправимы — если производитель не выпускает официально ПО для Linux, это делают энтузиасты, и если что-то не заработало сразу, можно некоторое время пострадать (возможно, попрограммировав на Си), и заработает (в отличие от Windows, где, как правило, если что-то не работает, то всё). Из того, с чем наверняка придётся столкнуться — некоторая ненадёжность автоматической настройки дисплеев, так что заставить линуксовый ноутбук правильно выводить на проектор может быть неким вызовом (программировать не придётся, но копаться в конфигах в трёх разных местах — возможно). Заставить работать три монитора с разным разрешением от двух видеокарт разных вендоров так, чтобы выполнялось адекватное масштабирование пользовательского интерфейса — автору так и не удалось.

С играми похуже, сообщество не считает их чем-то полезным, и там часто используются специфичные для Windows технологии, такие как DirectX. Однако в последнее время игровые движки делаются кроссплатформенными, так что игры на них под Linux в целом работают, хотя иногда странно. Но есть довольно много игр, которые не работают, и, видимо, никогда не будут.

Linux распространяется в виде дистрибутивов. Само ядро не имеет никакого пользовательского интерфейса, даже консольного, и его можно понимать как скомпилированную библиотеку, которая всем управляет и предоставляет программный интерфейс для прикладных программ и системных утилит. Ядро модульное, есть штуки, которые так и называются — «модули», их можно воспринимать как динамически загружаемые библиотеки, которые можно подгружать и выгружать прямо в процессе работы системы, и они обычно нужны для поддержки того или иного оборудования. Однако чтобы пользователю работать с системой, нужна куча прикладных программ — во-первых, шелл (shell), интерфейс командной строки, во-вторых, утилиты типа копирования файлов, в-третьих, графическая оболочка, в-четвёртых, пакетный менеджер, отвечающий за установку, обновление и удаление прикладных программ. Всё это вместе, плюс начальная конфигурация системы, плюс политика обновления ПО и т.п. — это и есть дистрибутив, то есть готовая к использованию система, которую можно поставить к себе на компьютер и начать ей пользоваться. Дистрибутивов сотни, даже известных и широкораспространённых десятки. Есть три-четыре популярных российских, которые на волне импортозамещения и ухода Microsoft стремительно занимают освободившуюся нишу в корпоративной и государственной сферах, и заодно и у обычных пользователей, хотя им приходится отчаянно конкурировать с западными. Для корпоративных нужд требуется техподдержка, что с западными дистрибутивами мо-

¹ Системы жёсткого реального времени должны давать гарантию, что такая-то процедура будет выполнена за такое-то физическое время, например, за 20 миллисекунд. Linux не может давать такой гарантии.

жет быть трудно, для домашнего использования техподдержка не нужна, поэтому вполне разумный выбор (хотя ходят слухи, что некоторые программы могут потихоньку гадить пользователям из России).

1.1. Чем Linux хорош для программистов

Linux популярен в программистском сообществе по целому ряду причин. Наверное, основная — бесплатность. Linux можно ставить на виртуальные машины в каких угодно количествах, не платя за лицензию, а поскольку сейчас повсеместно используется виртуализация и контейнеризация, это может быть критично. В правильно настроенном сервере *каждый* делающий полезную работу процесс работает в собственном контейнере (или на целой виртуальной машине), так что на лицензиях на Windows и ПО для него даже крупная компания могла легко разориться. Однако, как обычно, есть нюанс — бывают и даже вполне распространены платные дистрибутивы. Бесплатность открытого ПО вовсе не значит, что на нём нельзя зарабатывать — покупая платный дистрибутив, пользователь покупает отсутствие проблем и техподдержку, если проблемы всё-таки возникнут, плюс, иногда, проприетарные специальные программы и сервисы, которых нет в открытом доступе. Например, Астра Линукс распространяется именно по такой модели — есть очень устаревшая бесплатная версия и свежая платная для корпоративных клиентов, с «правильными» гарантиями безопасности, включая гарантию защиты от «protestware»².

Вторая причина — конфигурируемость. Есть готовые дистрибутивы, включающие в себя всё, необходимое для работы. Есть миниатюрные системы, которые, кроме ядра, имеют программ на 4-5 мегабайт (!), что идеально для работы в контейнерах, потому что контейнеризованному приложению ядро не нужно. Да и всего, вместе с ядром, вполне можно собрать работоспособный дистрибутив размером чуть больше сотни мегабайт, что позитивно для встроенных устройств и специализированных контроллеров. Есть дистрибутивы, которые можно целиком собрать из исходников под конкретную архитектуру, есть специализированные системы сборки, которые позволяют собрать ядро и нужные модули и положить поверх только нужные программы и конфигурацию.

Третья причина — это простое человеческое удобство, если привыкнуть. Наличие пакетного менеджера и централизованного репозитория с программами позволяет не искать ничего в интернете, устанавливать всё одной командой и обновлять вообще автоматически. При этом продвинутая система управления зависимостями поставит всё, что нужно, так, чтобы оно не мешало уже установленному программному обеспечению и библиотеки/вспомогательные программы не дублировались для сотни прикладных программ, как это принято в Windows³. В Linux традиционно удобная и прилично выглядящая консоль, поскольку по историческим причинам пользователи проводят в консоли очень много вре-

² «Protestware» — программное обеспечение, которое, помимо полезных функций, пытается донести до пользователя позицию автора по какому-либо политическому или социальному вопросу, вплоть до попытки форматирования диска пользователей, которых автор считает в чём-либо виновными. В мире открытых систем огромной сложности весьма опасно (поскольку может быть «притянута» зависимостями), и зачастую крайне незачинно. Не делайте так, какими бы благими ни были ваши намерения.

³ Тут, опять-таки, есть нюансы — зависимости хорошо работают внутри дистрибутива, но если нужного ПО нет в дистрибутиве, есть, но старой версии, или просто в пакете ошибка, поставить его на Linux-систему может быть нетривиально. Поэтому Linux сейчас тоже движется к тому, что в Windows/.NET называется «xcopy deployment» — всё, что нужно для работы, приложения носят с собой. На Linux-системах с этим может очень помогать контейнеризация.

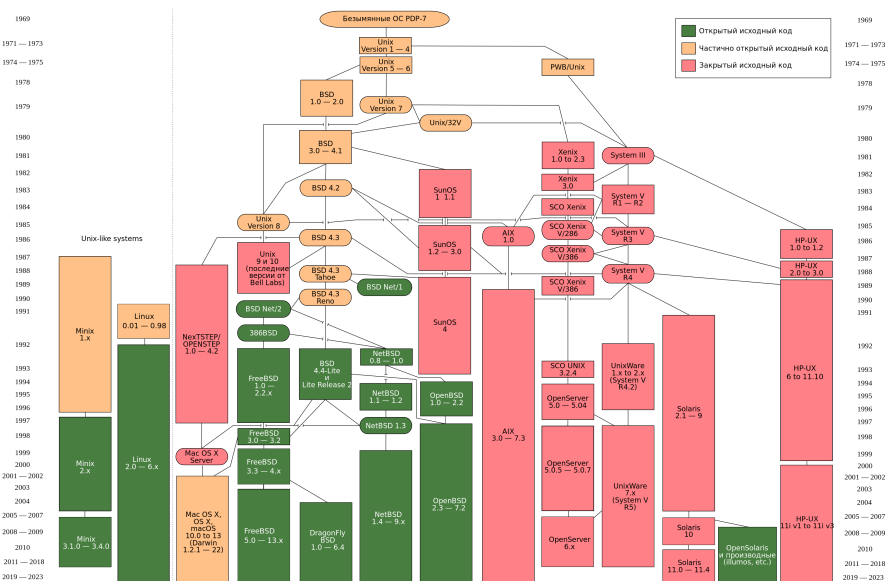
мени, там всё вылизано до блеска. Наверное, требуется некоторое время, чтобы привыкнуть и выучить горячие клавиши выбранного вами шелла (да-да, в Linux консоль не встроена в систему, есть штуки три только популярных консольных оболочек). Зато потом открывать консоль на Windows уже не захочется.

Кроме всего этого, есть целый ряд инструментов, которые не работают на Windows (и тем более Mac OS) вообще. Например, отладчик памяти Valgrind (в Visual Studio есть аналог, но Valgrind умеет гораздо больше Visual Studio), эмулятор аппаратного обеспечения QEMU (её вроде как можно запустить под Windows, но я бы не стал даже пытаться), эмулятор процессора Gem5 (там возможность запуска не под Linux разработчиками даже не рассматривается) и т.д. и т.п. Два последних инструмента могут быть критичны для ряда направлений разработки, включая встроенные системы и компиляторы.

1.2. История Linux

Как известно, Linux — это написанная с нуля как студенческая работа ОС (где-то в 1990 году), позаимствовавшая ряд идей из учебной ОС Minix, которая в свою очередь использовала ряд идей, стандартов и соглашений Unix. Unix пошла ещё от DEC-овских мэйнфреймов, появилась аж в начале 1970-х (когда создание единой ОС для серийных ЭВМ стало насущной необходимостью, до этого ОС писалась под каждую конкретную машину). От Unix произошла ОС BSD разных сортов (Berkeley Software Distribution, хотя на самом деле BSD — это более общая штука, чем ОС). От BSD — Mac OS (так что Mac OS более Unix, чем Linux). Кстати, Windows произошла от MS-DOS (хотя, конечно, была полностью переписана с тех времён), а MS-DOS — это клон системы CP/M 1974-го года, которая вроде как вообще никакого отношения к Unix не имела.

Вот родословная Unix и Unix-подобных операционных систем:



1.3. Отличия от Windows

Тут обсудим отличия Linux от Windows с точки зрения пользователя. Вообще, это совершенно разные операционные системы, с разной философией, архитектурой и, конечно, пользовательским опытом, однако общего у них тоже довольно много (что неудивительно, похожие задачи всегда влекут похожие решения). Вот с чем пользователь Windows, который впервые пересаживается на Linux, обязательно столкнётся.

- **Пакетный менеджер.** В Linux приложения не скачиваются с непонятных сайтов, как принято в Windows, а ставятся из централизованных репозиториях с помощью установочных пакетов и специальной программы — того самого пакетного менеджера. Причём, если в Windows программы обычно ставятся в Program Files и иногда имеют свою подпапку в папке пользователя, то в Linux приложение оказывается «размазанным» по файловой системе по определённым правилам, о которых чуть дальше. Поэтому установка, удаление, обновление и т.п. делается через пакетный менеджер, просто удалять программу с диска нельзя, и не принято встраивать в программы функциональность автообновления (например, VS Code обновляет себя под Windows, но обновляется силами пакетного менеджера под Linux). Пакеты могут зависеть друг от друга, поэтому прикладные программы, как правило, имеют в разы меньшие размеры, чем в Windows — всё нужное уже, скорее всего, установлено в системе, и если нет, то поставится автоматически и может быть переиспользовано ещё кем-то. Пакетная система дистрибуции программного обеспечения уже наверняка знакома — в Windows есть свой пакетный менеджер Chocolatey (но поскольку он не встроен в систему, о нём мало кто знает), есть Microsoft Store (но там своя атмосфера), ещё есть Android и iOS, где репозиторий централизованный и контролируется производителем. В Linux репозиторий обычно привязан к дистрибутиву и один, но никто не мешает добавить сколько угодно дополнительных, и тут это валидная модель распространения программы — например, VS Code предоставляет репозиторий, где лежат пакеты самого VS Code и зависимостей. Скачивать программы с непонятных сайтов тоже можно, тем более что непонятные сайты обычно предлагают удобный способ установки под Linux (например, .NET ставится именно так, скриптом dotnet-install.sh с сайта Microsoft), но делается это только в случае, если в репозитории дистрибутива нужного пакета нет.
- **Файловая система различает регистры в именах файлов и директорий, так что header.h и Header.h — совершенно разные файлы.** Поэтому на всякий случай всё обычно именуется со строчной через подчёркивание или дефис. Бывают, конечно, исключения (в дистрибутиве, на котором верстался этот конспект, например, есть директория «Рабочий стол»). У файлов нет атрибута «скрытый», поэтому по соглашению скрытые файлы именуются с точки в начале. Кстати, все имена файлов — в UTF-8.
- **В файловой системе нет понятия «диск», дерево файлов и директорий имеет один корень, директорию «/».** Абсолютно всё, включая файлы на разных физических дисках, флешках и сетевых шарах, лежит где-то в поддиректориях «/». Есть понятие «монтирование» — «прививание» внешней файловой системы внутрь какой-то из директорий «/». Например, флешка может быть доступна по пути «/media/user/флешка/».

Монтирование можно выполнять вручную, но в подавляющем большинстве случаев оно выполняется само.

- Обратите внимание, слешы в Linux всегда прямые. Windows умеет и так и так, но предпочитает обратные, Linux считает обратный слеш началом escape-последовательности.
- Оконный менеджер не является частью операционной системы. В Windows управление окошками — часть функциональности ядра, поэтому поменять их внешний вид и, тем более, функциональность, весьма сложно. В Linux есть как отдельные приложения низкоуровневые рисовалки, которые умеют рисовать собственно окна и обрабатывать события пользовательского ввода, так называемые «окноводы», которые управляют размерами и размещением окон, и «среды рабочего стола», которые добавляют всякие кнопки, панельки, виджеты, эффекты и т.п. И это более-менее ортогональные вещи, и по каждому пункту есть выбор, иногда весьма значительный. Поскольку всё это весьма конфигурируемо, каждый дистрибутив выглядит по-своему, да ещё и можно настроить под себя практически всё, вплоть до полной смены оконной подсистемы (что не так просто, но возможно).
- В Windows можно было набрать что-нибудь вроде «`gun.exe`», и если `gun.exe` есть в текущей папке, он запустится. Linux скажет, что файл не найден, и начинающие пользователи могут быть шокированы, потому что вот же он лежит. Дело в том, что Linux не ищет по умолчанию запускаемые файлы в текущей директории, а сразу идёт смотреть пути из PATH — по соображениям безопасности, чтобы нельзя было подложить что-то странное в директорию с программой. Чтобы запустить что-то из текущей директории, добавьте явную на неё ссылку: «`./gun.exe`». Ещё у файлов (и директорий) есть атрибут «исполнимый», если атрибут не выставлен, запустить программу/скрипт не выйдет даже с точкой. Windows понимала, что файл исполнимый, по расширению, Linux тут на расширения не смотрит. Выставить атрибут «исполнимый» можно командой `chmod +x имя-файла`.
- В Windows большая часть системных настроек была в реестре, который по сути большая иерархическая база данных, которую без `regedit.exe` даже открыть-то не получится, причём и с `regedit.exe` там весьма сложно что-то содержательно редактировать и надо либо знать, что делаешь, либо пользоваться программами для настройки. В Linux есть правило, что все настройки должны быть человекочитаемыми и человекомо-редактируемыми. Поэтому обычно для конфигурации всего, даже самых дебрей ОС, используются простые текстовые файлы. При этом если опций много и файл большой, применяется соглашение с разделением такого конфига на несколько файлов и выкладыванием их в директорию `<имя конфига>.d`, где файлы перенумерованы в порядке их применения. Так что конфиги не только человекочитаемы, но ещё и обозримы и понятны (как правило).
- Очень важное различие — папки в Linux называются директориями.

2. Что бывает в мире Linux

2.1. Дистрибутивы

Дистрибутив — это готовое к установке ядро Linux, набор программ и пакетная экосистема, включающая в себя репозиторий (репозитории) пакетов, политику их включения в репозиторий и обновления, документацию и техническую поддержку. В принципе, Linux можно собрать на коленке самостоятельно, без всяких дистрибутивов, но это достаточно большая работа, чтобы компании типа Canonical или Red Hat могли неплохо зарабатывать на распространении готовых дистрибутивов. Почему — в типичном Linux может быть больше десятка тысяч системных и прикладных программ, библиотек и технологий, и задача дистрибутива — сделать так, чтобы они все вместе хорошо работали, что на самом деле весьма нетривиально.

Вот некоторые распространённые дистрибутивы — так-то их сотни.

- Ubuntu — Linux для десктопных применений вместо Windows «по умолчанию». Большое сообщество, поддержка коммерческой компании, обновляется раз в полгода (в апреле и октябре).
- Mint — дистрибутив на базе Ubuntu, предполагалось, что будет почаще обновляться, имеет свой менеджер рабочего стола, кому-то может быть приятнее, чем Ubuntu, но дело вкуса.
- Fedora — старый проверенный дистрибутив от Red Hat и IBM (то есть за ним, как и за Ubuntu, коммерческие компании, которые его активно развивают), не любит проприетарное ПО (поэтому может быть не так удобен «вместо Windows»), но целостный, удобный и регулярно обновляемый. Достойный выбор.
- Debian — ещё более старый и проверенный дистрибутив (первый выпуск в 1993 году, когда Linux ещё только появился), на его основе делалось большинство более модных и современных дистрибутивов (Ubuntu, Mint, даже SteamOS). Обновляется пореже.
- Arch — дистрибутив на основе своего пакетного менеджера, в основе которого концепция частых (раз в месяц) обновлений. Для опытных пользователей.
- Gentoo — дистрибутив, где всё надо собирать из исходников. Сейчас это на самом деле уже неправда, но философия дистрибутива была в том, чтобы вообще не распространять бинарные пакеты, а поставлять исходники и иметь на машине пользователя всё необходимое для сборки. Тогда пользователь может собрать ПО конкретно под своё оборудование конкретно так, как ему нужно. На современных машинах сборка под конкретную архитектуру позволяет добиться ускорения работы примерно на 1-2 процента, но ждать результатов компиляции несколько часов, так что сообщество сошлось на том, что того не стоит. Но любителям всё контролировать и настраивать под себя зайдёт (*только* для опытных пользователей, не ставьте его как свой первый Linux!).
- Альт, РОСА, Астра, РЕД ОС — самые известные российские дистрибутивы, как правило, платные с бесплатной версией. Альт и РОСА базируются на Mandriva (причём

Альт отделился очень давно), Астра — на Debian. Хороши тем, что сертифицированы для использования в том числе в государственных органах, и поэтому активно замещают Windows для рабочих нужд конечных пользователей. Бесплатные варианты не всегда хорошо поддерживаются, имеют довольно старые пакеты в репозиториях и местами содержат ошибки (TeX из репозитория на Альте у автора не заработал, какой-то файл положить забыли). Однако если хотите пережить отключение российского сегмента интернета, отзыв корневых сертификатов или прекращение обслуживания зоны .ru корневыми DNS-серверами, или просто не хотите, чтобы какой-нибудь драйвер принтера удалил вам все данные за то, что у вас ip-адрес из России, это вполне достойный выбор. Данный текст верстался на ROSA Fresh Desktop 2021.1 (хоть они и не платят за рекламу).

Вот картинка из википедии, которая перечисляет несколько больше дистрибутивов и показывает их взаимосвязь: https://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg (дата обращения: 25.02.2024). Обратите внимание, что это SVG, так что можно искать по названию дистрибутива.

2.2. Пакеты

Основой любого дистрибутива является пакетная система. Пакет — это способ распространения приложений и библиотек в Linux, состоит из бинарников (собранных под какую-то целевую архитектуру), конфигурационных файлов, документации, иногда исходников⁴. Также в пакете есть метainформация — как пакет называется, зачем он, версия (в формате semantic versioning — мажорная и минорная версии, релиз; между минорными версиями гарантируется бинарная совместимость), зависимости (другие пакеты, которые должны быть установлены перед тем, как ставить этот). Зависимости образуют дерево, зачастую весьма ветвистое, потому что пакет в Linux — это, как правило, не целое приложение, а что-то гораздо более мелкое, вплоть до конкретного конфигурационного файла. Ещё бывают метапакеты — пакеты, где есть *только* зависимости. Они нужны, чтобы сразу поставить пачку пакетов, нужных для решения какой-то общей задачи (например, сразу весь TeX). Общий вид типичного имени пакета — <название>-<версия>-<выпуск>.<архитектура> (где архитектура — это подо что собран пакет, бывает *noarch*, то есть не важно, подо что, везде будет работать, или указание конкретной архитектуры процессора — например, пакеты под *i686* под *arm* не запустятся).

Пакеты хранятся в разных форматах. Это всегда сжатая каким-то архиватором директория, так что скачанный пакет можно открыть и посмотреть обычным архиватором. Детали в том, каким именно алгоритмом выполняется компрессия и как представляется метainформация (и что в ней вообще есть — например, «рекомендованные» пакеты). Популярных форматов всего два — *.deb* (из Debian, на базе которого делается много других дистрибутивов, например, Ubuntu) и *.rpm* (Red Hat Package Manager, Fedora и её многочисленные производные). В общем, все дистрибутивы могут работать с пакетами всех форматов, но

⁴ Большинство программ под Linux распространяется по традиции под лицензией GPL, которая требует распространения исходников — однако не требует обязательно поставлять их прямо с бинарниками, поэтому часто пакет делится на собранную программу (или библиотеку или что угодно), которую обычно ставят пользователи, и отдельный пакет с исходниками, для тех, кто готов засучить рукава и что-то править. Иногда пакет просто приводит ссылку на репозиторий и на том успокаивается.

репозиторий дистрибутива всегда строится на чём-то одном, поэтому пакеты в другом формате будут несколько тяжелее в установке. Это мягко говоря — если у нужного пакета много зависимостей, можно замучиться их разрешать.

Для работы с пакетами есть утилиты, которые по уже готовому пакету могут его корректно поставить — это не просто распаковать, а разложить по директориям, отметить в локальной базе пакетов, что такой-то установлен. Для .deb это `dpkg`, для .rpm — так и называется, `rpm`. У них есть и графические оболочки. Однако пользователь с ними напрямую обычно не взаимодействует, их вызывают пакетные менеджеры — более высокоуровневые штуки, которые умеют укачать нужный пакет и его зависимости из репозитория, искать нужный пакет локально и в репозиториях дистрибутива и т.п. Вот самые популярные пакетные менеджеры:

- `apt` (Advanced Packaging Tool) — используется в Debian и дистрибутивах на его базе: Ubuntu, Mint, Альт и т.п.
- `yum` (Yellowdog Updater, Modified) — используется в старой Fedora и дистрибутивах на её основе.
- `dnf` (Dandified YUM) — используется в новой Fedora и дистрибутивах на её основе, например, РОСА.

Функциональность пакетных менеджеров примерно одинакова (поскольку они одну задачу решают), но команды могут немного отличаться.

Также в Linux-сообществе набирает популярность подход с установкой приложения целиком со всеми зависимостями (к чему давно пришли в Windows), иногда даже с контейнеризацией. Примеры менеджеров пакетов, реализующих такой подход — Flatpak, Snappy, AppImage. Они позволяют создавать пакеты, не зависящие от дистрибутива, что в традиционном подходе невозможно, и тем избавиться от *dependency hell* и необходимости для одного приложения поддерживать десяток пакетов для популярных дистрибутивов. Недостаток — значительно большие требования к дисковому пространству и интернет-каналу, накладные расходы (хоть и не большие) по скорости работы на контейнеризацию.

2.3. Среды рабочего стола

Среды рабочего стола виднее всего пользователю, поэтому при выборе дистрибутива внезапно наиболее важный выбор, поскольку если UI неудобный, продуктивность работы даже с отличным прикладным ПО будет не очень. Вообще, напомним, что в Linux графическая подсистема не является частью ядра, а обычно разделена на графический сервер и клиенты, которые, посылая запросы графическому серверу, рисуют окна и всё необходимое. Графических сервера популярных два — исторически первый X Server, более новый (ну как новый, 2008 года), но всё ещё не очень распространённый и не очень поддерживаемый Wayland. Wayland считается более прогрессивным, но более болезненным для использования в ситуациях типа «пошарить окно в Teams» или даже «подключиться к проектору».

Однако если всё хорошо, пользователь даже не знает, какой у него графический сервер. Видит он среду рабочего стола. Есть две известные среды:

- GNOME и его производные (MATE, Cinnamon) — используется в Ubuntu, Mint и много где ещё. Довольно легковесный и быстрый, но не очень конфигурируемый и без впечатляющих эффектов. Реализован на графической библиотеке GTK+.

- Plasma (ранее известный как KDE) — используется по умолчанию в РОСА, в вариантах известных дистрибутивов — например, KUbuntu (это буквально «Ubuntu на KDE»). Очень конфигурируемый и сложный в настройке, красивый, небыстрый (из-за кучи отключаемых графических эффектов), близок по UX оконной системе Windows.

Ещё бывает XFCE, совсем уж легковесная и нефункциональная оконная система, которую, однако, вполне можно поставить на какой-нибудь забытый сервер, чтобы не только из консоли работать, или на слабенький ноутбук (или слабенький одноплатник, который взрослую среду не потянет). Тоже на GTK+, так что в каком-то смысле младший брат GNOME.

Важно понимать, что среда рабочего стола образует некую экосистему приложений. Есть приложения на GTK+, которые органично смотрятся в GNOME, но не очень хорошо интегрируются в Plasma и кажутся там противоестественными. Есть приложения на Qt, которые везде смотрятся хорошо, но, возможно, у автора предвзятое мнение после нескольких лет опыта разработки на этой библиотеке. Они, тем не менее, могут не выполнять принятые в GNOME соглашения, и потому не очень органично вписываться в привычный пользователям интерфейс. Даже более важно то, что оконные приложения базируются на оконных библиотеках, которые базируются на низкоуровневых графических и вспомогательных библиотеках, так что:

- поставить любое графическое приложение под любую среду рабочего стола вполне возможно, известные дистрибутивы имеют в репозиториях и Plasma (KDE), и GNOME, и всё, что нужно для их работы, но
- одно несчастное приложение, устанавливаемое на «неродную» среду, притянет в зависимости большую часть среды, для которой оно было написано, что, скорее всего, ничего не сломает, но может быть совершенно негуманно по отношению к месту на диске и скорости скачивания.

3. Работа с консолью

Теперь очень кратко рассмотрим основные знания про работу с консолью Linux, поскольку с консолью тут придётся взаимодействовать гораздо чаще, чем в Windows, и надо уметь использовать её эффективно.

3.1. Основные команды

Вот, пожалуй, самые частоиспользуемые команды в консоли Linux.

- `cd` — переход в директорию, так же, как в Windows; абсолютные пути начинаются с `/`, относительные без `/` в начале, могут содержать «`..`» или «`..`», которые имеют тот же смысл, что и в Windows (текущая директория и родительская директория соответственно — например, `cd ..` ничего не делает, `cd dir/..` тоже).

- `ls` — показать содержимое директории. Без параметров — текущей, с параметром — по заданному пути. Имеет много ключей, например, `-a`, который заставляет `ls` показывать и скрытые файлы тоже, или `-l`, который показывает файлы с правами доступа, размером, датой последней модификации и т.п.
- `mkdir` — создать директорию, как в Windows. С ключом `-p` создаёт весь переданный в качестве параметра путь, иначе — только поддиректорию в текущей.
- `touch` — обновить время последнего обращения к файлу или создать пустой файл. Наверное, самый простой способ создать файл в Linux. Ещё полезен в скриптах сборки, чтобы заставить систему сборки пересобрать файл, даже если не было изменений с последней сборки.
- `echo` — распечатать что-то в консоль, как в Windows.
- `cat` — вывести содержимое файла на консоль прямо как есть.
- `cp` — скопировать файл или директорию, так же, как в Windows.
- `mv` — переместить файл или директорию со всем содержимым.
- `rm` — удалить файл или директорию. Удаляет только один файл или директорию, если она пустая, что крайне неудобно, поэтому обычно используется с ключом `-rf` (Recursive, Force) — рекурсивно удалить всё поддерево, не спрашивать подтверждения. Если не знаете, что делаете, обычно плохая идея. Например, `rm -rf *.txt` удалит все файлы `.txt` в текущей директории, а `rm -rf * .txt` удалит всё вообще. Есть ещё старая шутка про `rm -rf /`, но от обычного пользователя эту команду вряд ли дадут исполнить.
- `ln` — создать ссылку на файл или директорию. Символьные ссылки создаются с ключом `-s`, жёсткие ссылки ключа не требуют, но создаются только на файл.
- `sudo` — запустить от имени суперпользователя. Например, `sudo rm -rf /` так убьёт вам систему. `sudo` приходится пользоваться довольно часто, потому что обычные пользователи в Linux не могут ничего, что влияет на всю систему (Linux сразу создавалась многопользовательской и сетевой, так что неудивительно). Даже установка пакета — всегда с `sudo`. Хорошая новость в том, что если пользователь состоит в группе `sudoers` или `wheel` (а по умолчанию при установке системы в большинстве дистрибутивов первый пользователь туда добавляется), никаких паролей или чего-то вводить не надо, `sudo` просто молча исполняет команду. Тем не менее, `sudo` честно переключает пользователя, то есть в команде, исполняющейся из `sudo`, например, пользовательская домашняя директория будет `/root`, а не `/home/имя-пользователя`.