Модуль PROD-1. Практический Linux

Linux — это open source-операционная система, относящаяся к семейству Unix-систем (вместе с Mac OS и Android).

Linux ОС была разработана в 1991 году финно-американским программистом **Линусом Торвальдсом** и с тех пор только набирает популярность. На сегодняшний день подавляющее большинство серьёзных вычислительных платформ и серверов работает именно на Linux. Именно поэтому любому уважающему себя IT-специалисту рекомендуется освоить Linux.

Структура файловой системы Linux

Директория в Linux — это список соответствий имени файла и узла, где располагается вся информация о файле.

Давайте посмотрим на список директорий, которые по умолчанию имеются в ОС Linux:

- `/` — корневой каталог файловой системы. Директории в нём (см. ниже) могут располагаться на различных физических дисках и разделах. В Linux про диски C: , D: и т.д. можно забыть.
- `/boot` — инструментарий запуска ОС, в повседневной работе не требуется. Ядро системы.
- `/etc` — директория, где в текстовых конфигурационных файлах хранятся системные настройки, чаще всего в формате "Parameter Value". Также — основные настройки некоторых приложений. Аналог Windows Registry.
- `/root` — домашний каталог пользователя root.
- `/bin` — здесь хранятся исполняемые файлы для утилит системы.
- `/lib` — здесь хранятся системные библиотеки.
- `/usr` — сюда устанавливаются пользовательские приложения. Аналог Program Files в Windows.
- `/usr/lib` — здесь обычно хранятся библиотеки установленных приложений.

- ➔ `/usr/bin` — здесь обычно хранятся исполняемые файлы для запуска приложений.
- ➔ `/dev` — в *nix используется метафора «всё есть файл». Поэтому каждому устройству ставится в соответствие псевдофайл, который служит для обмена данными с операционной системой и складывается в эту директорию.
- ➔ `/proc` — аналогичная схема используется и для процессов системы. Им также соответствуют псевдофайлы в этом каталоге, как это делается для устройств.
- ➔ `/tmp` — директория временных файлов. Аналог Windows\Temp.
- ➔ `/home` — директория, где хранится пользовательская информация. Для каждого пользователя, заведённого в систему, создаётся своя поддиректория по его login, например /home/myuser1. Это аналог Users\myuser1 в Windows. В этих каталогах хранятся файлы пользователя и его индивидуальные настройки.

→ Для взаимодействия с системой Linux используются так называемые bash-команды.

Bash — это оболочка командной строки в системах Linux. Иными словами, это интерфейс, через который мы можем взаимодействовать с системой и выполнять нужные нам задачи.

Конвенции команд Linux

1. Первое слово команды — это сама команда (что мы просим сделать).
2. За командой следуют опции, или параметры. Они начинаются со знака минус (-). Регистр имеет значение!
3. Параметры обозначаются одной буквой. Регистр имеет значение!
4. Параметры могут объединяться: -la = -l -a
5. Параметры могут следовать в любом порядке.
6. Списки разделяются пробельными символами.
7. `man` — подробная справка по любой команде.

Типы команд

→ Все команды в Linux можно поделить на два типа — базовые и те, которые управляют заранее установленными программами.

1. Базовые команды присутствуют в большинстве дистрибутивов Linux-систем: `ls`, `pwd`, `mkdir`, `rm` и так далее.
2. Команды, которые управляют заранее установленными программами: `python`, `docker`, `hadoop` и так далее.

Базовые команды — это такие команды, которые являются составляющей частью командной оболочки, доступны из любой папки практически в любой системе Linux. Ниже представлен неполный перечень таких команд.

Команда	Описание
`cd data`	Позволяет перейти в директорию `data`
`cd ..`	Перейти на директорию выше
`cd ~`	Перейти в домашнюю директорию: `/home/username/`
`cd -`	Перейти в предыдущую папку
`pwd`	Показать текущую директорию
`mkdir data`	Создать каталог с именем `data`
`touch text.txt`	Создать файл с именем `text.txt`
`rm -r data`	Удалить каталог с именем `data`
`rm text.txt`	Удалить файл с именем `text.txt`

Типы файлов в Linux

`-`	просто файл
`d`	директория
`-`	link
`s`	symbolic link
`b или c`	устройство
`/proc`	псевдофайл (в директории `/proc`)

Относительные и абсолютные имена в Linux

`/` в начале	абсолютный путь
`.`	текущая директория
`~`	домашняя (home) директория
`..`	директория на уровень выше

Создание и удаление

`mkdir`	создать директорию
`rm -r`	удалить файл или директорию (рекурсивно — включая поддиректории)
`touch`	создать пустой файл

Копирование и перемещение

`cp`	копирование файлов (откуда и куда)
`mv`	перемещение файлов
`-r`	опция для рекурсивного копирования или перемещения

Посмотреть начало и конец файла

`head`	посмотреть первые строки (байты)
`tail`	посмотреть последние строки (байты)

Тип файла и перекодировка

`file`	посмотреть тип файла (включая кодировку для текстовых файлов)
`iconv`	перекодировать файл (например, из кодировки Windows в UTF-8)
`tr`	удалить или заменить одиночный символ в файле

Просмотр файла

`more`	посмотреть файл постранично (с поиском), хорошо работает для очень больших файлов
`cat`	посмотреть файл полностью, удобно для небольших файлов

Количество строк

`wc -l`	посмотреть количество строк в файле (файлах)

Фильтрация строк

`grep`	фильтрация строк в файле

Посимвольный просмотр

`od -c`	просмотр файла в виде последовательности символов (можно задать внешний вид — восьмеричный, шестнадцатеричный и так далее)

Сжатие и архивирование

`gzip`	работа с GZIP-файлами
`unzip`	работа с ZIP-файлами
`unrar`	работа с RAR-файлами
`tar`	популярный в среде Linux архиватор (собирает файлы в архивный .tar)

Поиск файлов

`find`	найти файлы (по имени, размеру, дате изменения и так далее)

Почему задача “тормозит”?

Причина может быть связана с:

- ➔ процессором;
- ➔ памятью;
- ➔ вводом-выводом.

Процесс — это программа на этапе исполнения. Для удобства отслеживания процессов есть множество привязанных к ним данных, которые мы можем видеть и с которыми мы затем можем работать. Каждый процесс кодируется уникальным числом, которое называется PID, также есть меняющиеся параметры: к примеру, процессы имеют разные приоритеты, от которых зависит, сколько ресурсов они потребляют, и система для большей эффективности постоянно перераспределяет эти приоритеты, поэтому их значения динамически меняются.

Основные команды для контроля процессов

`top`	список выполняющихся процессов
`ps`	список всех процессов

`vmstat`	информация о ресурсах
`kill`	остановить процесс

Основные команды при работе с пользователями

`whoami`	показать логин текущего пользователя
`id`	показать полный набор атрибутов
`su`	“стать другим пользователем”
`sudo`	выполнить команду “от имени другого пользователя”

В Linux права доступа делятся на три категории:

- ➔ владелец (user);
- ➔ группа (group);
- ➔ все остальные (others).

Виды доступа (что мы можем разрешить делать):

- ➔ чтение (`r`);
- ➔ запись (`w`);
- ➔ исполнение (`x`). Для файла это означает запуск, если файл представляет из себя скрипт или программу. Для директорий же этот вид доступа означает, что вы можете заходить в директорию, просматривать и изменять содержимое файлов, которые в ней находятся.

Основные команды для управления доступом

`ls`	просмотр прав
`chmod`	изменение прав
`chown`	смена владельца

`chgrp`	смена группы

Выполнение цепочки команд

Командная оболочка Linux (например, bash) получает данные (команды) на вход и отправляет информацию на выход через потоки данных.

> **Поток данных** — абстракция для чтения и записи данных.

В Linux выделяют три основных потока:

- **stdin** — стандартный поток ввода. Читает данные (команды), которые вводит пользователь через терминал.
- **stdout** — стандартный поток вывода. Выводит в терминал (но можно перенаправить например в файл) данные, полученные в результате выполнения команд.
- **stderr** — стандартный поток вывода ошибок. Как правило, совпадает с stdout, но может быть перенаправлен, например в файл. Выводит ошибки в результате выполнения команд.

Используя эти знания, мы можем настроить так называемый конвейер — **pipeline**.

Основные моменты, которые важно здесь знать:

- почти все команды умеют читать из stdin;
- команда | команда - stdout команды = stdin следующей команды;
- (команда ; команда) — группировка команд.

Основные команды apt

Поиск пакетов для установки:

- `apt update` — обновить список доступных для установки пакетов;

- ➔ `apt-cache search package_name` — найти название пакета для установки (искать в доступных пакетах по строке package_name).

Установка пакетов:

- ➔ `apt-get install first_package second_package` — установить последние версии first_package и second_package;
- ➔ `apt-get install package_name1.1` — установить пакет package_name версии 1.1.

Обновить пакет(-ы) до последней версии:

- ➔ `apt upgrade` — обновить все пакеты до последних версий;
- ➔ `apt-get upgrade package_name` — обновить пакет package_name.

Удаление пакетов:

- ➔ `apt-get remove package_name` — удалить пакет package_name.

Bash-программирование

Переменные в Bash:

- ➔ содержат строки;
- ➔ оператор присваивания `=`;
- ➔ использование значений `${имя переменной}`.

Формат скрипта:

- ➔ всё после `#` — комментарий;
- ➔ перенос на другую строку — `\`;
- ➔ magic вначале — `#!`;
- ➔ право на выполнение — `x`.

Печать в stdout:

- ➔ `echo` — вывести в stdout строку (параметр);
- ➔ `echo -n` — вывести без перевода строки.

Цикл for:

```
for переменная in список
do
   оператор
done
```

Частое использование — сделать что-то с файлами из списка.

Условный оператор if:

```
if условие
 then
     оператор
 else
    оператор
 if
```

Часто комбинируется с проверкой условия.

Проверка условия test:

- ➔ `строка == строка` — истинно, если строки равны;
- ➔ `-f` — истинно, если файл существует;
- ➔ `-z` — истинно, если строка имеет нулевую длину.

Досрочное завершение:

- ➔ `exit` — выйти (с заданным кодом завершения).

Пауза в выполнении:

- ➔ `sleep` — приостановить выполнение программы на заданное количество секунд.

Результат выполнения:

- ➔ `exit code`;
- ➔ `stdout`;
- ➔ использование `stdout`.