

ЗАЧЕМ НАМ НУЖЕН LINUX

- *Hadoop* устанавливается на *Linux*;
- *Linux*-серверы очень популярны, и у инженера данных практически нет шансов избежать работы с ними;
- *Linux* и *Unix* появились давно и за счёт *Open source* точно выкристаллизовали фундамент своей работы. Знать и понимать его очень полезно.

ЧТО ТАКОЕ LINUX

- *Open source* операционная система;
- выросла из *Unix*;
- автор — *Linus Torvalds* (v 1.0 1994);
- дистрибутивы (*Ubuntu*).

В большинстве дистрибутивов *Linux* по умолчанию используется **командная оболочка Bash**.

ПОЧЕМУ КОМАНДНАЯ СТРОКА?

- На серверах обычно нет графического интерфейса (*GUI*).
- Командная строка (*CLI*) позволяет автоматизировать повторяющиеся действия.
- *Linux* очень хорошо интегрируется с *Jupyter Notebooks*.

ИНТЕГРАЦИЯ С JUPYTER NOTEBOOKS

- `!` в *IPython*;
- *bash cell magic* `%%bash`;
- окно терминала.

КОНВЕНЦИИ КОМАНД LINUX

- параметры (опции) начинаются с `-`;
- параметр — одна буква (регистр имеет значение);
- параметры могут объединяться `-la` = `-l -a`;
- параметры могут следовать в любом порядке;
- списки разделяются пробельными символами;
- `man` — подробная справка по любой команде.
-

ОБМЕН ДАННЫМИ С BASH В JUPYTER NOTEBOOK

В *Notebook* можно вызвать любую *shell*-команду. Результат (*stdout*) выполнения команды можно сохранить в переменной:

```
file_list = !ls
```

```
print(file_list)  
['01_rof.ipynb', '02_erview.ipynb', '03_uniq.ipynb']
```

Значение переменной можно использовать в *bash*-команде:

```
text_files = "/tmp/*.txt"
```

```
!ls -l {text_files}
```

ЗАЧЕМ НАМ ЭТО

- Все действия в системе *Linux* персонализированы (процессы имеют владельцев).
- Основной ресурс (файлы и директории) имеет владельцев.
- Часть проблем в кластере возникает из-за некорректной персонализации.

UID И LOGIN

- **пользователь** — это уникальная комбинация *UID* и *login*;
- *UID* — число;
- *login* — строка;
- отображается *login*;
- используется *UID*.

ГРУППЫ ПОЛЬЗОВАТЕЛЕЙ

- пользователи в *Linux* объединяются в группы;
- пользователь входит в одну или несколько групп;
- главная (*primary*) группа;

- уникальная комбинация *GID* и имени группы;
- используется при работе с файлами.

ПОЛЬЗОВАТЕЛИ В КЛАСТЕРЕ

- должны быть одинаковыми (*UID* и *login*);
- объединяются в группы (например, *Hadoop*);
- *HDFS* (файловая система *Hadoop*) использует *UID* и *login*.

ГДЕ ЖИВУТ И КАК НАЗНАЧАЮТСЯ

- информация о *UID* и *login* пользователя, а также о принадлежности к его первичной группе, хранятся в файле *"/etc/passwd"*;
- *ID* и имя группы, а также список пользователей, которые входят в эту группу, хранятся в файле *"/etc/group"*;
- *LDAP*, *Active Directory*;
- при входе пользователя в систему происходит поиск логина и назначение соответствующего ему *ID*;
- *ID* текущего пользователя можно сменить.

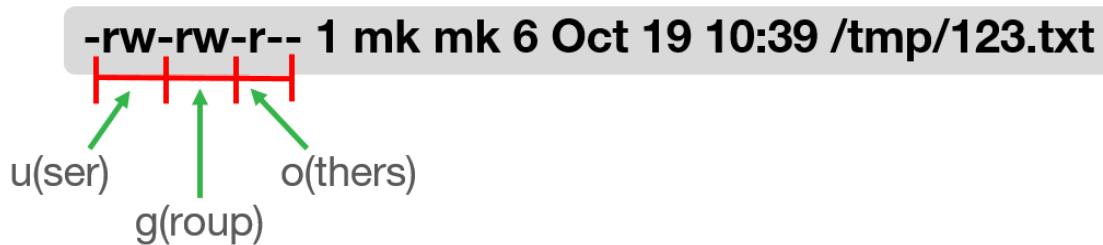
ОСНОВНЫЕ КОМАНДЫ

- `whoami` — показать логин текущего пользователя;
- `id` — показать полный набор атрибутов;
- `su` — «стать другим пользователем»;
- `sudo` — выполнить команду «от имени другого пользователя».

ПРАВА ДОСТУПА К ФАЙЛАМ

Рассмотрим пример:

`-rw-rw-r-- 1 mk mk 6 Oct 19 10:39 /tmp/123.txt`



u(ser) g(roup) o(thers)

«Категории» прав доступа в Linux:

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

«Виды» доступа:

- чтение (*r*);
 - запись (*w*);
 - выполнение (*x*);
- (*x*) для директорий = «доступ» к файлам директории

Директория в *Linux* — это список соответствий имени файла и узла, где располагается вся информация о файле, т. е. **право на чтение директории** — это право на чтение/запись/изменение этого каталога. **Право на выполнение директории** эмулируется возможностью или невозможностью доступа к самому файлу, который «живёт» не внутри каталога, а внутри файловой системы.

Права доступа в кластере строятся по этим же принципам.

ОСНОВНЫЕ КОМАНДЫ

- `ls` — просмотр прав;
- `chmod` — изменение прав;
- `chown` — смена владельца;
- `chgrp` — смена группы.

КОМАНДА `LS`

Назначение: показать список файлов. Список задаётся в качестве параметров (все списки в *Linux* разделяются *white space*: пробелы, табуляции и прочие «пробельные» символы).

Если параметры не указаны, показывается список файлов в текущей директории.

```
!ls
```

```
01_ds_prof.ipynb      07_file_mgmt.ipynb  de_star.png
```

Основные (часто используемые параметры) разберём ниже.

ДЛИННЫЙ ФОРМАТ `-l`

```
!ls -l
```

Каждая строка содержит информацию об одном файле. Поля в строке разделены пробелами.

Какая информация о файле выдаётся:

- права доступа к файлу;
- количество имён у файла;
- владелец файла;
- группа владельца файла;
- размер файла (в байтах, для директории — размер «файла с директорией»);
- дата последней модификации файла;
- имя файла.

Типы файлов в Linux:

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

Относительные и абсолютные имена в Linux:

- текущая директория (в *HDFS* — нет);
 - `/` в начале = абсолютный путь;
 - файловая система — дерево (если абстрагироваться от синонимов);
 - `.` = текущая директория;
 - `~` = «домашняя» (*home*) директория;
 - `..` = директория на уровень выше в дереве.
-

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) файла;
- `tail` — посмотреть последние строки (байты) файла;
- эффективно работают для очень больших файлов.

Тип файла и перекодировка:

- `file` — посмотреть тип файла (включая кодировку для текстовых файлов);
 - `iconv` — перекодировать файл (например, из *Windows*-кодировки в *utf-8*);
 - `tr` — удаление или замена одиночного символа в файле.
-

Постраничный просмотр:

- `more` — посмотреть файл постранично (с поиском);
- хорошо работает для очень больших 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 т. п.).

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*-файл).

Поиск файлов: `find` — найти файлы (по имени, размеру, дате изменения и т. п.).

Процесс — это программа на этапе исполнения. Каждый процесс имеет свой уникальный идентификатор *PID* (число) и много других атрибутов, например, приоритет, который изменяется динамически.

ОСНОВНЫЕ КОМАНДЫ

- `top` — список выполняющихся процессов;
- `ps` — список всех процессов;
- `vmstat` — информация о ресурсах;
- `kill` — остановить процесс.

Рассмотрим возможности командной оболочки *bash*, которые позволят избавиться от рутины, повторного набора текста и связанного с ним большого количества ошибок.

История выполнения

- все выполненные команды `bash` сохраняет в истории выполнения;
- можно повторить любую команду;
- можно модифицировать аргументы;
- пробел перед командой — нет записи в историю.

Использование истории

- стрелки (вверх-вниз);
- `history` — показать историю выполнения;
- `!номер` — повторить команду из истории.

Завершение слов

- `TAB` пытается «завершить» начатое слово;
- алгоритм завершения
 - команды;
 - файлы и директории;
- `ESC ?` — просмотр возможных завершений.

Перенаправление ввода-вывода

- `stdin, stdout, stderr` (дескрипторы 0, 1, 2);
- `> >>` — перенаправление в файл (режим `append`);
- последовательность перенаправлений.