

Утилита file – анализ начала содержимого файла. Типы файлов в GNU/Linux.

Category: file, GNU/Linux, утилиты
2025-02-01

Утилита file.

GNU/Linux не делает предположение о типе данных в **обычном файле**, есть специальная утилита, которая выполняет эту задачу – утилита file.

Для этого она анализирует начало содержимого файла и находит в нем специальные «сигналы», характерные для определенного типа – бинарного файла, текстового, изображения и других типов файлов.

```
# file script.sh
```

Ответ:



Команда file имеет множество различных ключей.

```
# file --help
```

Синтаксис: **file** [OPTION] [FILE]

Определим, что скрывается за символьной ссылкой?

```
# file /bin
```

Ответ:



```
# file -L /bin
```

Ответ:



В примере каталог /bin является символьной ссылкой на каталог, находящийся в другом месте, о чем нам говорит команда file без

ключей.

С помощью ключа `-L` мы можем узнать тип файла, на который ссылается файл-ссылка.

Типы файлов GNU/Linux.

В **GNU/Linux** понятие типа файла не связано с расширением файла, то есть несколькими символами после точки в конце имени файла, как это, например, сделано в **Windows**.

GNU/Linux не следит за расширениями файлов. Пользовательское приложение анализирует структуру данного файла, расширение ему также безразлично.

Таким образом, среди файловых атрибутов, хранящихся в операционной системе на базе ядра **GNU/Linux**, нет информации о типе данных в файле. Там есть информация о более существенном разделении, связанном с тем, что в **GNU/Linux** все объекты — это файлы. Все объекты весьма разнообразны. Поэтому тип файла в **GNU/Linux** — это скорее тип объекта, но не тип данных как, например, в **Windows**.

В операционной системе **GNU/Linux** существуют следующие типы файлов:

- `-` — обычные файлы,
- `d` — каталоги,
- `l` — символьные ссылки,
- `b` — блочные устройства,
- `c` — символьные устройства,
- `p` — каналы,
- `s` — сокеты.

Каждый тип имеет собственное обозначение одним символом. Знание этих символов пригодится в дальнейшем при изучении командной оболочки `bash`.



Обычные файлы (`-`).

Сюда относятся **все файлы с данными**, играющими роль ценной информации сами по себе. Операционной системе **GNU/Linux** все равно текстовый перед ней файл, исполняемый или картинка. В любом случае это будет обычный (regular) файл. Все они обозначаются знаком минус «`-`». Остальные типы файлов считаются специальными (special).

Каталоги (d) .

В **GNU/Linux** каталог представляет собой такой тип файла, данными которого является список имен других файлов и каталогов, вложенных в данный каталог. Напрямую, то есть через какой-либо редактор, пользователь не может редактировать данные файла-каталога. Редактированием занимается ядро операционной системы, получая, в том числе от пользователя, команды создания файла, удаления и прочие команды.

В **файле каталога** осуществляется связь между **именами файлов** (словесного обозначения для людей) и их **индексными дескрипторами** (истинным именем-числом, которым оперирует **GNU/Linux**).

В **GNU/Linux** один и тот же файл может существовать под разными именами и/или в разных каталогах: все имена будут связаны с одним и тем же индексным дескриптором (механизм **жестких ссылок**).

Также следует, что файлы всегда должны содержаться в каталогах, иначе станут недоступны, так как нигде не будет содержаться записи о них.

Символьные ссылки (l) .

Символьная ссылка – это файл, в данных которого содержится адрес другого файла по его имени, а не индексному дескриптору.

Выполнение символьной ссылки приводит к открытию файла, на который она указывает. Это аналог ярлыков в операционной системе **Windows**.

Если удалить исходный файл, то символьная ссылка продолжит существовать. Она по-прежнему будет указывать на файл, которого уже нет.

Символьные ссылки не содержат атрибутов файлов, на которые они указывают. У них есть собственные атрибуты: свое время создания, размер, права доступа.

Символьные (с) и блочные устройства (b) .

Файлы устройств предназначены для обращения к аппаратному обеспечению компьютера: дискам, принтерам, терминалам и так далее. Когда происходит обращение к файлу устройства, то ядро операционной системы передает запрос драйверу этого устройства.

К символьным устройствам обращение происходит последовательно, то есть символ за символом. Примером символьного устройства может служить **терминал**.

Считывать и записывать информацию на блочные устройства можно в произвольном порядке, причем блоками определенного размера. Пример: жесткий диск.

Сокеты (s) и каналы (p).

Чтобы понять, что такое **канал** и **сокет**, необходимо понимание процессов в операционной системе. И **каналы** и **сокеты** организуют их взаимодействие. Пользователь с данными типами файлов почти не сталкивается.

Ключевым отличием **канала** от **сокета** является то, что канал однонаправлен. Через **канал** один **процесс** всегда передает **данные** второму, но не наоборот. **Сокеты** позволяют передавать данные в разных направлениях, то есть осуществляют связь.

Также следует отметить, что **канал** представлен в структуре каталогов файлом, только если он именован. Когда возникают безымянные каналы, то они существуют только внутри ядра **GNU/Linux**.

Оригиналы источников информации.

1. younglinux.info «Типы файлов Linux».