Перенаправление ввода/вывода в Linux

Содержание

* [Введение](https://selectel.ru/blog/tutorials/linux-redirection/#intro)
* [Потоки](https://selectel.ru/blog/tutorials/linux-redirection/#streams)
* [Стандартный ввод](https://selectel.ru/blog/tutorials/linux-redirection/#standard-input)
* [Стандартный вывод](https://selectel.ru/blog/tutorials/linux-redirection/#stdout)
* [Стандартная ошибка](https://selectel.ru/blog/tutorials/linux-redirection/#standard-error)
* [Каналы](https://selectel.ru/blog/tutorials/linux-redirection/#channels)
* [Фильтры](https://selectel.ru/blog/tutorials/linux-redirection/#filters)
* [Заключение](https://selectel.ru/blog/tutorials/linux-redirection/#conclusion)

Введение

Стандартные потоки ввода и вывода в Linux являются одним из наиболее распространенных средств для обмена информацией процессов, а перенаправление **>**, **>>** и **|** является одной из самых популярных конструкций командного интерпретатора.

В данной статье мы ознакомимся с возможностями перенаправления потоков ввода/вывода, используемых при работе файлами и командами.

Требования

* Linux-система, например, Ubuntu 20.04

Потоки

Стандартный ввод при работе пользователя в терминале передается через клавиатуру.

Стандартный вывод и стандартная ошибка отображаются на дисплее терминала пользователя в виде текста.

Ввод и вывод распределяется между тремя стандартными потоками:

* stdin — стандартный ввод (клавиатура),
* stdout — стандартный вывод (экран),
* stderr — стандартная ошибка (вывод ошибок на экран).

Потоки также пронумерованы:

* stdin — 0,
* stdout — 1,
* stderr — 2.

Из стандартного ввода команда может только считывать данные, а два других потока могут использоваться только для записи. Данные выводятся на экран и считываются с клавиатуры, так как стандартные потоки по умолчанию ассоциированы с терминалом пользователя. Потоки можно подключать к чему угодно: к файлам, программам и даже устройствам. В командном интерпретаторе bash такая операция называется перенаправлением:

* **< file** — использовать файл как источник данных для стандартного потока ввода.
* **> file** — направить стандартный поток вывода в файл. Если файл не существует, он будет создан, если существует — перезаписан сверху.
* **2> file** — направить стандартный поток ошибок в файл. Если файл не существует, он будет создан, если существует — перезаписан сверху.
* **>>file** — направить стандартный поток вывода в файл. Если файл не существует, он будет создан, если существует — данные будут дописаны к нему в конец.
* **2>>file** — направить стандартный поток ошибок в файл. Если файл не существует, он будет создан, если существует — данные будут дописаны к нему в конец.
* **&>file** или **>&file** — направить стандартный поток вывода и стандартный поток ошибок в файл. Другая форма записи: **>file 2>&1**.

Стандартный ввод

Стандартный входной поток обычно переносит данные от пользователя к программе. Программы, которые предполагают стандартный ввод, обычно получают входные данные от устройства типа клавиатура. Стандартный ввод прекращается по достижении EOF (конец файла), который указывает на то, что данных для чтения больше нет.

EOF вводится нажатием сочетания клавиш Ctrl+D.

Рассмотрим работу со стандартным выводом на примере команды **cat** (от CONCATENATE, в переводе «связать» или «объединить что-то»).

**Cat** обычно используется для объединения содержимого двух файлов.

**Cat** отправляет полученные входные данные на дисплей терминала в качестве стандартного вывода и останавливается после того как получает EOF.

Пример

cat

В открывшейся строке введите, например, **1**и нажмите клавишу Enter. На дисплей выводится **1**. Введите**a** и нажмите клавишу Enter. На дисплей выводится **a**.

Дисплей терминала выглядит следующим образом:

test@111:~/stream$ cat

1

1

a

a

Для завершения ввода данных следует нажать сочетание клавиш Ctrl + D.

Стандартный вывод

Стандартный вывод записывает данные, сгенерированные программой. Когда стандартный выходной поток не перенаправляется в какой-либо файл, он выводит текст на дисплей терминала.

При использовании без каких-либо дополнительных опций, команда **echo** выводит на экран любой аргумент, который передается ему в командной строке:

echo Пример

Аргументом является то, что получено программой, в результате на дисплей терминала будет выведено:

Пример

При выполнении **echo** без каких-либо аргументов, возвращается пустая строка.

Пример

Команда объединяет три файла: file1, file2 и file3 в один файл bigfile:

cat file1 file1 file1 > bigfile

Команда cat по очереди выводит содержимое файлов, перечисленных в качестве параметров на стандартный поток вывода. Стандартный поток вывода перенаправлен в файл bigfile.

Стандартная ошибка

Стандартная ошибка записывает ошибки, возникающие в ходе исполнения программы. Как и в случае стандартного вывода, по умолчанию этот поток выводится на терминал дисплея.

Пример

Рассмотрим пример стандартной ошибки с помощью команды **ls**, которая выводит список содержимого каталогов.

При запуске без аргументов **ls** выводит содержимое в пределах текущего каталога.

Введем команду **ls** с каталогом % в качестве аргумента:

ls %

В результате должно выводиться содержимое соответствующей папки. Но так как каталога % не существует, на дисплей терминала будет выведен следующий текст стандартной ошибки:

ls: cannot access %: No such file or directory

Перенаправление потока

Linux включает в себя команды перенаправления для каждого потока.

Команды со знаками **>** или **<** означают перезапись существующего содержимого файла:

* **>** — стандартный вывод,
* **<** — стандартный ввод,
* **2>** — стандартная ошибка.

Команды со знаками **>>** или **<<** не перезаписывают существующее содержимое файла, а присоединяют данные к нему:

* **>>** — стандартный вывод,
* **<<** — стандартный ввод,
* 2**>>** — стандартная ошибка.

Пример

В приведенном примере команда **cat** используется для записи в файл file1, который создается в результате цикла:

cat > file1

a

b

c

Для завершения цикла нажмите сочетание клавиш Ctrl + D.

Если файла file1 не существует, то в текущем каталоге создается новый файл с таким именем.

Для просмотра содержимого файла file1 введите команду:

cat file1

В результате на дисплей терминала должно быть выведено следующее:

a

b

c

Для перезаписи содержимого файла введите следующее:

cat > file1

1

2

3

Для завершения цикла нажмите сочетание клавиш Ctrl + D.

В результате на дисплей терминала должно быть выведено следующее:

1

2

3

Предыдущего текста в текущем файле больше не существует, так как содержимое файла было переписано командой **>**.

Для добавления нового текста к уже существующему в файле с помощью двойных скобок **>>**выполните команду:

cat >> file1

a

b

c

Для завершения цикла нажмите сочетание клавиш Ctrl + D.

Откройте file1 снова и в результате на дисплее монитора должно быть отражено следующее:

1

2

3

a

b

c

Каналы

Каналы используются для перенаправления потока из одной программы в другую. Стандартный вывод данных после выполнения одной команды перенаправляется в другую через канал. Данные первой программы, которые получает вторая программа, не будут отображаться. На дисплей терминала будут выведены только отфильтрованные данные, возвращаемые второй командой.

Пример

Введите команду:

ls | less

В результате каждый файл текущего каталога будет размещен на новой строке:

file1

file2

t1

t2

Перенаправлять данные с помощью каналов можно как из одной команды в другую, так и из одного файла к другому, а перенаправление с помощью **>** и **>>** возможно только для перенаправления данных в файлах.

Пример

Для сохранения имен файлов, содержащих строку «LOG», используется следующая команда:

dir /catalog | find "LOG" > loglist

Вывод команды **dir** отсылается в команду-фильтр **find**. Имена файлов, содержащие строку «LOG», хранятся в файле loglist в виде списка (например, Config.log, Logdat.svd и Mylog.bat).

При использовании нескольких фильтров в одной команде рекомендуется разделять их с помощью знака канала **|**.

Фильтры

Фильтры представляют собой стандартные команды Linux, которые могут быть использованы без каналов:

* **find** — возвращает файлы с именами, которые соответствуют передаваемому аргументу.
* **grep** — возвращает только строки, содержащие (или не содержащие) заданное регулярное выражение.
* **tee** — перенаправляет стандартный ввод как стандартный вывод и один или несколько файлов.
* **tr** — находит и заменяет одну строку другой.
* **wc** — подсчитывает символы, линии и слова.

Как правило, все нижеприведенные команды работают как фильтры, если у них нет аргументов (опции могут быть):

* **cat** — считывает данные со стандартного потока ввода и передает их на стандартный поток вывода. Без опций работает как простой повторитель. С опциями может фильтровать пустые строки, нумеровать строки и делать другую подобную работу.
* **head** — показывает первые 10 строк (или другое заданное количество), считанных со стандартного потока ввода.
* **tail** — показывает последние 10 строк (или другое заданное количество), считанные со стандартного потока ввода. Важный частный случай tail -f, который в режиме слежения показывает концовку файла. Это используется, в частности, для просмотра файлов журнальных сообщений.
* **cut** — вырезает столбец (по символам или полям) из потока ввода и передает на поток вывода. В качестве разделителей полей могут использоваться любые символы.
* **sort** — сортирует данные в соответствии с какими-либо критериями, например, арифметически по второму столбцу.
* **uniq** — удаляет повторяющиеся строки. Или (с ключом -с) не просто удалить, а написать сколько таких строк было. Учитываются только подряд идущие одинаковые строки, поэтому часто данные сортируются перед тем как отправить их на вход программе.
* **bc** — вычисляет каждую отдельную строку потока и записывает вместо нее результат вычисления.
* **hexdump** — показывает шестнадцатеричное представление данных, поступающих на стандартный поток ввода.
* **strings** — выделяет и показывает в стандартном потоке (или файле) то, что напоминает строки. Всё что не похоже на строковые последовательности, игнорируется. Команда полезна в сочетании с grep для поиска интересующих строковых последовательностей в бинарных файлах.
* **sed** — обрабатывает текст в соответствии с заданным скриптом. Наиболее часто используется для замены текста в потоке: sed s/было/стало/g.
* **awk** — обрабатывает текст в соответствии с заданным скриптом. Как правило, используется для обработки текстовых таблиц, например, вывод ps aux и т.д.
* **sh -s** — текст, который передается на стандартный поток ввода sh -s. может интерпретироваться как последовательность команд shell. На выход передается результат их исполнения.
* **ssh** — средство удаленного доступа ssh, может работать как фильтр, который подхватывает данные, переданные ему на стандартный поток ввода, затем передает их на удаленный хост и подает на вход процессу программы, имя которой было передано ему в качестве аргумента. Результат выполнения программы (то есть то, что она выдала на стандартный поток вывода) передается со стандартного вывода ssh.

Если в качестве аргумента передается файл, команда-фильтр считывает данные из этого файла, а не со стандартного потока ввода (есть исключения, например, команда tr, обрабатывающая данные, поступающие исключительно через стандартный поток ввода).

Пример

Команда **tee**, как правило, используется для просмотра выводимого содержимого при одновременном сохранении его в файл.

wc ~/stream | tee file2

Пример

Допускается перенаправление нескольких потоков в один файл:

ls -z >> file3 2>&1

В результате сообщение о неверной опции «z» в команде **ls** будет записано в файл t2, поскольку stderr перенаправлен в файл.

Для просмотра содержимого файла file3 введите команду **cat**:

cat file3

В результате на дисплее терминала отобразиться следующее:

ls: invalid option -- 'z'

Try 'ls --help' for more information.