Перенаправление, конвейеры

Программы обычно что-то выводят на экран. Этот вывод можно разделить на два типа: к первому относятся результаты работы программы, ко второму – сообщения об ошибках.

В Unix-подобных операционных системах все представляется в виде файлов. Поэтому такая программа, например, как `ls`, выводит свои результаты в специальный файл, который называется *стандартным выводом* (standard output, stdout), а сообщения об ошибках – в специальный файл, называющийся *стандартным выводом ошибок* (standard error, stderr). По умолчанию оба эти файла связаны с экраном и не сохраняются на диск.

Кроме того, многие программы принимают ввод из специального файла *стандартного ввода* (standard input, stdin), который по умолчанию связан с клавиатурой.

Файлам операционной системой выдаются так называется *дескрипторы* — целые числа, по значению которых командная оболочка понимает, какой именно файл требуется. Стандартному вводу соответствует дескриптор 0, стандартному выводу — 1, стандартному выводу ошибок — 2.

Замечание
Часто вместо «стандартный вывод», «стандартный вывод ошибок», «стандартный ввод» используют термины «стандартный поток вывода», «стандартный поток вывода ошибок», «стандартный поток ввода». Слово «поток» добавляют, потому что программы получают ввод или возвращают вывод как последовательность (поток) символов.

Перенаправление стандартного вывода и стандартного вывода ошибок

Чтобы перенаправить стандартный вывод в другой файл вместо экрана, нужно добавить в команду знак больше (>) и имя файла.

```
user@linux-pc:~$ ls -l /usr/bin > ls_output.txt
```

Все, что должно было появиться на экране в качестве вывода команды, сохраняется в указанном выходном файле.

```
user@linux-pc:~$ ls -l ls_output.txt
-rw-r--r-- 1 user user 69300 Jul 30 17:41 ls_output.txt
```

Можно просмотреть содержимое файла ls_output.txt с помощью команды less и убедиться, что он действительно содержит результаты работы команды ls.

Повторим первую команду ls, но укажем имя несуществующего каталога.

```
user@linux-pc:~$ ls -l /bin/usr > ls_output.txt
ls: cannot access '/bin/usr': No such file or directory
```

Почему сообщение появилось на экране, а не было перенаправлено в файл ls_output.txt? Потому что программа ls выводит сообщения об ошибках в стандартный вывод ошибок, а этот поток мы не переопределяли. Обратите внимание, на размер файла ls_output.txt:

```
user@linux-pc:~$ ls -l ls_output.txt
-rw-r--r-- 1 user user 0 Jul 30 17:48 ls_output.txt
```

При перенаправлении вывода файл каждый раз перезаписывается. Поскольку команда ls ничего не вывела, файл был создан заново, но в него ничего не было записано.

Если требуется добавить вывод в конец существующего файл, не перезаписывая его, нужно использовать два знака больше >>:

```
user@linux-pc:~$ ls -l /usr/bin >> ls_output.txt
user@linux-pc:~$ ls -l /usr/bin >> ls_output.txt
user@linux-pc:~$ ls -l ls_output.txt
-rw-r--r-- 1 user user 138600 Jul 30 18:04 ls_output.txt
```

Перенеправление стандартного вывода ошибок выполняется сложнее:

```
user@linux-pc:~$ ls -l /bin/usr 2> ls_errors.txt
```

Номер файлового дескриптора помещается непосредственно перед оператором перенаправления.

Иногда требуется сохранить весь вывод команды в один файл:

```
user@linux-pc:~$ ls -l /usr/bin > ls_output.txt 2>&1
```

В данном случае выполняется два перенаправления: сначала стандартный вывод перенаправляется в файл ls_output.txt, а затем файловый дескриптор 2 (стандартный вывод ошибок) перенаправляется в файловый дескриптор 1 (стандартный вывод).

ВНИМАНИЕ

Порядок перенаправления играет важную роль. Если порядок перенаправления изменить на 2>&1 > ls_output.txt, стандартный вывод ошибок будет перенаправлен на экран.

Перенаправление стандартного ввода

Команда cat читает содержимое одного или нескольких файлов и выводит его на экран

```
cat [имя_файла...]
```

Команда cat часто используется для вывода коротких текстовых файлов.

```
user@linux-pc:~$ cat one.txt
one
user@linux-pc:~$ cat two.txt
two
```

Кроме этого, поскольку cat может принимать сразу несколько файлов, она используется для их объединения.

```
user@linux-pc:~$ cat one.txt two.txt > one_two.txt
user@linux-pc:~$ cat one_two.txt
one
two
```

Если у команды cat отсутствуют аргументы при вызове, она копирует содержимое стандартного ввода в стандартный вывод. Эту особенность можно использовать для создания коротких текстовых файлов.

Введите команду, затем текст, по окончанию ввода текста нажмите комбинацию CTRL+D.

```
user@linux-pc:~$ cat > my_name.txt
My name is [your name].
```

Чтобы увидеть результат, воспользуйтесь командой:

```
user@linux-pc:~$ cat my_name.txt
My name is Igor.
```

Операция перенаправления ввода является обратной по отношению к перенаправлению вывода. При перенаправлении вывода берется вывод команды и перенаправляется в файл, а при перенаправлении ввода берется содержимое файла и перенаправляется в команду.

Для обозначения перенаправления ввода используется знак меньше (<):

```
user@linux-pc:~$ cat < my_name.txt
My name is Igor.
```

Используя операцию перенаправления <, мы изменили источник данных для стандартного ввода с клавиатуры на файл my_name.txt. Результат получился такой же, как если бы мы передали имя файла в качестве аргумента. Этот способ не имеет никакого преимущества по сравнению с передачей аргумента, он просто демонстрирует, как можно использовать файлы в роли источника данных.

Pipes – конвейеры

Иногда возникает необходимость передавать выход одной команды на вход другой. Для этого можно использовать перенаправление, но это связано с применением довольно сложных конструкций:

```
user@linux-pc:~$ ls -l /usr/bin > ls_output.txt
user@linux-pc:~$ less < ls_output.txt
total 148280
lrwxrwxrwx 1 root   root           4 Feb 17  2020 NF -> col1
-rwxr-xr-x 1 root   root      141856 Mar 17 10:14 VGAuthService
lrwxrwxrwx 1 root   root           1 Feb  8  2020 X11 -> .
-rwxr-xr-x 1 root   root       59736 Sep  5  2019 [
...
```

Вместо перенаправления вывода команды в файл можно перенаправить этот вывод непосредственно в другую команду. Такой процесс называется передачей данных по конвейеру (или каналу):

```
user@linux-pc:~$ ls -l /usr/bin | less
```

Не следует рассматривать передачу данных по каналу как последовательное выполнение двух команд. Фактически Linux выполняет обе команды одновременно, соединяя в системе выход одной команды с входом другой. Как только первая команда вырабатывает какой-то вывод, он немедленно передается во вторую команду.

Фильтры

Конвейеры позволяют объединить вместе несколько команд. Часто команды, используемые таким образом, называют *фильтрами*. Фильтры принимают ввод, изменяют его определенным образом и выводят результат.

sort

Предположим, необходимо составить список все исполняемых файлов в каталогах /bin и /usr/bin, расположив их по алфавиту.

```
user@linux-pc:~$ ls /bin /usr/bin | less
```

Поскольку в команде указаны два каталога, вывод команды ls будет состоять из двух отсортированных списков, что не удовлетворяет решению задачи.

Воспользуемся командой sort.

```
user@linux-pc:~$ ls /bin /usr/bin | sort | less
```

uniq

Команда uniq часто используется вместе с командой sort. uniq принимает на вход отсортированный список (либо со стандартного ввода, либо из файла) и по умолчанию удаляет повторяющиеся строки из списка.

```
user@linux-pc:~$ ls /bin /usr/bin | sort | uniq | less
```

или просто

```
user@linux-pc:~$ ls /bin /usr/bin | sort -U | less
```

Если нам нужно получить список дубликатов, то команду uniq нужно использовать следующим образом:

```
user@linux-pc:~$ ls /bin /usr/bin | sort | uniq -d | less
```

wc

Команда wc (word count) используется для подсчета строк, слов и байтов в файле.

```
user@linux-pc:~$ wc ls_output.txt
 1100 10307 69300 ls_output.txt
```

Вот таким образом ее можно использовать для подсчета числа исполняемых файлов в каталогах `/bin` и `/usr/bin`

```
user@linux-pc:~$ ls /bin /usr/bin | sort | uniq | wc -l
```

grep

Команда grep часто используется для поиска в файлах текста по шаблону:

```
grep шаблон [файл...]
```

Когда grep находит в файле совпадение с шаблоном, она выводит строку с найденным совпадением.

Предположим нам нужно найти все программы, в названии которых есть подстрока «zip»

```
user@linux-pc:~$ ls /bin /usr/bin | sort | uniq | grep zip
bunzip2
bzip2
bzip2recover
funzip
gpg-zip
gunzip
gzip
unzip
unzipsfx
zipdetails
zipgrep
zipinfo
```

У команды grep есть два удобных параметра: -i требует от grep игнорировать регистр символов; -v требует от grep выводить строки, в которых совпадение с шаблоном не найдено.

Литература

1. Шотсс У. “Командная строка Linux.”, глава 7
2. Блум Р., Бреснахэн К. “Командная строка Linux и сценарии оболочки.”, глава 10, подраздел “Перенаправление ввода и вывода”