آموزش Linux Essentials

علی رشیدی

۱۳۹۵/۱۱/۲۵

ارائه شده توسط

وبلاگ علی رشیدی

http://Ali-RNT.blog.ir

برای آموزش‌های بیشتر به وبلاگ مراجعه کنید.

قسمت ۱۴

ورودی-خروجی استاندارد و دستورات فیلتر

علی رشیدی

بها: نشر رایگان این آموزش

۱. تغییر مسیر I/O و خط لوله های دستورات

۱.۱ کانال‌های استاندارد

اغلب دستورات لینوکس برای گرفتن ورودی، پردازش آن‌ها و دادن خروجی به کار میروند. برای مثال، با نوشتن:

```
$ grep xyz
```

خطوطی را مینویسید (ورودی)، grep آن‌ها را پردازش میکند و فقط خطوط شامل xyz را به خروجی میبرد (خطوط درشت ورودی ما و خطوط معمولی خروجی اند):

abc def

123 xyz

123 xyz

aaa bbb

YYYxyzYYY

YYYxyzYYY

Ctrl + d

مفهومی به نام «کانال‌های استاندارد»[1] وجود دارد و کانال‌های استاندارد ما در اینجا، صفحه‌کلید به عنوان «ورودی استاندارد[2]» و کنسول یا ترمینال به عنوان «خروجی استاندارد[3]» میباشند. یک کانال دیگر «خروجی استاندارد خطا[4]» است که پیغام های خطای برنامه، مانند فایل ورودی ناموجود یا اشتباه نحوی در عبارت باقاعده، به آن منتقل میشوند.

در این قسمت از آموزش به شما یاد میدهیم که چگونه خروجی استاندارد یک برنامه را به یک فایل منتقل کنید، ورودی استاندارد برنامه‌ای را از یک فایل بگیرید و خروجی استاندارد برنامه‌ای را به عنوان ورودی استاندارد به برنامه‌ای دیگر بدهید. به این طریق میتوانیم با در کنار هم قرار دادن ابزار های کوچک و ساده لینوکس یک ساختار هدفمند و یک برنامه پیچیده تشکیل دهیم.

کانال	نام	نام کوتاه	دستگاه	کاربرد
0	Standard Input	stdin	صفحه‌کلید	ورودی برنامه‌ها
1	Standard Output	stdout	صفحه نمایش	خروجی برنامه
2	Standard Error Output	stderr	صفحه نمایش	پیام‌های خطای برنامه ها

معمولاً از نام کوتاه یا شماره کانال استفاده میکنیم. شل میتواند این کانال‌ها را از برنامه‌ای به برنامه دیگر منتقل کند بدون آنکه برنامه متوجه شود. خروجی کانال‌ها تنها به صفحه نمایش یا کنسول منتقل نمیشود، بلکه گاهی به یک فایل انتقال می یابد. این فایل میتواند یک دستگاه، مانند پرینتر یا دستگاه ذخیره سازی، یا یک فایل متنی باشد.

کانال ورودی استاندارد هم میتواند به روش‌هایی منتقل شود، به گونه‌ای که به جای صفحه‌کلید ورودی اش را از یک فایل بگیرد.

1 Standard Channels
2 Standard Input
3 Standard Output
4 Standard Error Output

نکته: صفحه‌کلید و صفحه نمایش ترمینالی که با آن کار میکنید، به وسیله ی dev/tty/ در دسترس است. در صورتی که اطلاعات را از آن بخوانید به منزله کیبرد، و در صورت نوشتن اطلاعات در آن، به عنوان صفحه نمایش به کار میرود. مثال اول متنی را در tty مینویسد و میبینید که مانند این است که آنرا مستقیماً در صفحه نمایش بنویسیم:

```
$ echo “Hello” > /dev/tty
Hello
```

و در مثال دوم فایل ورودی grep را همین دستگاه تعیین کرده ایم، که مانند خواندن از صفحه‌کلید است:

```
$ grep ali /dev/tty
mahdi
alireza
alireza
```

این فایل‌های خاص در قسمت‌های بعدی به طور مفصل‌تر بحث خواهند شد.

۱.۲ انتقال (تغییر مسیر) کانال‌های استاندارد

عملگر شلِ < (علامت بزرگ‌تر) برای انتقال کانال خروجی استاندارد به کار میرود. برای مثال:

```
$ ls -laF > fileList
$ _
```

خروجی دستور ls -laF را به فایل fileList منتقل میکند و همانطور که میبینید چیزی روی صفحه نمایش داده نمیشود و شل آماده دریافت دستور بعدی است.

نکات:

۱. فایل fileList در صورتی که موجود نباشد ایجاد و در صورتی که هم‌اکنون موجود باشد، اطلاعات آن رونویسی میشود.

۲. ایجاد یا رونویسی فایل قبل از اجرای برنامه‌ای که خروجی آن به فایل منتقل میشود، انجام میگیرد. یعنی اگر دستور اشتباه باشد یا دارای اشتباه نوشتاری/نحوی باشد، قبل از بررسی این مورد فایل ایجاد/رونویسی می‌شود اما خالی میماند.

۳. برای جلوگیری از رونویسی فایل، از دستور زیر استفاده کنید تا در صورت انتقال خروجی به یک فایل موجود، یک خطا به جای رونویسی رخ دهد:

```
$ set -o noclobber
```

حالا که از عملگر انتقال استفاده کردید، بیایید محتویات فایل fileList را ببینیم، اگر دقت کنید میبینید که فایل fileList هم با حجم صفر وجود دارد:

```
$ less fileList
...
-rw-r--r--  1 ali-rnt ali-rnt      0 Feb 20 18:43 fileList
-rw-r--r--  1 ali-rnt ali-rnt  68593 Dec 21 21:39
IntegratingQtQml.odt
-rw-r--r--  1 ali-rnt ali-rnt  13280 Aug 31 21:09 LE-10.odt
```

```
-rw-r--r--  1 ali-rnt ali-rnt  17625 Sep 28 17:39 LE-11.odt
-rw-r--r--  1 ali-rnt ali-rnt  14336 Oct 27 19:16 LE-12.odt
-rw-r--r--  1 ali-rnt ali-rnt  99939 Oct 27 19:29 LE-12.pdf
…
```

این نشانی بر این نکته است که فایل fileList قبل از اجرای دستور ایجاد شده و حجم آن هم صفر بوده، و پس از اجرای دستور خروجی به آن منتقل شده. توجه کنید که ls ابتدا محتوای دیرکتوری را میخواند و سپس خروجی میدهد.

برای اینکه خروجی یک دستور را به فایلی ببرید، بدون اینکه محتوای قبلی آن پاک شود (اضافه کردن یا append کردن) از عملگر << به جای < استفاده کنید. فایل مورد نظر در صورت عدم وجود ایجاد میشود.

یک راه دیگر برای انتقال خروجی استاندارد، استفاده از بک تیک (علامت `) می باشد. این روش خروجی استاندارد دستوری که داخل بک تیک ها قرار دارد را به عنوان ورودی به برنامه میدهد. یک مثال:

```
$ less dates
22/02 Substitution and input
23/02 Pipelines

$ date +%d/%m
22/02
$ grep `date +%d/%m` dates
22/02 Substitution and input
```

اکنون نوبت به تغییر مسیر ورودی استاندارد میرسد که با عملگر < امکان‌پذیر است. مثلاً فیلتر wc که تعداد کلمات را میشمارد، در مثال زیر ورودی اش را به جای کیبرد از فایل میگیرد:

```
$ ls > fileList
$ wc -w < fileList
45
```

نکته: برای ارسال محتوای چند فایل به یک برنامه میتوانید از این روش استفاده کنید. با | کمی جلوتر آشنا میشوید. فعلاً همین شکل را به خاطر داشته باشید:

```
$ cat file1 file 2 file3 | wc -w
```

و اکنون شما باید بدانید که چگونه ورودی و خروجی استاندارد را همزمان منتقل کنید. مثلا:

```
$ wc -w <aFile >wordCount
$ cat wordCount
213
```

کمی قبل تر گفته شد که یک کانال استاندارد دیگر هم به نام کانال خروجی استاندارد خطا وجود دارد که با شماره ۲ مشخص میشود. نکته‌ای که هنگام استفاده از عملگر ها باید بدانید این است که باید کانالی که عملگر روی آن‌ها اعمال می‌شود مشخص شود.

اینکار برای کانال‌های ۰ و ۱ اختیاری است. یعنی نوشتن < با <1 و > با >0 فرقی ندارد. اما برای کانال ۲ اجباری است. با استفاده از این نکته و همچنین اینکه برای انتقال یک کانال به کانال دیگر از &< استفاده میکنیم، میتوانیم کار با کانال ۲ را شروع کنیم.

مثال:

```
$ make >make.log 2>&1
```

دستور بالا خروجی استاندارد و خروجی استاندارد خطای برنامه make را به make.log منتقل میکند.

توجه کنید که ترتیب مهم است، دستور بالا ابتدا مشخص میکند که خروجی استاندارد به make.log منتقل شود و بعد میگوید که کانال ۲ را به کانال ۱ یعنی خروجی استاندارد منتقل کند، و چون خروجی استاندارد را فایل make.log مشخص کرده‌ایم به آن منتقل میشود. اما اگر به این ترتیب بنویسیم:

```
$ make 2>&1 >make.log
```

ابتدا کانال ۲ را به خروجی استاندارد (یعنی صفحه نمایش، ترمینال یا کنسول) منتقل میکند و سپس خروجی استاندارد را به make log. میبرد که در این مرحله شامل کانال ۲ (خروجی استاندارد خطا) نمیشود.

تمرین‌ها

۱. با گزینه U- در دستور ls فایل‌ها بدون اینکه مرتب شوند به خروجی میروند. اما با این حال پس از اجرای ls -laU >fileList سایز fileList همچنان صفر میماند. علت چیست؟

۲. خروجی دو دستور زیر را مقایسه و علت را بیان کنید.

```
$ ls /tmp
$ ls /tmp >ls-tmp.txt
```