آموزش Linux Essentials

علی رشیدی

۱۳۹۶/۱۱/۲۵

ارائه شده توسط

کدنگار

http://Ali_RNT.blog.ir

قسمت ۱۵

خط لوله، فیلترها، خواندن و نوشتن فایل‌ها

خط لوله

قبلاً برای انتقال خروجی یک دستور به دستوری دیگر، از یک فایل میانی استفاده میکردیم. به این صورت که خروجی دستور اول را به یک فایل منتقل و ورودی برنامه دوم را فایل اول تعیین میکردیم. مثلا:

```
$ grep ab f1.txt > inter.tmp
$ wc -w < inter.tmp
```

اکنون میتوانیم بدون داشتن دردسر فایل‌های میانی، از خروجی برنامه grep به عنوان ورودی برنامه wc استفاده کنیم:

```
$ grep ab f1.txt | wc -w
```

با ترکیب این خط لوله ها میتوان ساختاری بزرگ و کاربردی پیاده‌سازی کرد. مانند:

```
$ cut -d: -f1 /etc/passwd | sort | pr -2 >users.txt
```

با دستورات استفاده شده در این خط بعداً آشنا خواهید شد اما این خط به طور خلاصه محتویات passwd را که کاربران هستند مرتب کرده و سپس در دو ستون در فایل users.txt مینویسد.

نتایج میانی

گاهی اوقات بر خلاف دستور قبلی، که فقط نتیجه پایانی مهم است، آنچه تا کنون در کانال‌ها وجود دارد نیز به دردمان میخورد. برای مثال میخواهیم لیست کاربران قبل از مرتب سازی را در USUsers.txt و پس از مرتب سازی را در Users.txt ذخیره کنیم. احتمالاً اولین راه حلی که به ذهن میرسد دستور زیر است:

```
$ cut -d: -f1 /etc/passwd >ususers.txt | sort | pr -2 >users.txt
```

اما همانطور که میبینید بعد از اجرای این دستور، محتویات ususers.txt همان است که انتظار میرفت اما users.txt خالی است. مشکل از کجاست؟ واضح است: خروجی cut فقط به ususers.txt می‌رود و چیزی به دستورات بعدی نمیرود تا چاپ شود. باید راهی پیدا کنیم تا خروجی یک دستور علاوه بر انتقال توسط خط لوله، بتواند به یک فایل نیز نوشته شود. دستور tee این کار را برای ما انجام میدهد. این دستور ورودی اش را به فایلی که با آرگومان مشخص شده میریزد و سپس همان را به خروجی میبرد، که در اینجا آن خروجی توسط خط لوله به برنامه بعدی داده میشود. شکل تصحیح شده دستور قبل به این صورت است:

```
$ cut -d: -f1 /etc/passwd | tee ususers.txt | sort | pr -2 >users.txt
```

دستورات فیلتر

در جهت پیشبرد ایده اصلی پشت لینوکس و البته در اصل یونیکس، ابزارهایی وجود دارند که ورودی را از ورودی استاندارد گرفته و آنرا تغییر داده، در خروجی استاندارد مینویسند. این ابزار ها فیلتر نام دارند. اگر ورودی برای آن‌ها در نظر گرفته نشود، مطابق استاندارد ورودی را از کیبرد میخوانند. نمونه‌ای از این ابزارها را در قبل دیدیم: sort یک فیلتر است که خطوط ورودی اش را مرتب میکند.در ادامه مبحث با فیلتر های بیشتری آشنا میشویم.

خواندن و نوشتن فایل‌ها

به خروجی بردن و اتصال فایل‌ها – cat

دستور (concatenate) cat برای متصل کردن فایل‌های داده شده به آن توسط آرگومان ها ساخته شده، اگر این دستور را با یک فایل اجرا کنید، محتوای فایل را به خروجی میبرد. اگر هیچ فایلی تعیین نشود، ورودی اش را از کیبرد میخواند. مزیت اصلی cat گزینه های مفید آن است که در جدول زیر میبینید:

گزینه	نتیجه
-b	**تمامی خطوط غیر خالی را در خروجی شماره گذاری میکند.**
-E	**چاپ کاراکتر $ در انتهای هر خط، مفید برای تشخیص پایان خط**
-n	**شماره گذاری تمام خطوط در خروجی**
-s	**به جای خطوط خالی متوالی، یک خط خالی تنها چاپ میکند.**
-T	**نمایش کاراکتر Tab به صورت ^I**
-v	**کاراکتر کنترلی c را به صورت ^c نمایش میدهد، کاراکتر a با کد اسکی بیش از ۱۲۷ را با M- a نمایش میدهد.**
-A	**نمایش همه چیز، مانند -vET**

تمرین

چگونه میتوان فهمید که فایل‌های یک دیرکتوری دارای اسم عجیب (مثلاً شامل space در انتهای اسم یا کاراکترهای کنترلی) است یا خیر؟

ابتدا و انتهای فایل – دستورات head و tail

دستور head به طور پیشفرض ۱۰ خط اول فایل و دستور tail نیز ۱۰ خط آخر فایل را نمایش میدهند. گزینه ی n- تعداد خطوط را تعیین میکند.

```
$ head -n 20 myFile.txt
```

این دستور ۲۰ خط اول myFile.txt را چاپ میکند. گزینه دیگر این دو دستور c- است که به جای تعداد خط، مشخص میکند چه حجمی از فایل نمایش داده شود، به طور پیشفرض این حجم بر حسب بایت است اما با افزودن b, k, m به انتهای عدد، این عدد در ۵۱۲، ۱۰۲۴ یا ۱۰۴۸۵۷۶ ضرب میشود. دستور tail هم به طور مشابهی، حجم یا خطوط مشخص شده پایانی را نمایش میدهد.

نکته: دادن عدد منفی به دستور head (چه برای حجم چه تعداد خط) باعث می‌شود تمام فایل از ابتدا نمایش داده شود، به جز قسمت مشخص شده. مثلا:

```
$ head -c -20
```

تمام ورودی استاندارد (ورودی داده شده با صفحه کلید) را خوانده، تمام آن به جز ۲۰ بایت آخر را نشان میدهد.

نکته: دستور tail نیز در صورت گرفتن عدد با علامت مثبت، همه چیز را از موقعیت مشخص شده به بعد میخواند. مثلا:

```
$ tail -n +3 file
```

محتویات فایل را از خط ۳ به بعد نمایش میدهد.

دستور tail گزینه مهم f- را هم دارد که بعد از نمایش خروجی، منتظر میماند و هر گاه چیزی به انتهای فایل اضافه شود، آن را نمایش میدهد. اگر نام چند فایل به tail داده شود، با نشان دادن یک سرخط به شما نشان میدهد که چه فایلی تغییر کرده است.

تمرین

- چگونه میتوان فقط خط ۱۳ ام ورودی را به خروجی برد؟
- فایلی را ایجاد و tail -f را روی آن فراخوانی کنید. با یک برنامه دیگر یا یک ترمینال دیگر در آن بنویسید و سیو کنید. خروجی tail را آنالیز کنید. این خروجی هنگام فراخوانی چند فایل چگونه است؟
- اگر از فایلی که توسط tail -f تماشا می‌شود چیزی کم کنیم چه میشود؟
- خروجی دستورات زیر را

```
$ echo Hello >/tmp/hello
$ echo “Hiya World” >/tmp/hello
```

بعد از اجرای دستور زیر در یک پنجره دیگر، بعد از اجرای دستور اول توضیح دهید:

```
$ tail -f /tmp/hello
```