پژوهشی دات آی آر
Linux | Security

آموزش فارسی لینوکس : قسمت سوم

پنجشنبه, ۱۲ تیر ۱۳۹۳، ۰۴:۴۷ ق.ظ

100.2: First Steps in Linux

Redirecting .....

در لینوکس هر برنامه یک ورودی و یک خروجی دارد . به طور دقیق تر یک خروجی به عنوان خروجی صحیح (stdout) و یک خروجی غلط (stderr) و یک ورودی به نام stdin دارد . به صورت پیش فرض در خیلی از برنامه ها مانند ls و ... خروجی (هم درست و هم غلط) در Display نمایش داده می شود یعنی می توان آن را دید برای مثال در ترمینال

اما زمانی می خواهیم خروجی درست و یا خروجی غلط یک برنامه را به صورت جدا در داخل فایلی ذخیره کنیم . به مثال زیر توجه کنید :

```
$ls -l
```

در دستور و برنامه ی بالا تنها stdout داریم لذا تنها خروجی درست در standard output چاپ می شود .

```
$ls -l > Out
```

در دستور بالا خروجی درست برنامه را درون فایل Out ریختیم . توجه داشته باشید که دستور بالا معادل دستور زیر است :

```
$ls -l 1> Out
```

حال فرض کنید فایلی با نام Pazhoheshi در دایرکتوری جاری وجود ندارد . به مثال زیر توجه کنید :

```
$ls -l Pazhoheshi
```

نتیجه ی حاصل نمایش ارور زیر در Standard Output می شود :

```
ls: cannot access Pazhoheshi: No such file or directory
```

حال برای ذخیره کردن خروجی غلط یا همان Stderr یا همین ارور دریافتی بالا , در یک فایل به صورت زیر عمل می کنیم :

```
$ls -l Pazhoheshi 2> Out
```

طبیعتا اگر بخواهیم به صورت هم زمان خروجی درست و غلط را در دو فایل جدا از هم , ذخیره کنیم . به صورت زیر عمل خواهیم کرد : (با فرض وجود داشتن فایل Test و وجود نداشتن فایل Pazhoheshi)

```
$ls -l Pazhoheshi Test > Stdout 2> Stderr
```

طبقه بندی موضوعی

- مقاله آموزشی (۷)
- فیلم آموزشی (۷)
- برنامه اختصاصی (۵)
- ارسال بلاگ (۲)
- آموزش فارسی لینوکس (۳)

خلاصه آمار

مجموع نمایش‌ها	۱۴۳۸۰
مجموع بازدیدکننده‌ها	۵۰۲۵
بازدیدکننده‌های امروز	۶
نمایش‌های دیروز	۱۰۰
مجموع مطالب	۲۰
مجموع نظرات	۴۳
عمر سایت	۱۶۶ روز
حاضرین در سایت	۱

کلمات کلیدی

متاسپلویت | Port Forwarding | کرک هندشیک | کرک WPA | pyrit | آموزش فارسی لینوکس | آموزش فارسی lpic 1 | آموزش فارسی lpic | آموزش lpic | mini wireless ids | exploits | اسنیف VPN | جعل dns | باگ | امنیت | Heartbleed | کش سرور | نحوه ی کار dns | query | nslookup | dns چیست | dns سرور | سویچ های nmap | سویچ sT | اسکن پورت | آموزش nmap | handshake | sS- | باگ در https | امنیت https

بایگانی

- شهریور ۱۳۹۳ (۱)
- تیر ۱۳۹۳ (۷)
- خرداد ۱۳۹۳ (۱)
- اردیبهشت ۱۳۹۳ (۳)
- فروردین ۱۳۹۳ (۸)

پیوندها

پلیس فضای تولید و تبادل اطلاعات نیروی انتظامی جمهوری اسلامی ایران
وبلاگ تخصصی تیم امنیتی سرزمین امن
آریا دیتا سنتر
تیم امنیتی هکران کلاه سیاه ایران - آموزش امنیت و راه های مقابله با هک
TBH | Forums | Turk Black Hat
وب سایت شخصی مسلم حقیقیان
اشتراک هک و امنیت

اما زمانی پیش می آید که می خواهیم خروجی درست و غلط در یک فایل ذخیره شود . برای این منظور می توان به یکی از صورت های زیر عمل کرد :

```
$ls -l Pazhoheshi Test > Out 2>&1
$ls -l Pazhoheshi Test 2> Out 1>&2
$ls -l Pazhoheshi Test &> Out
```

نکته : در تمامی مثال های بالا از کارکتر بزرگ تر (>) استفاده کردیم . این کار باعث overwrite می شود (در صورتی که فایل از قبل وجود داشته باشد) . برای Append کردن خروجی به ادامه ی فایل باید از >> استفاده کرد . برای مثال : (با فرض وجود نداشتن دو فایل Text , Test)

```
$ls -l Pazhoheshi Test Text &>> Out
```

Piping ....

دستور less :

زمانی پیش می آید که تعداد خطوط چاپ شده ی یک فایل در ترمینال زیاد است و ترمینال فقط خطوط آخر را برای ما نمایش می دهد . لذا برای کنترل صفحه های باز شده می توان از این برنامه استفاده کرد

رایج ترین کلید های مورد استفاده در این برنامه up / down / q / Ctrl + b / Ctrl + d می باشد . دو کلید Ctrl + b و Ctrl + d به ترتیب به صفحه ی قبلی و صفحه ی بعدی می روند .

دستور more :

دقیقا همان کار less را انجام می دهد با این تفاوت که دیگر نمی توان به خطوط و صفحات بالایی رفت . معروف ترین کلید های استفاده شده در این برنامه q / Enter / Space می باشد .

more is a basic pager, which allows you to scroll downwards, one page at a time. Only downwards.

less is also a pager, but has addition functionality to scroll upwards and downwards through the input, in addition to several other extensions.

less is more

```
$cat /etc/apt/sources.list | less
$cat /etc/apt/sources.list | more
```

اصطلاحا به چنین برنامه هایی Pager می گویند .

نکته : با توجه به قسمت Redirecting می توان نتیجه گرفت که در دو دستور بالا تنها خروجی درست یا Stdout به عنوان ورودی وارد برنامه ی less و more می شود . این موضوع با مثال زیر قابل فهم است : (با فرض وجود نداشتن فایل Test)

```
$cat /etc/passwd Test | less
```

نتیجه این خواهد شد که فقط محتویات فایل passwd برای ما نمایش داده می شود . اما برای این که خروجی غلط یا همان stderr هم به عنوان ورودی وارد برنامه ی less شود به صورت زیر عمل می کنیم :

```
$cat /etc/passwd Test 2>&1 | less
```

نتیجه این می شود که در Standard Output عبارت زیر را نیز خواهیم دید :

```
cat: Test: No such file or directory
```

Using Man Pages .....

برای هر برنامه توضیحاتی مفصل همراه با معرفی انواع سوییچ ها و ... وجود دارد . برای این منظور می توان از برنامه ی man استفاده کرد . مثال :

```
$man ls
$man passwd
```

نکته ی حایز اهمین این جاست که هر برنامه برای خود یک section number دارد :

```
The table below shows the section numbers of the manual followed by the type
contain.

1   Executable programs or shell commands
2   System calls (functions provided by the kernel)
3   Library calls (functions within program libraries)
4   Special files (usually found in /dev)
5   File formats and conventions eg /etc/passwd
6   Games
7   Miscellaneous (including macro packages and conventions), e.g. man(7), g
8   System administration commands (usually only for root)
9   Kernel routines [Non standard]
```

برای مثال در برنامه ی ls عبارت زیر را خواهیم دید :

```
LS(1) User Commands
```

یعنی section برنامه ی ls یک می باشد و همان طور که در جلوی آن نوشته شده است , این برنامه یک برنامه ی User Commands است . مثال دیگر :

```
$man shadow
SHADOW(5) File Formats and Conversions
```

همانند دیگر برنامه ها , این برنامه هم دارای سوییچ های متعدد است . برای مثال سوییچ a- و w- و k- . اجازه دهید در قالب مثال این دو سوییچ را بررسی کنیم .

```
$man -a passwd
```

با استفاده از این سوییچ می توان تمامی passwd ها را مشاهده کرد .

```
iman@ubuntu:~/test$ man -a passwd
--Man-- next: passwd(1ssl) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]

--Man-- next: passwd(5) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]
```

و سوییچ w- :

```
iman@ubuntu:~/test$ man -w shadow
/usr/share/man/man5/shadow.5.gz

-w, --where, --location
 Don't  actually display the manual pages, but do print the location(s) of
 the source nroff files that would be for matted.
```

و در نهایت سوییچ k- :

```
man -k printf
 Search the short descriptions and manual page names for the keyword printf
 as  regular  expression.   Print  out  any matches.  Equivalent to apropos
```

برای مثال :

```
$man -k partition
```

با استفاده از این سوییچ می توان تمامی برنامه های مرتبط با partition همراه با توضیحی مختصر و شماره ی section آن مشاهده کرد .

حال به مثال زیر توجه کنید : (به قسمتی که Bold هست توجه کنید)

```
iman@ubuntu:~/test$ man -k passwd
chgpasswd (8)         - update group passwords in batch mode
chpasswd (8)          - update passwords in batch mode
Crypt::PasswdMD5 (3pm) - Provides interoperable MD5-based crypt() functions
fgetpwent_r (3)       - get passwd file entry reentrantly
getpwent_r (3)        - get passwd file entry reentrantly
gpasswd (1)           - administer /etc/group and /etc/gshadow
grub-mkpasswd-pbkdf2 (1) - generate hashed password for GRUB
htpasswd (1)          - Manage user files for basic authentication
lppasswd (1)          - add, change, or delete digest passwords.
mkpasswd (1)          - Overfeatured front end to crypt(3)
pam_localuser (8)     - require users to be listed in /etc/passwd
passwd (1)            - change user password
passwd (1ssl)         - compute password hashes
passwd (5)            - the password file
passwd2des (3)        - RFS password encryption
smbpasswd (5)         - The Samba encrypted password file
smbpasswd (8)         - change a user's SMB password
```

همان طور که مشاهده می کنید دو passwd با شماره section مختلف داریم . یعنی 1 و 5 . حال می توان به صورت زیر passwd مورد نظر خودمون رو انتخاب کنیم . برای مثال :

```
$man 5 passwd
```

توجه داشته باشید که اگر شماره ی 5 را ذکر نکنیم . به صورت پیش فرض passwd با شماره ی section یک باز خواهد شد .

در داخل توضیحاتی که برنامه ی man در اختیار ما قرار می دهد می توان به قسمتی به نام SEE ALSO اشاره کرد . که در آن دیگر برنامه ی ها مرتبط را می توان مشاهده کرد . برای مثال :

```
$man fdisk

SEE ALSO
       cfdisk(8), sfdisk(8), mkfs(8), parted(8), partprobe(8), kpartx(8)
```

Using whatis and apropos ....

کار دو برنامه ی apropos , whatis تقریبا شبیه به man هست با این تفاوت که توضیحاتی بسیار جزیی تر را در اختیار ما قرار می دهد . برای مثال :

```
iman@ubuntu:~/test$ whatis mkdir
mkdir (2)            - create a directory
mkdir (1)            - make directories
```

نکته : این دو برنامه یک دیتابیس و یا پایگاه داده برای خود دارند که هر برنامه همراه با توضیحاتی مختصر درون آن ثبت شده است . این DB متشکل از دو ستون است :

الف) ستون نام برنامه
ب) ستون توضیحات

نکته ی قابل توجه این جاست که whatis فقط در ستون الف به دنبال ورودی می گردد اما apropos هم در ستون الف و هم در ستون ب به دنبال ورودی می گردد

نتیجه : apropos زمانی استفاده می شود که هیچ ایده ای نداریم مثلا اسم برنامه (ها) پارتیشن بندی رو نمی دونیم . مثال :

```
iman@ubuntu:~/test$ apropos partition
addpart (8)     - simple wrapper around the "add partition" ioctl
all-swaps (7)   - event signalling that all swap partitions have been activat
cfdisk (8)      - display or manipulate disk partition table
delpart (8)     - simple wrapper around the "del partition" ioctl
fdisk (8)       - manipulate disk partition table
mpartition (1)  - partition an MSDOS hard disk
partprobe (8)   - inform the OS of partition table changes
partx (8)       - tell the Linux kernel about the presence and numbering of o
sfdisk (8)      - partition table manipulator for Linux
```

104.5: Manage File Permissions and Ownership

Description Candidates should be able to control file access through the proper use of permissions and ownerships.

Key Knowledge Areas:

- Manage access permissions on regular and special files as well as directories.
- Use access modes such as suid, sgid and the sticky bit to maintain security.
- Know how to change the file creation mask.
- Use the group field to grant file access to group members.

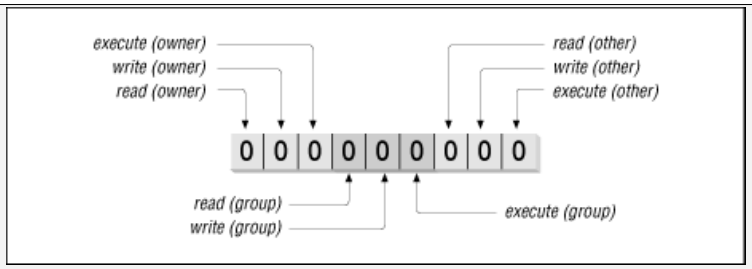
The following is a partial list of the used files, terms and utilities:

- chmod
- umask
- chown
- chgrp

مفاهیم پرمیشن در لینوکس :

ابتدا فرض کنید که Permissions در لینوکس با 9 بیت مشخص می شود یعنی 3 تا گروه 3 تایی . به شکل زیر توجه کنید :

تصویر اول



هر خانه مشخصه ی 1 بیت است که در مجموع 9 بیت را تشکل می کند هم چنین هر خانه دو مقدار بیش تر نمی تواند داشته باشد : صفر و یک (یک یعنی حضور سیگنال و صفر یعنی عدم حضور سیگنال)

همان طور که از شکل مشخص است این 9 بیت به 3 دسته ی Owner , Group , Other تقسیم شده است و هر کدام از 3 دسته می تواند به ترتیب 3 مقدار Read , Write , execute را داشته باشد . اگر این مقدار ها فعال باشد آن بیت مربوطه عدد 1 را به خود اختصاص می دهد و اگر این مقدار غیر فعال باشد آن بیت مربوطه عدد 0 را به خود اختصاص می دهد .

برای مثال اگر در دسته ی Owner مقدار Read را فقط داشته باشیم : 100
برای مثال اگر در دسته ی Group دو مقدار Write و execute را داشته باشیم : 011
برای مثال اگر در دسته ی Other دو مقدار Read , Write را داشته باشیم : 110

توجه داشته باشید که این اعداد بدست آمده یعنی 100 و 011 و 110 همگی در مبنای دو یعنی باینری می باشند .

حال اگر این 3 دسته را به طور منظم پشت سر هم قرار دهیم به عدد رو به رو خواهیم رسید : 100/011/110 یا 100011110

حال اگر هر کدام از این اعداد را به مبنای 10 (دسیمال) ببریم و آن ها را به صورت منظم پشت سر هر قرار دهیم به عدد رو به رو خواهیم رسید : 4/3/6 یا 436

100 => (1 x 2^**2**) + (0 x 2^**1**) + (0 x 2^**0**) = 4 + 0 + 0 = 4
011 => (0 x 2^**2**) + (1 x 2^**1**) + (1 x 2^**0**) = 0 + 2 + 1 = 3
110 => (1 x 2^**2**) + (1 x 2^**1**) + (0 x 2^**0**) = 4 + 2 + 0 = 6

پس عدد دسیمال 4 مشخصه ی پرمیشن دسته ی Owner و عدد دسیمال 3 مشخصه ی پرمیشن دسته ی Group و عدد دسیمال 6 مشخصه ی پرمیشن دسته ی Other می باشد .

مثال : پرمیشن یک فایل 665 است .
حل : از سمت چپ , 6 اولی برای دسته ی Owner و 6 دومی برای دسته Group و 5 برای دسته ی Other می باشد .

اگر 6 دسیمال را به باینری تبدیل کنیم نتیجه می شود : 110
اگر 5 دسیمال را به باینری تبدیل کنیم نتیجه می شود : 101 چرا که :

101 => (1 x 2^**2**) + (0 x 2^**1**) + (1 x 2^**0**) = 4 + 0 + 1 = 5

اما در نمایش پرمیشن فایل ها و دایرکتوری ها نمی توان از این اعداد استفاده کرد چرا که بسیار گیج کننده است . برای این منظور از نوع دیگری از نمایش استفاده می کنند که اصطلاحا به آن Symbolic می گویند . برای مثال :

```
iman@ubuntu:~/test$ ls -l text
-rw-rw-r-- 1 iman iman 10 Jul  2 17:52 text
```

و یا :

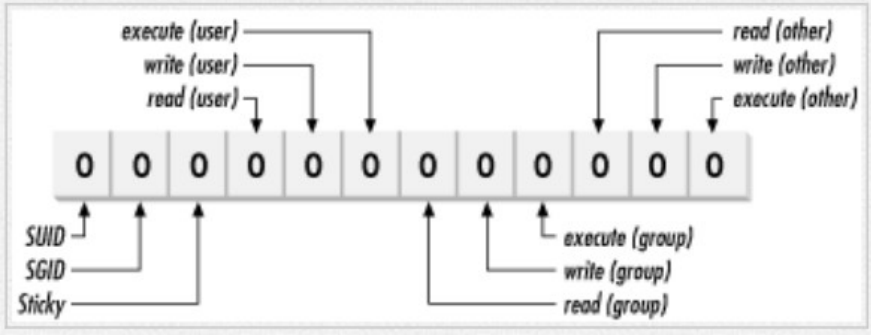
```
drwxrwxr-x 2 iman iman  4096 Jul  2 19:44 Dir
```

توجه داشته باشید d اول ستون پرمیشن مشخصه ی دایرکتوری بودن می باشد . حال به پرمیشن دقت کنید : rwxrwxr-x و یا rwx/rwx/r-x و یا :

دسته ی rwx : Owner یعنی این دسته قابلیت read و write و execute را دارد . توجه داشته باشید که execute را با x نمایش می دهند .
دسته ی rwx : Group یعنی این دسته قابلیت read و write و execute را دارد .
دسته ی r-x : Other یعنی این دسته قابلیت read را دارد . قابلیت write را ندارد و قابلیت execute را دارد . توجه داشته باشید که - نمایانگر نداشتن پرمیشن write می باشد .

ابتدا گفته شد فرض کنید که پرمیشن ها در لینوکس با 9 بیت تعیین می شود . در واقع پرمیشن ها با 12 بیت مشخص می شود یعنی یک دسته ی 3 تایی دیگر در ابتدا داریم اما کمتر با آن سر و کار داریم و معمولا هم متوجه حضور آن نمی شویم . برای مثال در خروجی دستور ls -l فقط 9 بیتی که در بالا صحبت کردیم را مشاهده خواهیم کرد .

تصویر دوم



اما این 3 بیت چی هستند و چه کارایی ای دارند :

این دسته ی 3 تایی یا 3 بیتی از 3 خانه به ترتیب با نام های suid و sgid و sticky تشکیل شده است :

1. وظیفه ی suid این است که در صورت فعال بودن بر روی فایلی . آن فایل را با پرمیشن سازنده یا همان Owner فایل اجرا می کند مثلا یک فایل اجرایی را فرض کنید که مقدار suid دارد و سازنده ی این فایل اجرایی root است . حال اگر یوزر دیگری (limit) این فایل اجرایی را اجرا کند (در صورت داشتن پرمیشن execute) . آن فایل با یوزر root به اجرا در می آید .

2. وظیفه ی sgid دقیقا همانند وظیفه ی suid است با تفاوت که با گروه و یا Group فایل سرو کار دارد . یعنی در صورت فعال بودن بر روی فایلی . در صورت اجرا توسط هر یوزر . فایل با گروه سازنده ی فایل به اجرا در می آید .

3. و در آخر هم sticky . ابتدا باید گفت که این مقدار تنها بر روی دایرکتوری می تواند ست شود و در صورت فعال بودن به یوزر های مختلف اجازه ی پاک کردن فایل ها و دایرکتوری های یک دیگر را نمی دهد . حتی اگر دو یوزر در یک گروه بوده و پرمیشن مربوطه را داشته باشد .

نکته : بر خلاف قسمت قبل که هم فایل و هم دایرکتوری می توانستند پرمشین بگیرند . فایل های اجرایی یعنی execute فقط می توان دو مقدار suid و sgid را بگیرند هم چنین دایرکتوری ها فقط می توانند دو مقدار sticky و sgid را به خود اختصاص دهند که مقدار این 3 خانه یعنی مقدار suid و sgid و sticky صفر و یک است . به طور خلاصه :

suid : فقط exe
sgid : هم exe و هم dir
sticky : فقط dir

مثال : فایلی اجرایی دارای پرمیشن 4775 است . 3 بیت اولیه پرمیشن این فایل را تفسیر کنید !
حل : از سمت چپ به راست شروع به خواندن می کنیم . پس در نتیجه اولین عدد 4 است . این عدد 4 مشخصه 3 بیت اولیه از 12 بیت است یا نمایانگر دسته ی 3 تایی اول است .

گفتیم که 3 بیت اولیه به ترتیب 3 مقدار suid و sgid و sticky را شامل می شود . هم چنین عدد دسیمال (ده دهی) این دسته هم 4 است .

پس می توان نتیجه گرفت باینری (دو دویی) این دسته به این صورت است : 100 چرا که :

100 = (1 x 2^**2**) + (0 x 2^**1**) + (0 x 2^**0**) = 4 + 0 + 0 = 4

هم چنین گفتیم که 0 یعنی عدم حضور سیگنال یا همان غیر فعال بودن و 1 یعنی حضور سیگنال یا همان فعال بودن پس در نتیجه 100 یا 1/0/0 یعنی suid فعال و sgid و sticky غیر فعال می باشد .

تصویر سوم

Working with SUID, SGID, and Sticky Bit

Permission	Numerical Value	Relative Value	On Files	On Directories
SUID	4	u+s	User executes file with permissions of file owner.	No meaning.
SGID	2	g+s	User executes file with permissions of group owner.	File created in directory gets the same group owner.
Sticky bit	1	+t	No meaning.	Users are prevented from deleting files from other users.

اما این 3 بیت اولیه کجا استفاده دارد ؟

ابتدا suid را فرض کنید . گفتیم اگر فایلی اجرایی این پرمیشن را داشته باشد , هر یوزری که این فایل را اجرا کنید , آن فایل با یوزر سازنده و یا Owner به اجرا در می آید . حال دستور passwd را در نظر بگیرید (با این دستور هر یوزر می تواند پسورد خود را عوض کند) این برنامه پرمیشن suid دارد پس هر یوزری که این برنامه را اجرا کنید , این برنامه با سطح دسترسی یوزر سازنده ی برنامه یعنی root به اجرا در می آید . پس در نتیجه یک یوزر محدود به این صورت می تواند تغییرات لازم را در فایل passwd و shadow اعمال کند .

When we try to change our password we will use passwd command, which is owned by root. This passwd command file will try to edit some system config files such as /etc/passwd, /etc/shadow etc when we try to change our password. Some of these files cannot be opened or viewed by normal user only root user will have permissions. So if we try to remove SUID and give full permissions to this passwd command file it cannot open other files such as /etc/shadow file to update the changes and we will get permission denied error or some other error when tried to execute passwd command. So passwd command is set with SUID to give root user permissions to normal user so that it can update /etc/shadow and other files.

به عنوان مثالی دیگر می توان دستور ping را مثال زد .

پرمیشن دهی به فایل و دایرکتوری :

برای پرمیشن دادن به یک فایل و یا یک دایرکتوری از دستور chmod استفاده می کنیم . ابتدا به Syntax این دستور توجه کنید :

```
chmod [OPTION]... MODE[,MODE]... FILE...
chmod [OPTION]... OCTAL-MODE FILE...
chmod [OPTION]... --reference=RFILE FILE...
```

همان طور که از Syntax پیداست , به 3 روش می توان پرمیشن یک فایل را تعیین کرد :

1. استفاده از حروف برای مشخص کردن پرمیشن .
2. استفاده از اعداد اکتال برای مشخص کردن پرمیشن .

3. رفرنس به فایلی دیگر برای مشخص کردن پرمیشن .

روش اول :

در این روش دقیقا پرمیشنی که لازم است به فایل داده شود یا گرفته شود باید ذکر شود . برای مثال :

```
$chmod u+x file1
```

توجه داشته باشید که u مشخصه ی دسته ی Owner و x مشخصه ی execute می باشد . معنی این دستور یعنی فایل file1 برای سازنده پرمیشن اجرا داشته باشد (اضافه شود) .

مثالی دیگر :

```
$chmod g-r file1
```

این دستور یعنی از دسته ی Group پرمیشن write گرفته شود . توجه داشته باشید که علامت - به معنای گرفتن پرمیشن می باشد . مثال :

```
$chmod gu+r file1
$chmod o-rw file1
```

توجه داشته باشید که منظور از o دسته ی Other می باشد . مثال :

```
$chmod g-rw , u+w , o-rwx file1
```

مطابق Syntax می توان چندین mode را مشخص کرد . مثال :

```
$chmod o=rw file1
```

دستور بالا به این معناست که دسته ی Other فقط و فقط پرمیشن read و write داشته باشد . مثال :

```
$chmod ug=r file1
```

روش دوم :

مطابق Syntax در این روش از اعداد اکتال برای مشخص کردن پرمیشن یک یا چند فایل استفاده می کنیم . برای مثال قصد داریم پرمیشن 644 را به فایل file1 بدهیم :

```
$chmod 644 file1
```

روش سوم :

مطابق Syntax و تعریف , در این روش پرمیشن یک فایل از پرمیشن فایلی دیگر خوانده می شود . به عنوان مثال قصد داریم پرمیشن فایل file1 دقیقا برابر پرمیشن فایل shadow شود :

```
$chmod --reference=/etc/passwd file1
```

توجه داشته باشید که مشخص کردن پرمیشن دایرکتوری نیز به همین صورت است . به عنوان مثال :

```
$chmod 775 dir1/
```

نکته : برای تغییر پرمیشن تمامی فایل های درون یک دایرکتوری از سوییچ R- استفاده می کنیم . به عنوان مثال :

```
$chmod -Rv o-w dir1/
```

نکته : تا به این جا فقط تعیین 9 بیت پرمیشن معروف گفته شد . در واقع برای مشخص کردن پرمیشن در 3 بیت اول که شامل suid و sgid و sticky هست , دوباره به 3 روش می توان عمل کرد :

1. استفاده از حروف برای مشخص کردن پرمیشن .
2. استفاده از اعداد اکتال برای مشخص کردن پرمیشن .
3. رفرنس از یک فایل برای مشخص کردن پرمیشن .

روش اول :

ابتدا باید گفت که :

الف) به صورت خلاصه suid را با s و gsid را با s و sticky را با t مشخص می کنند .
ب) تغییرات suid در دسته ی Owner و تغییرات gsid در دسته ی Group و تغییرات sticky در دسته ی Other اعمال می شود .
ج) تغییرات suid و guid و sticky در پرمیشن x یا همان execute دسته ی خود اعمال می شود . یعنی suid در پرمیشن x دسته ی Owner و guid در پرمیشن x دسته ی Group و sticky در پرمیشن x دسته ی Other تغییر اعمال خواهد کرد .

حال به مثال زیر توجه کنید :

```
$chmod u+s file1
```

دستور بالا به این معناست که به فایل file1 پرمیشن suid داده شود . توجه داشته باشید که s به معنای suid است چرا که u به معنای دسته ی Owner می باشد . حال به مثال زیر توجه کنید :

```
$chmod g+s dir1/
```

g به معنای دسته ی Group است و از آن جایی که در این دسته فقط gsid اعمال می شود . پس نتیجه می گیریم که s در این جا به معنای gsid است نه usid . حال به مثالی دیگر توجه کنید :

```
$chmod o-t dir1/
```

همان طور که در ابتدا گفته شد , t به معنای sticky و o به معنای دسته ی Other می باشد .

روش دوم :

در این روش از اعداد اکتال برای مشخص کردن پرمیشن استفاده می کنیم . در قسمتی که 9 بیت دوم رو بررسی کردیم دیدیم که پرمیشن ها به صورت 3 عدد اکتال 1 رقمی داده می شد . در این جا یک عدد اکتال دیگر به این 3 عدد اضافه شده و به عنوان پرمیشن به فایل یا دایرکتوری تخصیص داده می شود . برای مثال :

```
$chmod 6750 file1
```

توجه داشته باشید که اولین عدد از سمت چپ نمانگر پرمیشن برای دسته اول یا همان 3 بیت اول می باشد که در این مثال 6 است یعنی :

110 = (1 x 2^**2**) + (1 x 2^**1**) + (0 x 2^**0**) = 4 + 2 + 0 = 6

با توجه به تصویر دوم . عدد اکتال 6 یعنی پرمیشن suid و sgid به فایل file1 داده شده است .

طبیعتا اگر مثل بالا حساب کنیم , می توانیم به این نتیجه برسیم که عدد 4 مشخصه ی پرمیشن suid و عدد 1 مشخصه ی پرمیشن sticky می باشد .

توجه داشته باشید که در مثال بالا اگر به جای 6750 , 750 را وارد می کردیم , 3 بیت اول یعنی suid و sgid و sticky هیچ تغییری نمی کند و مقدار پیش فرض خود را که 0 (غیر فعال) است حفظ می کند .

روش سوم

روش سوم با استفاد از رفرنس است و همانند قسمت سوم 9 بیت دوم (قسمت قبل) می باشد .

Displaying File's Information ....

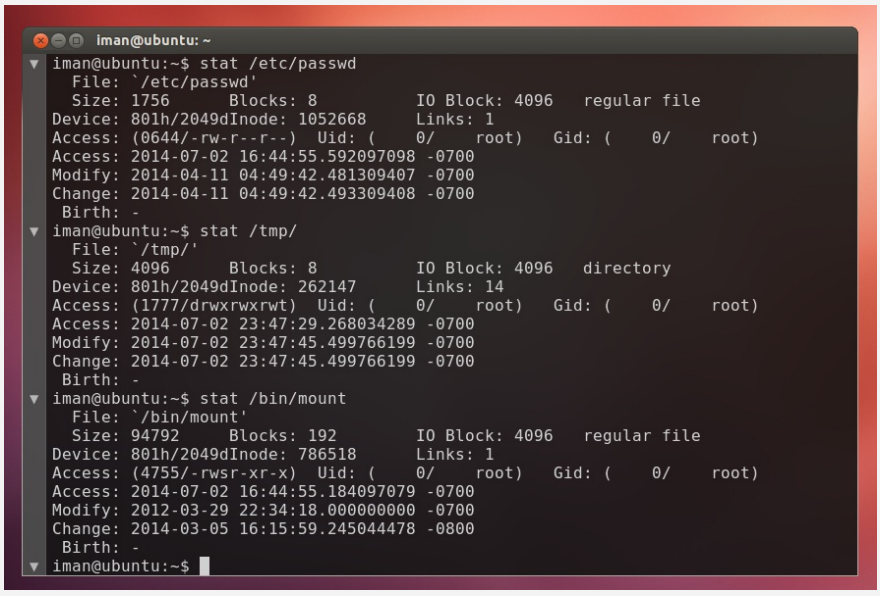
دستور stat :

```
stat [OPTION]... FILE...
```

در واقع این دستور از کلمه ی status میاید و جزییاتی از فایل و یا دایرکتوری را برای ما نمایش می دهد . نمایش دقیق پرمیشن یکی از قابلیت های این دستور می باشد که ما در این جا از آن استفاده می کنیم . به عنوان مثال :

```
$stat /bin/mount
$stat /usr/bin/passwd
$stat /tmp
```

حالا به تصویر زیر نگاه کنید (به قسمت access و عدد اکتال پرمیشن ها توجه کنید)



همان طور که در تصویر می بینید به ترتیب 3 عدد 0644 و 1777 و 4755 را در پیش رو داریم . 0 به معنای 0 بودن یا غیر فعال بودن تمامی خانه های 3 بایت اول می باشد . 1 به معنای 2 به توان صفر یعنی فعال بودن پرمیشن sticky می باشد . 4 به معنای 2 به توان 2 و فعال بودن پرم suid می باشد .

نکته : /tmp مسیری است که تمامی برنامه ها فایل های موقت خود را در آن می ریزند لذا پیمشن آن 777 است اما برای جلوگیری از این مورد که کسی نتواند فایل دیگری را پاک کند , پرمیشن sticky هم در این دایرکتوری فعال می باشد .

Setting umask ....

```
umask [-S] [mask definition]
```

در قسمت های قبل دیدیم که چه طور پرمیشن فایل و یا دایرکتوری های مختلف را کنترل و تعیین کنیم . اما پرمیشن اولیه فایل ها و دایرکتوری هایی که ساخته می شوند به صورت پیش فرض چی هست ؟

برای تعیین پرمیشن فایل ها و دایرکتوری ها به صورت خودکار با تعریفی به نام umask سرو کار داریم که تقریبا به صورت یک مکانیزم امنیتی عمل می کند .

اگر پرمیشن فایل را 666 و پرمیشن دایرکتوری ها را 777 در نظر بگیریم , پرمیشن فایل ها و دایکتوری هایی که ساخته می شوند (به صورت پیش فرض) برابر با :

666 - umask = file's permission
777 - umask = dir's permission

برای مثال در اوبونتو 12.04 این عدد 0002 می باشد و اگر از دو عدد 666 و 777 کم شود به ترتیب دو عدد 664 و 775 بدست می آید که پرمیشن پیش فرض فایل ها و دایرکتوری های ساخته شده می باشد .

برای تغییر این عدد پیشفرض می توان به عنوان نمونه به صورت زیر عمل کرد :

```
umask 0022
```

نکته : این عدد در توزیع های مختلف متفاوت است .

Changing file's owner ....

نکته : در لینوکس هیچ کسی نمی توان owner یک فایل و یا دایرکتوری را تغییر دهد حتی اگر سازنده ی آن باشد . به غیر از یوزر Root

برای این منظور از دستور chown استفاده می کنیم . ابتدا به Syntax این دستور توجه کنید :

```
chown [OPTION]... [OWNER][:[GROUP]] FILE...
chown [OPTION]... --reference=RFILE FILE...
```

همان طور که از Syntax پیداست . از این دستور می توان به 2 روش استفاده کرد :

الف) مشخص کردن owner و group
ب) رفرنس دادن به فایلی دیگر

مثال :

```
#chown sara file1
#chown sara:test file1
#chown sara: file1
```

توجه داشـته باشـید که در قسـمت سوم , گروهی مشخص نشـده است لذا به صورت پیش فرض گروه دیفالت یوزر به فایل تخصیص داده می شود .

مثال :

```
#chown :test file1
#chown --reference=/etc/passwd file1
#chown -Rv sara dir1/
```

در قسمت اول هیچ یوزری در نظر گرفته نشـده است لذا در این حالت فقط گروه فایل file1 تغییر می کند .

توجه داشته باشید که در قسمت سوم از سوییچ R- برای تغییر owner کل دایرکتوری dir1 استفاده شده است .

Changing file's group owner ....

بر خلاف قسمت قبل که هیچ یوزری نمی توانست owner خود را تغییر دهد , هر یوزر می تواند Group Owner خود را تغییر دهید (بدون احتیاج به دسترسی root) . ابتدا به Syntax این قسمت توجه کنید :

```
chgrp [OPTION]... GROUP FILE...
chgrp [OPTION]... --reference=RFILE FILE...
```

به عنوان مثال :

```
$chgrp test file1
$chgrp -Rv test dir1/
```

منبع : پژوهشی دات آی آر | Pazhoheshi.ir

دانلود PDF این مقاله

هر گونه نظر و انتقاد رو در قالب نظر با من در میون بزارید .

E2MA3N ۹۳/۰۴/۱۲ ۰ ۰

lpic 1 101 | آموزش lpic | آموزش فارسی lpic | آموزش فارسی lpic 1 | اموزش لینوکس | دستورات لینوکس

لینوکس LPIC 1 101

نظرات (۰)

هیچ نظری هنوز ثبت نشده است

ارسال نظر به صورت ناشناس

نام * E2MA3N

پست الکترونیک e2ma3n@gmail.com

سایت یا وبلاگ pazhoheshi.ir

پیام *

کد امنیتی *

نظر بصورت خصوصی ارسال شود

پست الکترونیک برای عموم قابل مشاهده باشد

ارسال

بلاگ بیان، رسانه متخصصان و اهل قلم