در این مقاله قصد داریم به بیان نکاتی در مورد کار کردن با کاربران لینوکس از لیست کاربران و جمع‌آوری اطلاعات در مورد آن‌ها گرفته تا اضافه و حذف کردن کاربران در لینوکس و تغییر رمز آن‌ها بپردازیم. در هر بخش فایل‌های استفاده شده برای نگهداری تنظیمات و دستورات کار با فایل‌ها را معرفی کرده و با ذکر مثال‌هایی به بررسی آن‌ها خواهیم پرداخت.

دو فایلی که برای نگهداری اطلاعات کاربران و همچنین رمز آن‌ها (به صورت کد شده) استفاده می‌شوند، یکی /etc/passwd بوده که اطلاعات کاربران از جمله نام کاربر، آدرس home آن، shell پیش فرض کاربر را نگهداری می‌کند. در نسخه‌های قدیمی لینوکس رمز کاربر نیز در این فایل نگهداری می‌شد ولی در نسخه‌هایی که الان استفاده می‌شود، فیلد دوم این فایل x بوده و بیانگر این نکته است که اطلاعات رمز کاربر در فایل دیگری نگهداری می‌شود. این فایل /etc/shadow می‌باشد که کد رمز به همراه اطلاعات دیگری از جمله آخرین باری که رمز کاربر تغییر کرده، مدت زمانی که رمز می‌تواند استفاده شود و بعد از آن باید تغییر کند، حداقل مدت زمان نگهداری رمز، در آن نگهداری می‌شود.

نکته: دقت شود که کد رمز به صورت hash بوده و به صورت encrypt نمی‌باشد و معنی این موضوع این است که قابلیت پیدا کردن رمز از روی این کد تنها به صورت brute force وجود داشته و این کد بازگشت پذیر نمی‌باشد.

در این فایل‌ها هر سطر بیانگر اطلاعات یک کاربر می‌باشد. هر سطر به ستون‌هایی تقسیم می‌شود که با : از یکدیگر جدا شده و هر ستون اطلاعات خاصی در مورد کاربر را در خود نگه می‌دارد. برای مشاهده کردن اطلاعات این دو فایل می‌توان به صفحه‌ی ۵ از manual لینوکس مراجعه نمود.

man 5 passwd

man 5 shadow

به عنوان مثال اگر خط اول فایل passwd را مشاهده کنیم و با اطلاعات مشاهده شده در manual مقایسه کنیم به اطلاعات زیر خواهیم رسید.

```
u1dblog@ubudblog:~$ head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

محیط shell : مسیر home : اطلاعات تکمیلی : شناسه‌گروه : شناسه‌کاربر : کد رمز : نام‌کاربر

همچنین با مشاهده کردن خط اول فایل shadow می‌توان اطلاعاتی در مورد رمز کاربر بدست آورد که از چپ به راست برخی آن‌ها به صورت زیر می‌باشند.

```
u1dblog@ubudblog:~$ sudo head -1 /etc/shadow
root:!:16961:0:99999:7:::
```

..... حداکثر طول عمر رمز : حداقل طول عمر رمز : زمان آخرین تغییر رمز : کد رمز : نام کاربر

در عکس‌های مربوط به دو فایل معرفی شده چند نکته حائز اهمیت است. اولین نکته اینکه زمان آخرین تغییر رمز کاربر به صورت یک عدد بیان شده است و نه به صورت یک تاریخ کامل. این عدد تعداد روز گذشته از ۱ ژانویه سال ۱۹۷۰ را مشخص می‌کند و با محاسبه زمانی که از این تاریخ می‌گذرد می‌توان به تاریخ دقیق تغییر رمز پی برد.

نکته‌ی بعدی مربوط به نوع کاربران و مجوزهای آن‌ها در لینوکس می‌باشد. اگر به دستوراتی که برای نمایش خط اول فایل‌ها استفاده شده‌اند دقت نمایید متوجه خواهید شد که برای نمایش اطلاعات از فایل shadow از دستور sudo استفاده شده است. در صورت استفاده از sudo در ابتدای اجرای دستور، آن دستور با مجوزهای کاربر root اجرا می‌شود که در لینوکس مدیر سیستم بوده و توانایی اجرای هر دستوری و اعمال هر تغییری در سیستم را دارد. کاربران عادی توانایی اجرای دستوراتی که تغییری در سیستم بوجود می‌آورند و یا حتی دسترسی و خواندن اطلاعات فایل‌هایی که مالک آن‌ها کاربر root می‌باشد را ندارند. اما در شرایطی که کاربری عضو گروه sudo باشد (یا اجازه‌ی استفاده از دستورات برای آن در فایل /etc/sudoers تنظیم شده باشد) می‌تواند مجوزهای خود را بالاتر برده و به استفاده از دستوری بپردازد که در شرایط عادی امکان استفاده از آن را ندارد. عضویت در گروه sudo (که البته برای آن تنظیمی در فایل /etc/sudoers انجام گرفته که کار ساده‌تر شده و با عضویت در یک گروه بجای تغییر در فایل امکان استفاده از مجوزهای root فراهم شود) توانایی استفاده از دستور sudo را به کاربران خواهد داد که command نوشته شده را با مجوزهای root اجرا خواهد نمود.

اکنون که دلیل استفاده از sudo را متوجه شدیم به بررسی مجوزهای دو فایل و مقایسه‌ی آن‌ها با یکدیگر می‌پردازیم که دلیل نیاز به استفاده از sudo برای خواندن محتویات فایل shadow را متوجه شویم.

```
u1dblog@ubudblog:~$ ls -l /etc/passwd /etc/shadow
-rw-r--r-- 1 root root   2338 Dec 30 17:12 /etc/passwd
-rw-r----- 1 root shadow 1497 Dec 30 17:31 /etc/shadow
```

در خروجی ls -l ستون‌های نمایش داده شده (از چپ به راست) به ترتیب مجوزهای فایل، تعداد لینک‌ها، کاربر مالک فایل، گروه مالک فایل، اندازه‌ی فایل به بایت، زمان آخرین تغییر فایل و در نهایت نام فایل آورده می‌شود.

مجوزهای فایل (به همراه نوع فایل) یک رشته‌ی ۱۰ کاراکتری می‌باشد که اولین کاراکتر آن (از چپ) نوع فایل را مشخص نموده و ۹تای باقی‌مانده مجوزهای دسترسی به فایل توسط کاربر مالک، گروه مالک و مابقی افراد می‌باشد. نحوه‌ی تعیین مجوز ها با سه کاراکتر rwx برای خواندن، نوشتن و اجرا کردن فایل و یا – برای عدم وجود مجوز می‌باشد. در صورتیکه به خروجی دقت نمایید، متوجه می‌شوید که مالک این دو فایل کاربر root بوده و دیگران (بخش سوم مجوزها) اجازه‌ی خواندن passwd را داشته (با مجوز r--) ولی این مجوز را برای فایل shadow ندارند که دلیل استفاده از sudo برای خواندن خط اول فایل می‌باشد، که به این صورت باعث اجرای دستور head (برای خواندن یک خط از ابتدای فایل) با مجوزهای کاربر root و امکان خواندن فایل می‌شود.

نکته‌ی آخری که در خروجی خطوط اول فایل‌ها وجود دارد این است که کد رمزی برای کاربر root وجود نداشته و بجای آن از ! استفاده شده است. در لینوکس در صورتیکه نیاز به منع کاربری از ورودی به سیستم باشد، می‌توان بجای رمز آن (یا حتی در صورت داشتن رمز، در ابتدای رشته‌ی کد رمز) از ! استفاده نموده و امکان ورود به سیستم را از کاربر گرفت.

کاربر root به دلیل مسائل امنیتی از ورود به سیستم منع شده و در صورت انتساب رمز نیز ممکن است امکان ورود به سیستم را نداشته باشد. در صورت نیاز به استفاده از آن (هر چند توصیه نمی‌شود) می‌توان امکان ورود به سیستم از طریق shell با کاربر root را توسط دو مرحله‌ی زیر فعال نمود.

$sudo passwd root

۱) برای انتساب رمز به کاربر root

$sudo passwd –u root

۲) برای unlock کردن کاربر root

در صورتیکه این کاربر را فعال کرده و بعد از مدتی مجددا تصمیم به غیر فعال کردن آن گرفته شود می‌توان با استفاده از l– بجای u– در دستور passwd اینکار را انجام داد. غیر فعال کردن root و تغییری که در محتویات فایلshadow ایجاد می‌شود را می‌توان در شکل زیر مشاهده نمود.

```
u1dblog@ubudblog:~$ sudo head -1 /etc/shadow
root:$6$0aBdnVPl$hnJCWLbVL10V134zdf3cUmQC/DDIcbXi3/6czfKQQPk9PSweGpl9zl3XLdPUD74pS6blVH81jYd3l6BgC/x3P.:
17165:0:99999:7:::
u1dblog@ubudblog:~$ sudo passwd -l root
passwd: password expiry information changed.
u1dblog@ubudblog:~$ sudo head -1 /etc/shadow
root:!$6$0aBdnVPl$hnJCWLbVL10V134zdf3cUmQC/DDIcbXi3/6czfKQQPk9PSweGpl9zl3XLdPUD74pS6blVH81jYd3l6BgC/x3P.
:17165:0:99999:7:::
```

اکنون که با نحوه‌ی ذخیره‌ی اطلاعات کاربران در لینوکس آشنا شده‌ایم به ساخت کاربر جدید پرداخته و نکاتی که بیان شد را مرور کرده و چند نکته‌ی مهم دیگر را نیز بیان خواهیم نمود.

دستوری که برای ساخت کاربر در لینوکس استفاده می‌شود، useradd می‌باشد. این دستور پارامترهای مختلفی دریافت کرده و اطلاعات کاربر را در فایل passwd ذخیره خواهد نمود. برای تست کاربر جدید نیز می‌توانیم با زدن دستور su –l یک ورود جدید به لینوکس انجام داده و کاربر جدید را تست نمود. البته دقت نمایید که باید رمزی برای کاربر تنظیم کنید تا بتوانید با آن وارد شوید. در شکل زیر ساخت و تست کاربر جدید نمایش داده شده است.

```
u1dblog@ubudblog:~$ sudo useradd dbloguser
u1dblog@ubudblog:~$ sudo grep dbloguser /etc/shadow
dbloguser:!:17165:0:99999:7:::
u1dblog@ubudblog:~$ sudo passwd dbloguser
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
u1dblog@ubudblog:~$ sudo grep dbloguser /etc/shadow
dbloguser:$6$JKdAITGv$ouddYsFbFcpdLMKjR6gs56egXaN/cV9J0aSv1j84MGswp1SOw91YOXBlWvMEcNwV/vETgXb.AV8IceN8SM
nzq1:17165:0:99999:7:::
u1dblog@ubudblog:~$ su -l dbloguser
Password:
No directory, logging in with HOME=/
$ whoami
dbloguser
$
```

در این شکل اعلانی که برای کاربر مشخص شده را با یک $ تنها مشاهده می‌کنید (دستور whoami استفاده شده که نشان دهد با کاربر جدید وارد شده‌ایم) که این اعلان بر خلاف چیزی است که در لینوکس به صورت پیش فرض وجود دارد. اعلان پیش فرض لینوکس (حداقل در Ubuntu) به صورت زیر می‌باشد:

username@hostname:path$

که در صورت ورود با کاربر root بجای $ یک # نمایش داده خواهد شد.

اگر نگاهی به خط مربوط به کاربر جدید در فایل passwd بیاندازید می‌بینید که برای این کاربر home تنظیم شده ولی ساخته نشده است و shell پیش فرض نیز ندارد.

```
$ grep dbloguser /etc/passwd
dbloguser:x:1002:1002::/home/dbloguser:
$ ls /home/dbloguser
ls: cannot access '/home/dbloguser': No such file or directory
$
```

تاثیر shell را می‌توان در نحوه‌ی اعلان مشاهده نمود و تاثیر home را در مواردی مثل سابقه‌ی دستورات. به این شکل که اگر دستوراتی را تایپ کرده، خارج شده و مجدد وارد شوید، هیچ سابقه‌ای از دستورات نشست قبل نخواهید داشت. علت این است که محیط کاربر از طریق home تنظیم می‌شود و سابقه‌ی دستورات که مثال زدم در فایلی به نام .bash_history که در home می‌باشد ذخیره می‌گردد. اگر دستور useradd با پارامترهای –m و –s استفاده شود، هم home ساخته شده و هم shell تنظیم می‌گردد. اما برای کاربر جدید که ساخته‌شده است چطور می‌توان home را ساخته و shell را تنظیم نمود؟ تغییر در ساختار تنظیمات کاربر با usermod امکان‌پذیر می‌باشد که پارامترهایی مشابه useradd دارد (مثل –s که در هر دو وجود دارد) و برای ساخت home نیز می‌توان از یک دستور کمکی به نام mkhomedir_helper استفاده نمود.

نکته: در صورت استفاده از دستور adduser بجای useradd برای ساخت کاربر، home ساخته شده و shell نیز تنظیم می‌گردد.

```
u1dblog@ubudblog:~$ sudo usermod -s /bin/bash dbloguser
u1dblog@ubudblog:~$ sudo mkhomedir_helper dbloguser
u1dblog@ubudblog:~$ ls -ld /home/dbloguser/
drwxr-xr-x 2 dbloguser dbloguser 4096 Dec 30 18:45 /home/dbloguser/
u1dblog@ubudblog:~$ su -l dbloguser
Password:
dbloguser@ubudblog:~$
```

کاربر جدید اکنون می‌تواند مشابه یک کاربر عادی به کار در سیستم بپردازد ولی هنوز یک مساله برای این کاربر وجود دارد. اگر لیست گروه‌های این کاربر را با دستور groups مشاهده کنید، می‌بینید که گروه sudo در آن نبوده و تاثیر را وقتی متوجه خواهید شد که مثلا به اجرای دستوری به کمک sudo بپردازید.

```
dbloguser@ubudblog:~$ groups
dbloguser
dbloguser@ubudblog:~$ sudo head -1 /etc/shadow
[sudo] password for dbloguser:
dbloguser is not in the sudoers file.  This incident will be reported.
dbloguser@ubudblog:~$
```

پیغامی که مشاهده می‌کنید در مورد نبود این کاربر در فایل sudoers می‌باشد. در صورتیکه از کاربر اصلی سیستم که اجازه‌ی استفاده از دستور sudo را دارد استفاده کرده و با usermod گروه sudo را به گروه‌های کاربر جدید اضافه کنید (با aG– که برای اضافه کردن گروه به لیست گروه‌های فعلی است) این مشکل برطرف شده و امکان استفاده از دستور sudo فراهم خواهد شد.

```
u1dblog@ubudblog:~$ sudo usermod -aG sudo dbloguser
u1dblog@ubudblog:~$ su -l dbloguser
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

dbloguser@ubudblog:~$ sudo head -1 /etc/shadow
[sudo] password for dbloguser:
root:!$6$0aBdnVPl$hnJCWLbVL10V134zdf3cUmQC/DDIcbXi3/6czfKQQPk9PSweGpl9zl3XLdPUD74pS6blVH81jYd3l6BgC/x3P.
:17165:0:99999:7:::
dbloguser@ubudblog:~$
```

مبحث آخری که در این مقاله به آن خواهیم پرداخت نحوه‌ی تولید کد از روی رمز کاربر می‌باشد. فیلد دوم در فایل shadow که حاوی کد رمز می‌باشد، به صورت زیر تعیین می‌شود.

$algorithm_id$salt_string$encrypted_password

الگوریتم‌های مختلفی می‌توانند برای تولید کد از روی رمز استفاده شوند (مثل md5 یا sha) و هر الگوریتم می‌تواند تعداد بیت مختلفی استفاده کرده و کد رمز با طول متفاوتی تولید نماید. در نسخه‌های فعلی لینوکس (Ubuntu 16 در مثال‌های این مقاله شده) الگوریتم پیش فرض SHA-512 می‌باشد. برای شناسایی الگوریتم استفاده شده در تولید کد رمز و مقایسه‌ی رمز وارد شده با رمز صحیح و احرازهویت کاربران، شناسه‌ای برای مشخص نمودن الگوریتم در این بخش نگهداری می‌شود که برای الگوریتم SHA-512 (الگوریتم پیش فرض) ۶ می‌باشد. برای اطلاع از الگوریتم‌های مخلف و شناسه‌ی آن‌ها می‌توان از صفحه ۳ فایل‌های کمکی مربوط به crypt با دستور man 3 crypt استفاده نمود. روش کار به این صورت است که در زمان ورود کاربر، با خواندن این فایل و بدست آوردن الگوریتم استفاده شده، کد رمزی از روی رمز وارد شده توسط کاربر تولید می‌شود که با کد رمز ذخیره شده در این فایل مقایسه شده و در صورت یکسان بودن، به کاربر اجازه‌ی ورود داده می‌شود. تا اینجای کار با دو بخش از سه بخش جدا شده با $ در فیلد دوم فایل shadow آشنا شده‌ایم و یک بخش که رشته‌ی salt می‌باشد باقی مانده است. برای درک کاربرد آن فرض کنید که این رشته استفاده نشده و دو کاربر رمز یکسانی داشته باشند. به دلیل اینکه الگوریتم‌های تولید hash از روی رمز، در هر بار استفاده، رشته‌ی یکسانی به عنوان کد رمز تولید می‌کنند، در فایل shadow مقدار یکسانی برای کاربرانی که رمز یکسانی دارند خواهیم داشت که ممکن است به لو رفتن رمز کاربران منجر شود و در شکستن رمزها نیز استفاده خواهد شد. حال اگر تابع تولید hash بجای دریافت یک مقدار رمز، دو مقدار دریافت نماید و مقدار دوم یک رشته‌ی تصادفی بوده و در هر باز عوض کردن رمز مقدار جدیدی خواهد داشت، دیگر برای رمزهای یکسان نیز مقدار کد متفاوتی ذخیره شده و مشکلی پیش نخواهد آمد. این تعریف و کاربرد رشته‌ی salt است

که یک رشته‌ی تصادفی بین ۸ تا ۱۶ کاراکتر بوده و به عنوان ورودی دوم در کنار رمز به تابع hash ارسال می‌شود تا کد رمز تولیدی هر دفعه تغییر کرده و با دفعه قبل و برای کاربران مختلف یکسان نباشد.

برای تولید کد رمز با ارسال salt و رمز می‌توانید به صورت زیر از دستور mkpasswd استفاده نمایید. (برای نصب آن در صورت عدم وجود از دستور sudo apt-get install whois استفاده کنید)

```
u1dblog@ubudblog:~$ mkpasswd -m help
Available methods:
des      standard 56 bit DES-based crypt(3)
md5      MD5
sha-256 SHA-256
sha-512 SHA-512
u1dblog@ubudblog:~$ mkpasswd -S 0aBdnVPl -m sha-512 12
$6$0aBdnVPl$hnJCWLbVL10V134zdf3cUmQC/DDIcbXi3/6czfKQQPk9PSweGpl9zl3XLdPUD74pS6blVH81jYd3l6BgC/x3P.
u1dblog@ubudblog:~$ sudo head -1 /etc/shadow | cut -d : -f 2
$6$0aBdnVPl$hnJCWLbVL10V134zdf3cUmQC/DDIcbXi3/6czfKQQPk9PSweGpl9zl3XLdPUD74pS6blVH81jYd3l6BgC/x3P.
u1dblog@ubudblog:~$
```

با دقت در شکل بالا می‌بینید که اگر salt ذخیره شده در فایل shadow را استفاده کرده و رمز کاربر root (که من ۱۲ گذاشته بودم☺) را بکار بریم، دقیقا همان مقداری که در فیلد دوم shadow ذخیره شده است را بدست خواهیم آورد.

در نهایت برای حذف کاربر از دستور userdel استفاده کنید. در صورتیکه بخواهید فایل‌های موجود در home نیز به همراه کاربر حذف گردند می‌توانید از r– به همراه دستور استفاده نماید.

امیدوارم مفید بوده باشه، موفق باشید.