

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ctr+alt+f1 وارد محیط اصلی لینوکس می‌شوید

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

passwd : پسورد هر کاربر را میتوانید عوض کنید اگر تنها passwd را وارد کنید پسورد کاربری که با آن لاگین هستید عوض می‌شود اما اگر همراه با username وارد شود (passwd user\_name) پسورد کاربر مورد نظرتان عوض می‌شود

=====================================================================================

دستورات مقدماتی :

shell : به معنی پوسته میباشد که از یه هسته (صدف) مراقبت میکند . من اگر بخواهم یک دستور را به هسته ی سیستم عامل ارسال کنم بایستی از طریق یک رابط کاربری این کار را انجام دهم در اینجا منظور از رابط کاربری shell میباشد در سیستم عامل های مختلف shell های مختلف وجود دارد bash معروفترین shell در خانواده ی linux میباشد

command : دستوراتی هست که از طریق shell به هسته ارسال میکنیم از دستورات میتوانند option یا arguments داشته باشند

نکته مهمی که در سیستم عامل لینوکس باید به آن توجه کرد این است که بین حروف بزرگ و کوچک تفاوت میگذارد

در ترمینال قبل از @ نام کاربری شما که با آن login کرده‌اید و بعد از @ اسم کامپیوتر شما میباشد

اگر به عنوان کاربر root لاگین کرده باشید در انتها علامت # را مشاهده میکنید و اگر به عنوان کاربر معمولی login کرده باشید در انتها علامت $ را مشاهده میکنیم

ls ====> list cd ====> change directory clear or ctrl+l ====> ترمینال را پاک میکند

ls -l ====> با جزییات بیشتر نمایش میدهد در قسمت اول سطوح دسترس ls \*.txt ls file??.txt ==>هر ؟ یک کاراکتر

cd ~ ====> در هر کجایی که باشید شماره به دایرکتوری هوم کاربر میبرد cat file00.txt file01.txt

با استفاده از دستور cat میتوانید محتویات فایل‌ها را مشاهده کنید همینطور مشاهده محتویات ۲ فایل به صورت همزمان

echo sajjad ==== > با استفاده از این دستور میتوانید یک خروجی دریافت کنید مثلاً سجاد خروجی این مثال است

توجه داشته باشید که فاصله (space) به معنی option بعدی میباشد و در دستور echo خروجی مناسب شما چاپ نشود

شما میتوانید ماننده زبان‌های برنامه نویسی به متغیر ها مقدار دهی کنید و متغیر ها را فرا خوانی کنید(بعد از علامت $)

a=sajjad

b=ahmad

echo $a ====> sajjad echo $a$b ====> sajjadahmad echo $a $b ===> sajjad ahmad

میتوانیم دستوراتی که وارد میکنید را در یک خط قرار دهیم اما خروجیشان در یک خط نباشد (با استفاده از ;)

echo $a; echo $b

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

توجه شود که متغیر هایی که تعریف می‌شوند با بسته شدن ترمینال از بین می‌روند

متغیر هایی وجود دارد که به صورت سراسری تعریف می‌شوند ماننده PATH که مسیری که دستورات در آن‌ها دخیره شده است را نشان میدهد

echo $PATH

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور زیر نشان میدهد که هر دستور (مثلاً ls ) در کدام پوشه ذخیره شده است

which ls

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور history تاریخچه دستوراتی که در ترمینال وارد شده است را نمایش میدهد

===================================================================================

چگونه از help استفاده کنیم ؟

help man info whatis whereis less

مکانیزم استفاده از دستور help =======> command - - help ======> ls - - help (برای echo نمیتوانیم از help استفاده کنیم )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور man =======> man command ======> man echo( توضیحات بیشتری نسبت به help دارد )

دستور man آپشنی دارد دارد که میتوانید با استفاده از آن در تمام manualها دنبال کلمه مورد نظرتان بگردید (در مثال زیر هر توضیحی که در آن کلمه ping وجود دارد نشان میدهد ) ==========================> man -k ping

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور info =======> info command ======> info echo( توضیحات بیشتری نسبت به man دارد )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور whatis به صورت یک خط به توضیح دستور مورد نظر میپردازد

whatis command =======> whatis ls

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور whereis محل ذخیره دستور را نشان میدهد

Whereis command =======> whereis echo

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

در لینوکس محلی وجود دارد که تمام doc یا مستنداتی که در لینوکس وجود دارد در آنجا ذخیره میشود ======> /usr/share/doc/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

همانطور که میدانیم با استفاده از دستور cat نمیتوانیم محتویات یک فایل زیپ شده )فشرده شده ) را ببینیم . دستور less این کار را برای ما انجام میدهد

less name.gz

=====================================================================================

استفاده از فایل و دایرکتوری ها

Cd (exact path , partially path)

. and .. (current directory , above directory)

~ (home directory) ~ ===> tilda

Pwd ( print working directory , present working directory)

./installer.pl cd .. cd /home/sajjad cd ~/desktop

cd خالی هم شمارا به home directory میبرد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ls

ls -a (all files )

ls -l (long detail listing )

ls -R (recursive view)

میتوان از سوییج ها (آپشن ها) به صورت ترکیبی استفاده کرد

در ls -l اگر اولین حرف d بود نشان دهنده ی یک directory میباشد

در ls -l اگر اولین حرف - بود نشان دهنده ی یک file میباشد (تمامی فایل ها )

اگر در اول اسم یک فایل یا فولدر نقطه (.) وجود داشت آن فایل یا فولدر مخفی میباشد

=====================================================================================

ایجاد کردن , منتقل کردن و حذف کردن فایل ها و فولدرها



\_ توجه شود که در محیط لینوکس دستوری برای rename کردن نداریم و برای این کار از دستور mv استفاده میکنیم

\_ اگر با استفاده از دستور touch فایلی که وجود دارد را touch کنیم تنها زمان دسترسی آن update میشود

\_ اگر در هنگام ساخت یک فایل با استفاده از دستور touch فاصله گذاشته شود چندین فایل ساخته میشود (touch sajjad ahmad هم فایل sajjad ساخته میشود و هم فایل ahmad ) برای اینکه فایلی ساخته شود که نامش sajjad ahmad باشد با از “” استفاده شود (touch “sajjad ahmad”)

\_ مکانیزم استفاده از دستور mv =====> mv name\_file destination ======> mv sajjad.txt /home/Documents

\_برای rename کردن =======> mv sajjad.txt sajjad-1.txt

\_ مکانیزم استفاده از دستور cp =====> cp name\_file destination ======> cp sajjad.txt /home/Documents

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

برای اینکه بتوانیم یک پوشه را داخل یک پوشه دیگر کپی کنیم باید از آپشن -R استفاده کنیم

cp -R Desktop/ Documents/

الان پوشه Desktop به داخل پوشه Documents کپی میشود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

همانطور که در بالا گفته شد mrdir تنها پوشه های خالی را پاک میکند برای پاک کردن میتوانید از دستور rm استفاده شود

rm برای پاک کردن فایل استفاده میشود اما میتوان از آن برای پاک کردن پوشه های پر نیز استفاده کرد برای این منظور باید از آپشن -rf استفاده کرد

rm -rf Documents

توجه شود که rm -rf دستور خطرناکی میباشد و باید هنگام استفاده دقت کنید

برای پاک کردن تمامی فایلها در یک پوشه میتوان از دستور rm \* استفاده کرد (یا مثلا تمامی فایل های txt : rm \*.txt (

=====================================================================================

آرشیو سازی و فشرده سازی:



نکته : در دو دستور آخر حتما باید آپشن f آخر نوشته شود چرا که بعد از f باید نام file نوشته شود

tar -czf یا tar -zcf

tar -cjf یا tar -jcf

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور zip =====> zip namefile.zip namefile …. =====> zip pictures.zip 1.jpg 2.jpg 3.jpg

هر 3 فایل jpg را داخل pictures.zip قرار میدهد

به طور پیش فرض از دستور zip برای فشرده سازی fileها استفاده میشود برای اینکه بتوانیم یک پوشه را به درستی zip کنیم باید از آپشن -r (recursive) استفاده کرد =============> zip -r files.zip file الان پوشه فایل به درستی در فایل files.zip قرار گرفته است

برای خروج از حالت zip باید دستور unzip namefile.zip استفاده شود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور tar =====> tar -cf namefile.tar namefile …. =====> tar -cf pictures.tar 1.jpg 2.jpg 3.jpg

(-cf به معنی create file میباشد )

برای خروج از حالت tar باید دستور tar -xf namefile.tar استفاده شود (-xf به معنی extract file میباشد)

برای اینکه ببینیم چه فایل هایی از حالت tar خارج میشوند میتوانیم از آپشن -v (verbose) استفاده کنیم

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور gzip =====> gzip namefile …. =====> gzip pictures.tar اکنون فایل جدید ساخته میشود به نام pictures.tar.gz که فشرده pictures.tar میباشد

برای خروج از حالت gzip باید دستور gunzip namefile.gz استفاده شود

توجه شود که اگر بخواهید فایلی را gzip کنید بعد از اینکه فایله gzip شده ساخته شده فایله اصلی حذف میشود و در هنگام gunzip هم بعد ازاینکه از حالت فشرده خارج شده فایل .gz حذف میشود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

مکانیزم استفاده از دستور bzip2 =====> bzip2 namefile …. =====> bzip2 pictures.tar اکنون فایل جدید ساخته میشود به نام pictures.tar.bz2 که فشرده pictures.tar میباشد

برای خروج از حالت bzip2 باید دستور bunzip2 namefile.gz استفاده شود

توجه شود که اگر بخواهید فایلی را bzip کنید بعد از اینکه فایله bzip شده ساخته شده فایله اصلی حذف میشود و در هنگام bunzip هم بعد ازاینکه از حالت فشرده خارج شده فایل .bz2 حذف میشود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

میتوانیم هنگامیکه با استفاده از دستور tar فایل ها را archive میکنیم با استفاده از یکی از الگوریتم های فشرده سازی gzip یا bzip2 فایل فشرده tar ایجاد کنیم

tar -czf namefile.tgz 1.jpg 2.jpg 3.gpg =====> الگوریتم gzip

tar -czf namefile.gz 1.jpg 2.jpg 3.jpg =====> الگوریتم gzip

tar -cjf namefile.bz2 1.jpg 2.jpg 3.jpg =====> الگوریتم bzip2

=====================================================================================

جستجو و خارج کردن اطلاعات از فایل ها



با استفاده از cat میتوانید محتویات یک فایله متنی را داخل ترمینال مشاهده کنید اما اگر فایل شما یک فایل فشرده شده باشد برای این کار از دستور less باید استفاده شود (توجه شود که از less برای فایل‌های متنی هم استفاده می‌شود)

cat abc.txt less abc.zip less abc.txt

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور head ابتدای یک فایل را نمایش میدهد (به طور پیش‌فرض ۱۰ خط اول نمایش داده می‌شود اما میتوانید تعداد این خط هارا کنترل کنید)

head abc.txt ===> 10 lines head -n 2 abc.txt ======> 2 lines

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور tail انتهای یک فایل را نمایش میدهد (به طور پیش‌فرض ۱۰ خط آخر نمایش داده می‌شود اما میتوانید تعداد این خط هارا کنترل کنید)

tail abc.txt ===> 10 lines tail -n 2 abc.txt ======> 2 lines

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور find هماننده سرچ در ویندوز میباشد

find directory option

find . -type f =====> در پوشه ای که هستید تمام فایل‌ها را برمیگرداند

find . -type d ======> در پوشه ای که هستید تمام پوشه ها (دایرکتوری) را برمیگرداند

find . -name “saj\*” ===> در پوشه ای که هستید تمام فایل و دایرکتوری هایی که با این ۳ حرف شروع می‌شوند را برمیگرداند

find . -name “\*saj\*” ===> در پوشه ای که هستید تمام فایل و دایرکتوری هایی که شامل این ۳ حرف میباشد را برمیگرداند

در لینوکس علامت ( | ) به این صورت معنی می‌شود که : خروجی دستور اول ورودی دستور دوم

find . -type f | grep saj ====>

ابتدا تمامی فایل‌هایی که در پوشه ی مورد نظر وجود داشت را پیدا میکند اسم این فایل‌ها را به دستور grep میدهد و سرچ میکند که کدام فایل اسمش شامل saj میباشد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور greb محتویات یک فایل را سرچ میکند

greb search-value file-name

greb sajjad abc.txt ====> در فایل abc.txt هر کجا که sajjad وجود داشته باشد را نشان میدهد

greb sajjad \* ====> در تمامی فایل‌هایی که در پوشه فعلی وجود دارد به دنبال کلمه مورد نظر میگردد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور cut قسمتی از یک فایل بر اساس در خواست ما برمیگرداند

cut – c3 – 5 abc.txt =======> در فایل abc.txt از ستون ۳ تا ستون ۵ را نمایش میدهد

cut – c3,5 abc.txt ===========> در فایل abc.txt تنها ستون ۳ و ۵ را برمیگرداند

cut – c 3 – abc.txt ====> در فایل abc.txt از ستون ۳ به بعد را نمایش میدهد (بعد از (-) چیزی نوشته نمیشود)

در دستور cut آپشنی وجود دارد که میتوانید با آن جدا کننده را مشخص کنید و بر اساس فیلد خروجی بگیرید

cut -d” “ - f 2 abc.txt ====> در فایل abc.txt (فاصله ” ”) به عنوان جدا کننده میباشد و فیلد دوم نمایش داده می‌شود

cut -d “ - “ -f 2 – abc.txt =====> در فایل abc.txt (“ - “) به عنوان جدا کننده میباشد و از فیلد دوم به آخر نمایش داده می‌شود

دستور cut نمیتوان آپشن های ترکیبی را دریافت کند برای این کار میتوانیم از ( | ) استفاده کنیم

cut – d “ “ - f 3 abc.txt | cut – c 2 ======> ابتدا از فایل abc.txt بر اساس جدا کننده ی فاصله فیلد ۳ را برگردان حال ستون ۲ آن را نمایش بده

میتوانیم با استفاده از < خروجی دستور را در یک فایل دیگر ذخیره کنیم

cut -d “ - “ -f 2 – abc.txt > testfile.txt ===> خروجی دستور cut در فایل testfile.txt ذخیره می‌شود

میتوانیم با استفاده از << خروجی دستور را به انتهای یک فایل اضافه کنیم

cut -d “ - “ -f 2 – abc.txt >> testfile.txt ===> خروجی دستور cut به انتهای testfile.txt اضافه می‌شود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

دستور wc تعداد خطوط , تعداد کلمات و تعداد کاراکتر ها را نمایش میدهد ====> wc abc.txt

=====================================================================================

تبدیل کردن دستورات به shell script :

مجموعه از دستورات لینوکس را در فایلی در کنار هم قرار دهید و این فایل را به صورت یک فایل اجرایی اجرا کنید

تفاوت مسفر ( interpreter ) با کامپایلر در این است که کامپایلر همه ی کد را یک‌جا میگرفت و یک‌جا اجرا میکرد اما مفسر خط به خط اجرا میکند ( با توجه به اینکه script نویسی هم به صورت خط به خط میباشد ما از مفسر استفاده میکنیم )

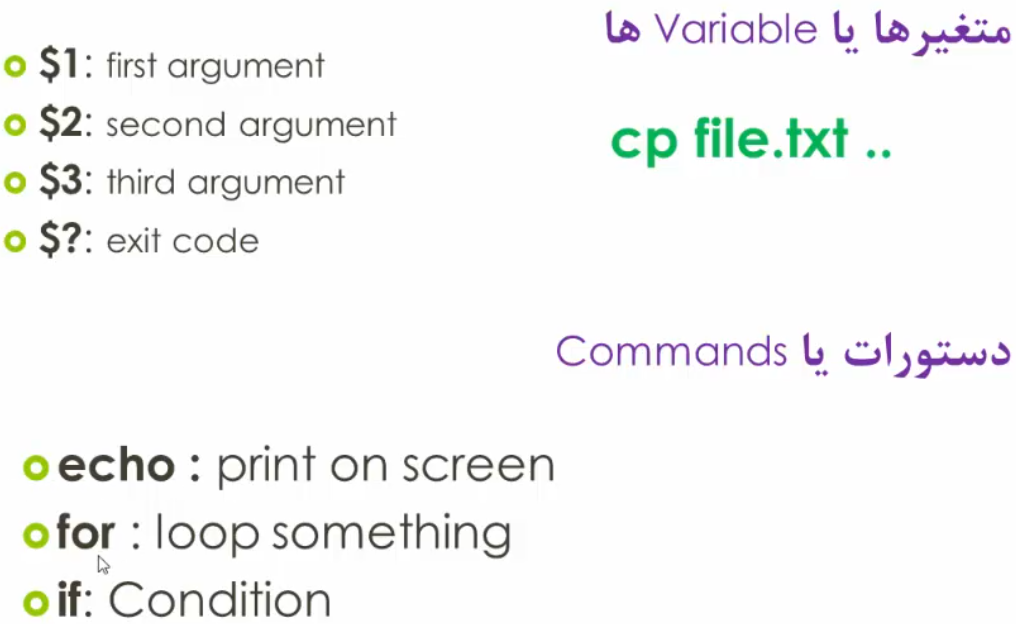
زمانی که درباره مفسر در لینوکس صحبت می‌شود منظور shell میباشد ( shell اون رابط کاربری که اون دستوراتی که در محیط لینوکس وارد میکنیم را برای سیستم عامل ترجمه میکند )

اولین خط هر اسکریپ باید مشخص شود که قرار است از چه نوع shell استفاده شود ( خطوط سبز رنگ در یک زیر )



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

$1 در اسکریپت نویسی به این معنی میباشد که اولین آرگومان دستورم را به عنوان ورودی متغیرم قبول کن ( مثلاً در دستور   
 .. cp file.txt عبارت file.txt به عنوان ورودی اول( که در 1$ ذخیره می‌شود ) و .. به عنوان ورودی دوم ( در 2$ ذخیره می‌شود ) در نظر گرفته می‌شود )



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



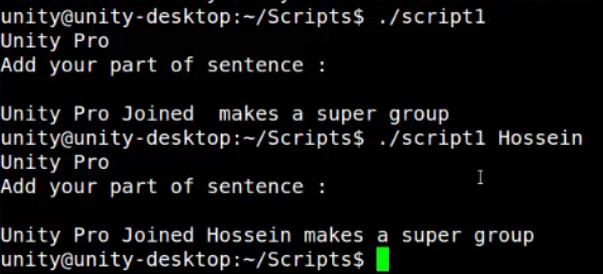
برای اینکه بتوانیم یک فایل اسکریپ را به صورت یک فایل اجرایی در آوریم باید دستور زیر را وارد کنیم ( chmod +x script1 )

اکنون فایل script1 یک فایل اجرایی اسکریپت میباشد

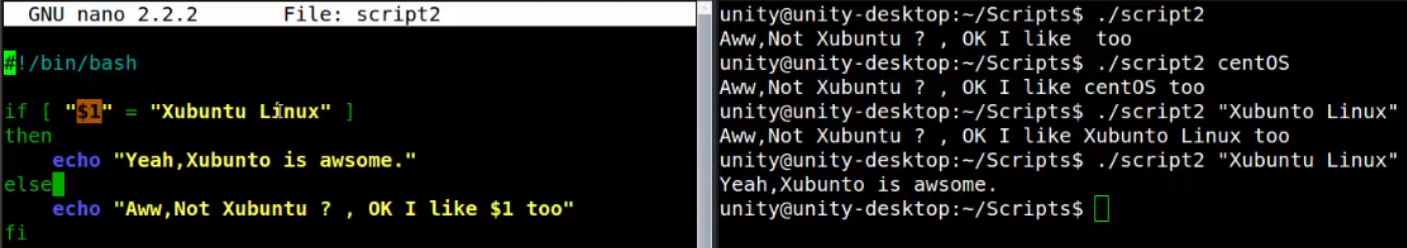
هنگام تعریف کردن یک متغیر لازم نیست که از علامت $ استفاده شود اما هنگام فراخوانی لازم است که علامت $ پشت نام متغیر قرار گیرد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

حالت اجرایی script1

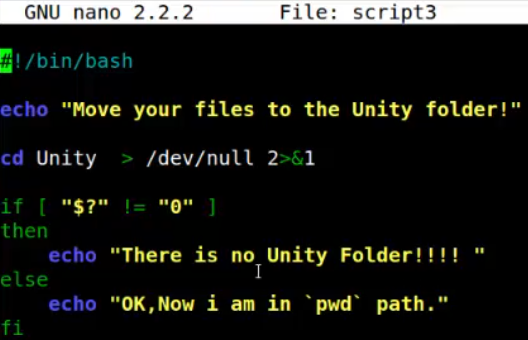


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



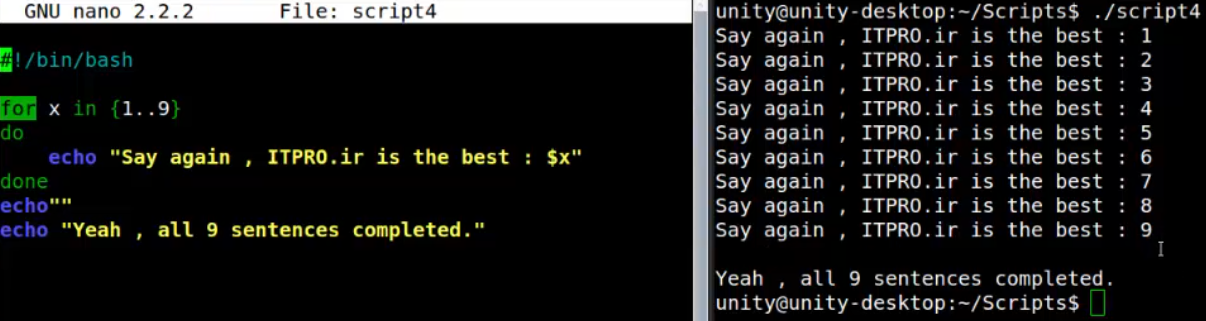
مکانیزم اینکه بتوانیم از if خارج شویم fi است

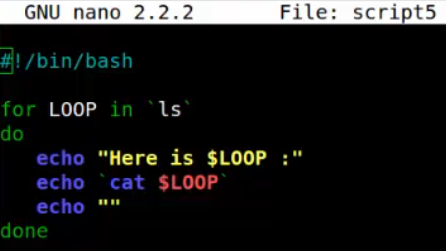
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



در این اسکریپت ۲ نکته وجود دارد : اول 1$<2 میباشد به این معنی که به ارور ها توجه نکن و به ادامه اسکریپت را اجرا کن ( در این مثال به دلیل اینکه ممکن هست پوشه Unity وجود ندارد به ارور بخوریم اما با نوشتن این خط کد از بروز ارور جلوگیری میکنیم ) دومین نکته علامت ` میباشد ( این علامت بالای دکمه ی tab قرار دارد ) معنی این علامت به این صورت میباشد که دستوری که در بین این دوعلامت قرار گرفته است را اجرا کن ( در این مثال دستور pwd به دلیل اینکه بین دو ` قرار گرفته است هنگام اجرای اسکریپ اجرا می‌شود و خروجیش به وسیله دستور echo چاپ می‌شود )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

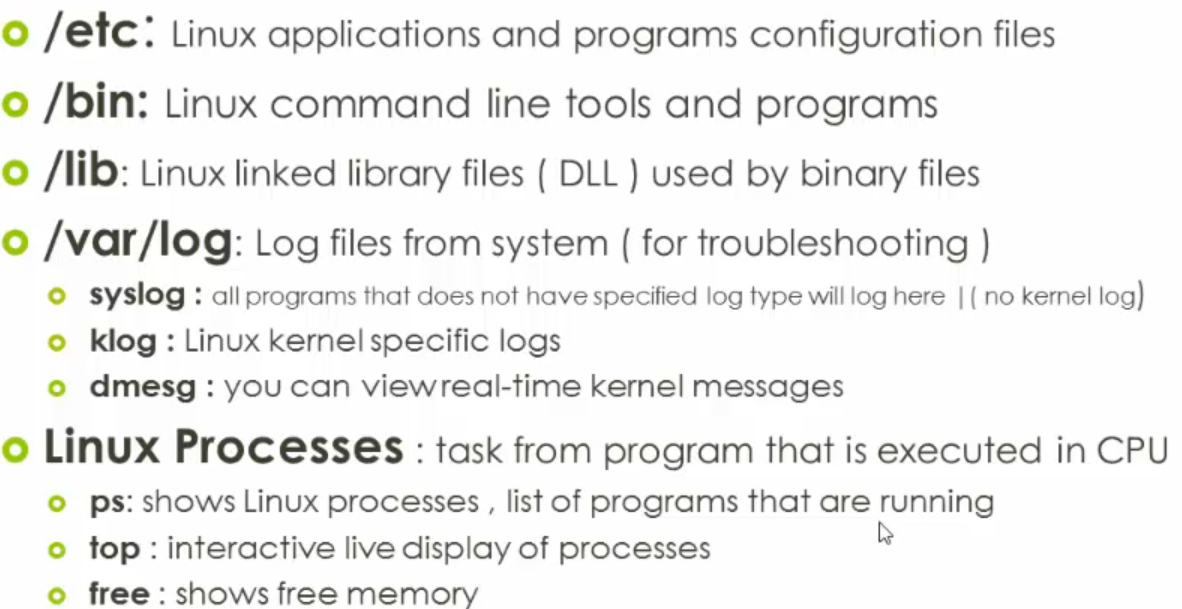
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



در این اسکریت به ازای هر فایل یا فولدری که وجود دارد یک بار حلقه ی for اجرا می‌شود و خروجی هایش یک بار نام فایل یا فولدر میباشد و خروجی دیگرش نمایش محتویات آن فایل مورد نظر میباشد که با استفاده از دستور cat این کار را انجام داده‌ایم

=====================================================================================

کاربرد etc , lib , log و process ها :

  
برای درک بهتر دایرکتوری lib بهتر است به یک نرم‌افزاری فکر کنید که به صورت ماژولار نوشته شده است مثلاً ماژول تبدیل تاریخ شمسی را نوشته‌اید این ماژول را در دایرکتوری lib قرار میدهید و از این به بعد هر نرم‌افزار دیگری نیاز به تبدیل تاریخ شمسی داشت میتواند از این library استفاده کند ( همان‌طور که در عکس بالا گفته شد در پوشه ی bin برنامه هایمان وجود داشت که اجرا میشد حال اگر برنامه‌ای نیاز داشت به تاریخ شمسی به پوشه lib لینک می‌شود )

مثلاً در پوشه lib فایلی وجود دارد به نام init که فرایند بوت سیستم یا خاموش و روشن کردن سیستم وابسته به library (dll) init میباشد حال اگر برنامه‌ای را نصب کرده‌اید که لازم باشد سیستم یک بار ریستارت شود و در این هنگام آپدیتی هم دریافت کرده باشید لازم نیست که هر دو برنامه سیستم را یک باز خاموش و روشن کنن آن‌ها درخواست خود را از طریق library (dll) init به cpu ارسال میکند اینگونه مدیریت منابع هم بهتر می‌شود

نکته ی دیگر این است که هر کاربر میتواند برای خود lib مخصوص به خود را داشته باشد (user/lib/) ( این نکته برای تمامی موارد بالا وجود دارد یعنی اینکه میتوانیم مثلاً در پوشه etc فایل config برنامه‌ای را عوض کنیم به صورتی که برای بقیه ی کاربران تغییر نکند ( system wide configuration یا user wide configuration ))

دستور ps به تنهایی process های مربوط به همان user نمایش میدهد اما با سوییچ aux تمامی process های سیتم را نمایش میدهد

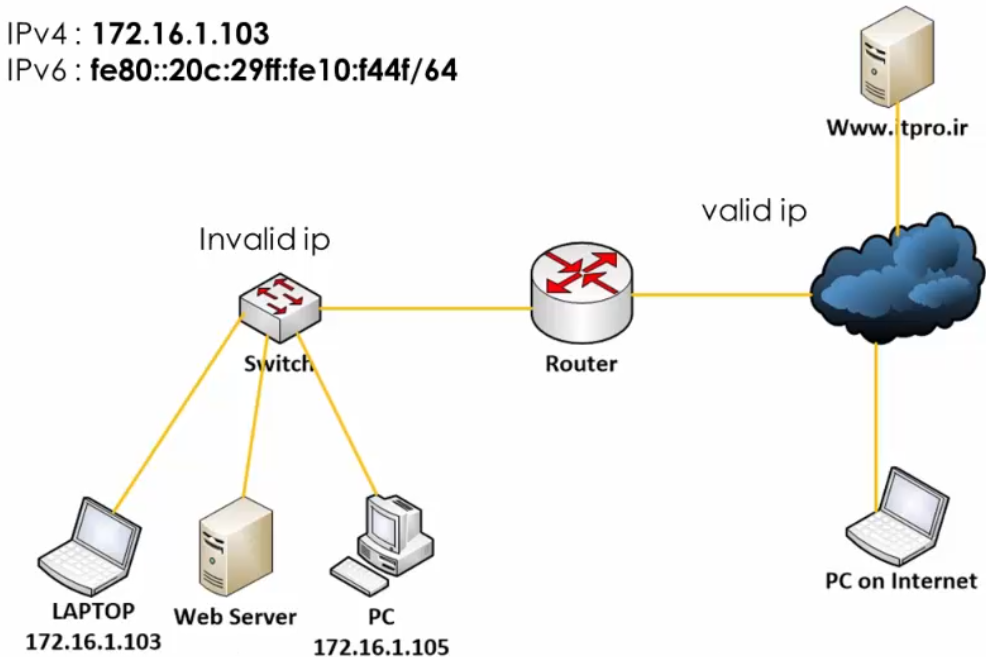
با دستور kill میتوانیم process های در حال اجرا را ببندیم و تمام کنیم ( kill PID )

دستور top به صورت live تمامی process ها را نمایش میدهد

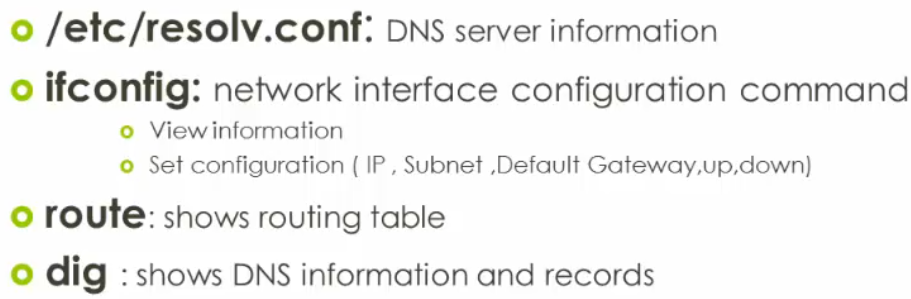
free -tom free – tob ====> سوییچ tom به صورت مگابایت و tob به صورت بایت نمایش میدهد

=====================================================================================

تنظیمات مقدماتی شبکه :



در عکس بالا منظور از valid ip همان public ip میباشد و منظور از invalid ip همان private ip میباشد



دستور ifconfig سوییچی دارد به نام a که تمام اطلاعات کارت شبکه‌هایتان چه فعال و چه غیر فعال را نمایش میدهد

برای فعال کردن یک کارت شبکه ی غیر فعال میتوانیم از دستور ifconfig eth1 up استفاده کنیم

برای آدرس دهی هم میتوانیم از دستور ifconfig استفاده کنیم ( ifconfig eth1 192.168.1.10 netmask 255.255.255.0 )

dig google.com

route google.com

=====================================================================================

مقدمات امنیت و انواع کاربران :



system user ها در linux به این صورت هستند که شما نمیتوانید ببینید کجا هستند , home folder ندارند چون اصلاً قرار نیست با آن‌ها بتوانید login کنید وظیفه‌ی آن‌ها اجرا کردن سرویس ها میباشد ( مثلاً شما آپاچی وب سرور دارید قطعاً یک کاربر سیتمی دارد که آن را اجرا میکند )

دستور دیگری هم در قسمت ابزارهای شناسایی اطلاعات کاربران وجود دارد : whoami



تمام اطلاعات کاربرانمان از جمله نام کاربری , شناسه ی کاربری و … در فایل passwd ذخیره می‌شود

در گذشته پسورد ها هم در فایل passwd به صورت رمزنگاری شده وجود داشت اما الان پسورد ها در فایل shadow قرار دارد

اگر دستوراتمان را با استفاده از sudo وارد کنیم در فایله مربوطه log برداری می‌شود

su یک نقطه ضعف امنیتی دارد چرا که هنگام وارد کردن پسورد در آن لحظه رمزنگاری نمیشود

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

هنگامیکه از دستور id استفاده میکنیم اطلاعات زیر را میتوانیم استخراج کنیم ( اگر در جلوی دستور id چیزی قرار ندهیم اطلاعت نمایش داده شده مربوط به همان کاربری است که با آن لاگین کرده‌ایم اما اگر نام کاربری را قرار دهیم میتوانیم اطلاعات آن کاربر مورد نظرمان را مشاهده کنیم ):

uid در لینکوس اعدادی هستند که معنا و مفهوم دارند به این معنی که کاربر روت در لینوکس uid صفر دارد و کاربرانی که  
system user هستند uid آن‌ها معمولاً بین رنج 1 تا 499 میگیرند و کاربران عادی یا همان standard user ها uid آن‌ها بین رنج 500 تا 6000 را میگیرند و کاربرانی که در شبکه استفاده می‌شوند ( مصارف سرویس های شبکه دارند ) uid آن‌ها از 6000 به بالا میباشد ( 65534 , uid میباشد که رزرو شده نمیباشد و معروف به no body میباشد )

gid هم ماننده uid تقریباً همان مفاهیم را دارد

groups هم اطلاعاتی درباره گروه‌هایی که کاربران در آن‌ها عضو میباشد را نشان میدهد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

در دایرکتوری home/ میتوانیم پروفایل standard user هایی که ساخته شده‌اند را مشاهده کنیم

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

در فایل passwd پارامتر ها بوسیله ی : از هم جدا می‌شوند ( اطلاعات تمامی کاربران چه standard user و چه system user در فایل passwd وجود دارد )

x در فایل passwd به فایل shadow اشاره دارد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

su دستوری میباشد که میتوانید به وسیله ی آن کاربر خود را در ترمینال عوض کنید به این صورت که اگر جلوی دستور su نام کاربری مورد نظر خود را وارد کنید و پسوردش را بزنید کاربر شما در آن session terminal عوض می‌شود اما اگر دستور su خالی را وارد کنید به عنوان کاربر root وارد می‌شوید

=====================================================================================

ایجاد و مدیریت کاربران در لینوکس :



مفهومی وجود دارد به این صورت که اگر کاربری ایجاد شود با سطح دسترسی های محدود اما این کاربر را عضور گروه sudo کنیم تمام سطح دسترسی های که در گروه sudo وجود دارد ( بالاترین سطح دسترسی ) به این کاربر داده می‌شود

ssh [sajjad@localhost](mailto:sajjad@localhost) apt-get install openssh-clint openssh-server

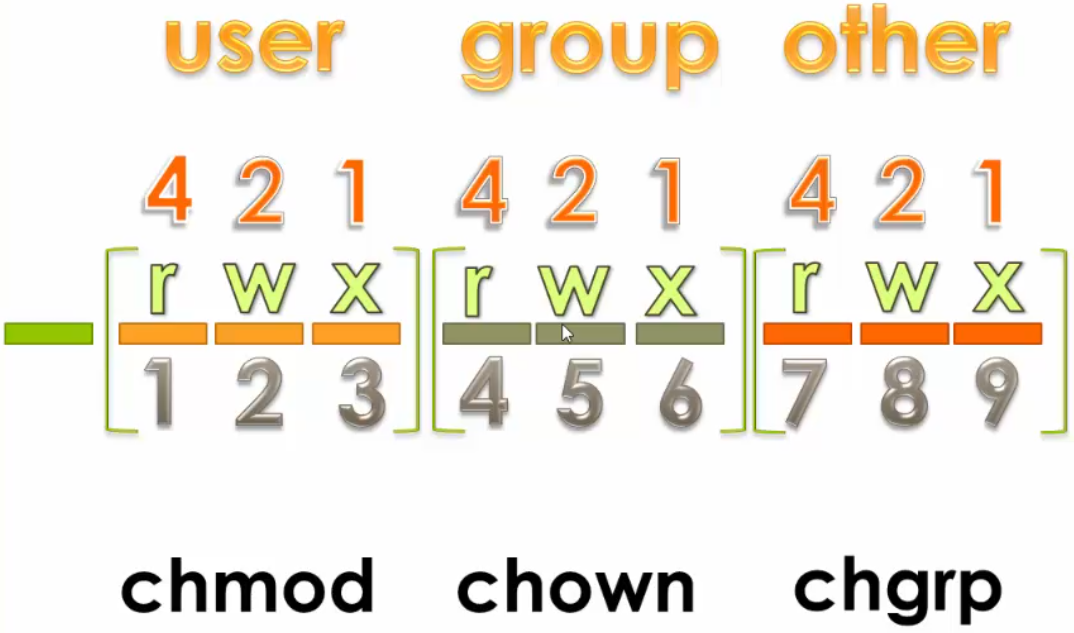
با استفاده از دستور usermod -a -G admin sajjad میتوانیم کاربر sajjad را عضو گروه admin کنیم ( بالاترین سطح دسترسی گروه‌ها گروه sudo میباشد )

سوییچ های مهم دستور userdel :

-r, --remove                  remove home directory and mail spool  
-f, --force               force removal of files

=====================================================================================

سطوح دسترسی فایل‌ها و مالکیت در لینوکس :



سطوح دسترسی به 10 قسمت تقسیم‌بندی می‌شود اولین قسمت نشان دهنده ی فایل یا دایرکتوری میباشد ( اگر – بود مربوط به یک فایل میباشد اما اگر d بود مربوط به یک directory میباشد )

9 قسمت دیگر را به 3 بخش مساوی تقسیم میکنیم :

۳ قسمت اول یا بخش اول مربوط به سطوح دسترسی کاربر ( u یا user )

۳ قسمت دوم یا بخش دوم مربوط به سطوح دسترسی گروه ( g یا group‌ )

۳ قسمت سوم یا بخش سوم مربوط به سطوح دسترسی other یا سایرین میباشد ( هر کسی که در قسمت user یا group ما تعریف نشده است ) ( o یا other )

r به معنی read و w به معنی write و x به معنی execute یا اجرا کردن میباشد ( execute برای یک directory به این معنی میباشد که میتوانیم داخل آن پوشه cd کنیم و read به معنی ls گرفتن در داخل directory میباشد )

chmod به معنی change modifications که permission ها را تغییر میدهد

chown به معنی change owner که بحش اول را تغییر میدهد

chgrp به معنی change group که بخش دوم را تغییر میدهد

chmod u+w sajjad.txt chmod u-w sajjad.txt chmod ugo+w sajjad.txt chmod o+wx sajjad.txt

chmod g+rwx,o-rwx sajjad.txt

chmod 721 sajjad.txt chmod 000 sajjad.txt

chmod 777 Folder1 -R === سطوح دسترسی تمام زیر مجموعه های پوشه مورد نظر را نیز تغییر میدهد

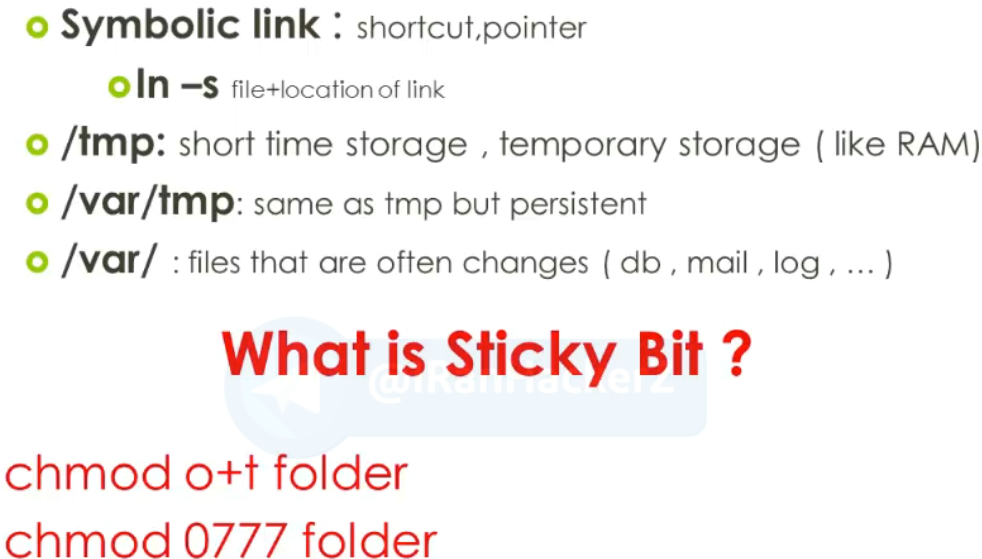
chown root sajjad.txt ===> مالک فایل کاربر روت شد

chown root Folder1 -R

chgrp adm sajjad.txt

=====================================================================================

ایجاد لینک و معرفی دایرکتوری های خاص :



- برای ساخت symbolic link باید از دستور زیر استفاده کنیم:

ln -s main\_name\_file shortcut\_name\_file

ln -s sajjad.txt /*Desctop*/test/ahmad.txt

در مثال بالا فایلی به نام ahmad.txt در دایرکتوری Deskrop/test ایجاد می‌شود که به فایل sajjad.txt اشاره دارد

- tmp به معنی موقتی میباشد . هر نرم افزاری بخواهد به صورت موقتی یک فایلی را ایجاد کند این کار را در پوشه ی tmp انجام میدهد (تفاوتی که بین دایرکتوری tmp/ و var/tmp/ وجود دارد این است که tmp ماننده ram میباشد یعنی با خاموش شدن سیستم تمام اطلاعاتی که در این دایرکتوری میباشد از بین می‌رود اما در دایرکتور var/tmp/ بعد از ریستارت شدن سیستم تمامی اطلاعات وجود دارد و از بین نمیرود )

- در دایرکتوری var میتوان گفت بیشتر مواردی که تغییرات زیادی دارند در این پوشه قرار میگیرند

- طبیعتاً دایرکتوری tmp/ دایرکتوری هست که همه ی افراد ( همه ی user هایی که داخل سیستم عامل ) هستند به این دایرکتوری دسترسی write داشته باشند ( نکته این است که اگر همه ی افراد به این پوشه دسترسی داشته باشند میتوانند اطلاعات دیگر user ها را نیز دستکاری کنند ) در این مورد sticky bit راهکار میباشد به این صورت که درست است که همه ی افراد به این پوشه دسترسی دارند ولی فقط میتوانند فایل‌هایی رو تغییر دهند که مالک آن باشند و نمیتوانند دیگر فایل هارا تغییر دهند

- به صورت پیش‌فرض sticky bit در پوشه tmp فعال و در دیگر پوشه ها غیر فعال میباشد

- به صورت کلی اگر پوشه والد اجازه ی full control به other داده بود فایل فرزند با این شرایط که به other اجازه ی write نداده باشد بازهم other میتوانند محتویات فایل فرزند را تغییر دهند

chmod o+t folder

chmod 777 folder

chmod o-t folder

=====================================================================================