강의내용

psftp – ftp. pc간에 파일올리는..



사용방법은 좀더 알아봐야 할 것 같습니다.

오늘은 리눅스 기본 명령어에 대해 알아보는 수업입니다.

참고 - <http://www.oreillynet.com/linux/cmd/> 에서 리눅스 명령어의 정보를 찾을 수 있습니다.

명령어는 강의노트에도 정리되어 있습니다.

리눅스 또한 파일이 트리구조를 이루고 있습니다.

pwd = present – 현재 dir

ls = list – 전체 리스트

/로 시작하지 않는 경로는 상대경로이고, 이는 현재 위치에서 이어지는 경로를 의미합니다. (현재 상태에 상대적으로 달라지게 되는 것 입니다.)

cat = file의 내용을 보여주는 명령어

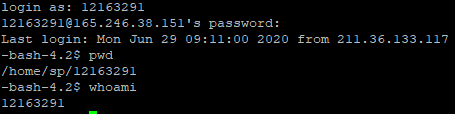
HW

5.1 Do following steps.

**1) Log in to a Linux server. Find your login directory with "pwd". Find your user ID with "whoami".**

$ pwd

$ whoami



학교 서버에 학번으로 접속 한 후,

**pwd**로 현재 dir위치를 확인하여 보니

**…/12163291** 인 것을 확인할 수 있었습니다.

**whoami** 로 저의 userID를 확인하여 보니

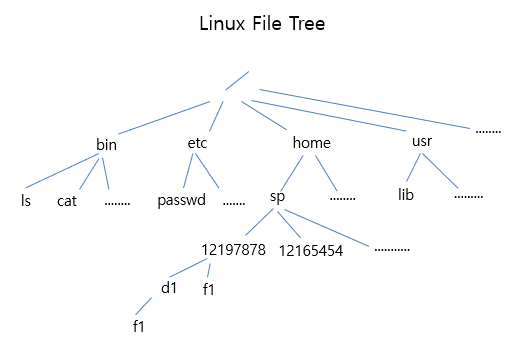
**12163291** 인 것을 확인할 수 있었습니다.

**2) Go up the file tree with "..", display the current location with "pwd", and find other students directory name with "ls".**

$ cd ..

$ pwd

$ ls

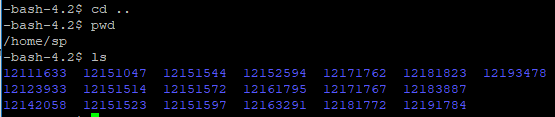


파일 트리 구조가 위와 같으므로,

cd .. 명령어를 통해 한 노드 위로 올라간후,

ls 명령어를 통해 자식 node 리스트를 확인하여 보면

다른 학생 분들의 학번을 확인할 수 있을 것입니다.



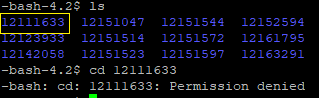
pwd 로 확인하여 보니, 저는 12163291의 윗 노드인 sp에 잘 위치해 있는 것을 확인할 수 있었습니다.

현재 총 19명의 학생분들이 있다는 것 또한 확인할 수 있었습니다.

**3) Pick one of the student IDs (suppose it was 12345) and try to enter his/her directory with "cd".**

**$ cd 12345**

**What happens?**



첫번 째 분인 12111633 디렉토리에 접근하려 했지만,

**Permission denied**를 당했습니다.

**4) Use "ls -l" to see the file permission of the directory you tried to enter.**

$ ls -l

............

drwx------ 2 12345 12345 4096 Feb 27 11:47 12345

............

/\* 위 결과의 part별 설명 \*/

d: this file is a directory

rwx: the owner can read/write/execute this file

--- : the group member has no read/write/execute permission

--- : other people has no read/write/execute permission

2 : there are two links to this file

12345 : the user ID of the owner of this file

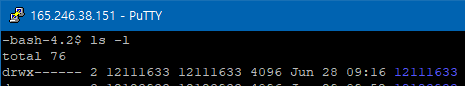
12345 : the group ID of of the owner of this file

4096 : the byte size of this file

Feb 27 11:47 : the creation time or last modification time of this file

12345 : file name

Explain why you couldn't enter the directory you picked.



강의 노트 내용을 참고하면,

12111633 directory는

1. 사용자에게의 권한 – 읽기可 / 쓰기可 / 실행可 입니다.

2. 그룹멤버에게의 권한 – 읽기不可 / 쓰기不可 / 실행不可 입니다.

3. 다른 이들에게의 권한 - 읽기不可 / 쓰기不可 / 실행不可 입니다.

즉, 이는 **사용자 이외에는 접근이 불가능 합니다**.

따라서 Permission denied 가 된 것입니다.

**5) Find out your current location again with "pwd".**

$ pwd

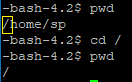


또 다시 현재의 directory 위치를 확인하여 보니, ..sp 인 것을 확인할 수 있었습니다.

6) Go to the root directory with "cd /" , and make sure you are really at the top directory with "pwd".

$ cd /

$ pwd



5) 번에서 dir의 route를 보면, 가장 첫 부분 = root가 / 인것을 확인할 수 있었습니다.

cd / ( / 有 이므로 절대경로 ) 를 통해 /로 이동한 후,

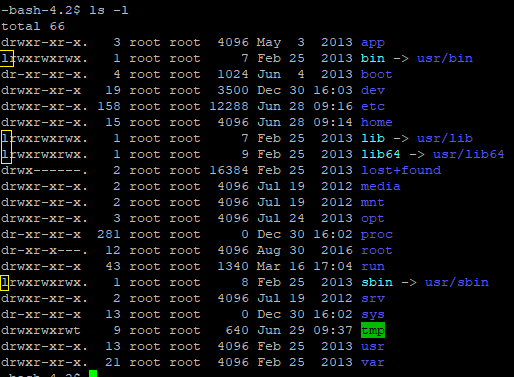
pwd 를 통해 확인하여 보니, 현재 위치가 root인 / 라는 것을 확인할 수 있었습니다.

**7) How many files you have in the root directory? Some of them are not directory files. Find them with "ls -l". Use "file" command to see more detailed info.**

$ ls -l

$ file \*

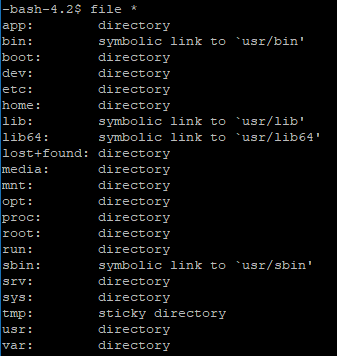
$ file lib



4) 번 문제의 내용을 참고하면, ls 결과의 맨 앞글자는 파일의 유형을 의미합니다, 따라서 어떤 파일의 ls 값이 맨 앞글자가 d가 아니라면 그 파일은 디렉토리가 아닐 것입니다.

즉, root의 자식중 directory 가 아닌 것들은

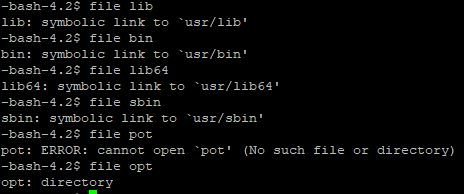
**bin, lib, lib64, sbin** 이 있습니다.



file \* 로 root의 모든 자식의 파일 유형을 확인하여 보았습니다.

l = symbolic link 를 의미하는 것을 알 수 있었습니다.

file \*는 모든 파일의 유형을 한번에 볼 수 있지만,



위와 같이, file file이름

으로는 각 파일의 유형을 하나씩 검색할 수도 있습니다.

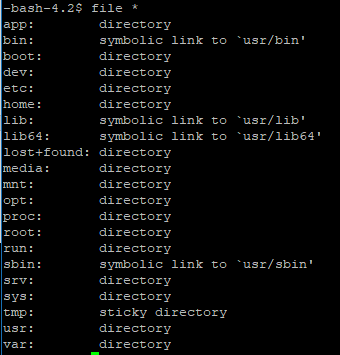
이는 [ 바 로 하 나 ] 하위 노드의 파일에 한해 가능 한 것 입니다.

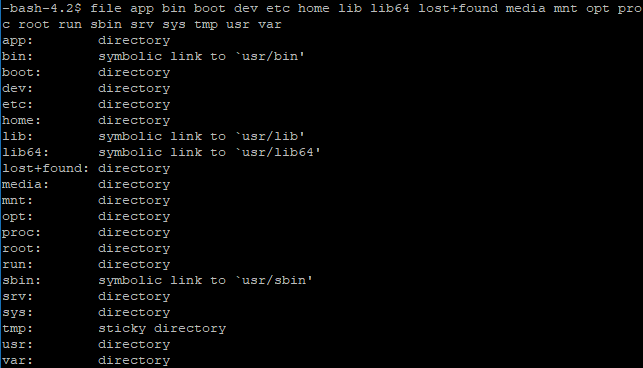
**8) \* is a wild card character meaning it will be replaced by all file names in the current directory.**

$ file \*

is same as

$ file app bin boot dev .... (all file names in the current directory)





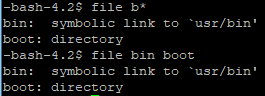
\* = 모든 파일 이름

을 의미하는 것을 알 수 있습니다.

$ file b\*

is same as

$ file bin boot



b\* = b로 시작하는 모든 단어

를 의미함을 알 수 있었습니다.

**9) If you want to go back to your login directory (suppose it was /home/sp1/12345), you can cd with absolute path, cd with relative path, or just "cd".**

cd /home/sp1/12345 -- go to absolute path "/home/sp1/12345"

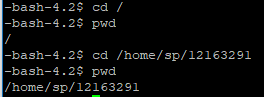
cd home/sp1/12345 -- go to "home" in the current directory and go to "sp1"

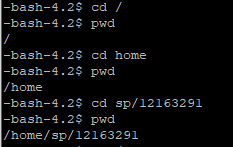
and then to "12345", that is go to relative directory

"home/sp1/12345"

cd -- go to the login directory of the current user

Use one of above methods to go back to your login directory (or home directory).





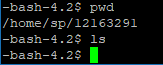


**10) Confirm your current location with "pwd".**



pwd 명령어를 통해 현재 위치를 알 수 있습니다.

**11) List all files in your directory with "ls" command.**



현재 …12163291 에는

자식 파일(노드)이 없습니다. 따라서 바로 커맨드 라인이 결과로 나오게 됩니다.

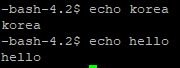
**12) Try "echo" command.**

$ echo korea

korea

$ echo hello

hello

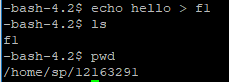


echo로 문자열을 출력할 수 있는 것을 확인할 수 있습니다.

11) Try "echo" with ">" symbol. ">" is called "standard output redirection".

$ echo hello > f1

">" sends the result to "f1" so there is no output shown in the terminal but you will have f1 in the current directory.



현재 …12163291 에

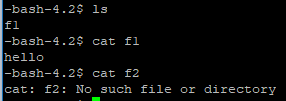
f1이라는 자식 파일(노드)이 생성된 것을 확인 할 수 있습니다.

이는 아마 echo + > 커맨드에 의해 ‘hello’ 라는 내용이 담겨있을 것이라 유추됩니다.

12) Do "ls" to see you can find f1 in the current directory. Show its content with "cat".

$ ls

$ cat f1



cat으로는 파일의 내용을 확인할 수 있습니다.

11) 문제에서 유추했던 것 과 같이, hello의 내용이 출력된 것을 확인할 수 있었습니다.

f2 도 cat으로 내용을 확인해보려 헀지만,

존재하지 않는 파일이므로 역시 error가 발생했습니다.

13) Make a directory, d1, with "mkdir".

$ mkdir d1



mkdir 커맨드를 통해 d1이라는 디렉토리를 …12163291 의 하위에 생성해 주었습니다.

이로인해, dir는 파란글자로 표현되는 것을 확실히 이해했습니다.

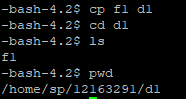
ls를 통해 d1이 성공적으로 생성되었는지 확인했습니다.

14) Copy f1 into directory d1.

$ cp f1 d1 -- make a copy of f1 inside directory d1 under the same name.

$ cd d1

$ ls

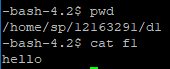


f1파일을 d1이라는 dir안에 copy 해 넣어줍니다

cd d1으로 d1으로 위치를 옮깁니다 – 더 안으로 들어갑니다

ls로 확인해보니 f1이 생성되어있는 것을 확인할 수 있었습니다.

$ cat f1

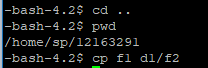


cat을 통해 f1의 내용을 확인하여보니 앞서 만들어 두었던 f1의 내용과 같은 것을 확인하였고,

즉 copy가 성공적으로 이루어진 것을 알 수 있었습니다.

$ cd ..

$ cp f1 d1/f2 -- make a copy of f1 inside d1 under another name f2



cd ..를 통해 파일 트리에서 한번 부모쪽으로 – 위쪽으로 가줍니다.

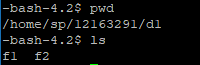
pwd로 잘 갔는지 위치확인을 해준후,

f1을 d1안의 f2라는 파일에 copy 해줍니다.

이때, f2가 없으면, 파일이 새로생성 된 후, 그 위에 카피됩니다.

$ cd d1

$ ls



cd d1으로 d1에 이동한후, pwd로 현 위치를 확인합니다.

ls 로 자식파일목록을 확인합니다.

f2가 생성되어있는 것을 확인할 수 있었습니다.

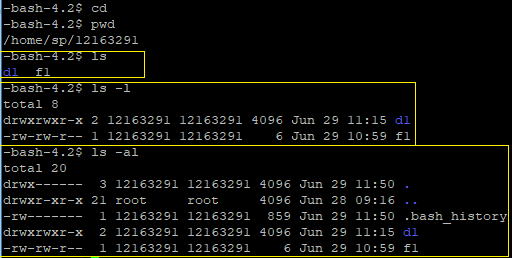
5.2 Do followings and explain what happens and why.

$ cd

$ ls

$ ls –l

$ ls –al



cd로 …/12163291 로그인 위치로 가줍니다.

pwd로 위치확인을 해주었습니다.

ls로 현재 어떤 자식파일이 있는지 확인해줍니다.

-l 은 자식 파일의 정보를

-al은 모든 파일의 정보를 출력하는 것이라 유추할 수 있습니다.

$ cd /

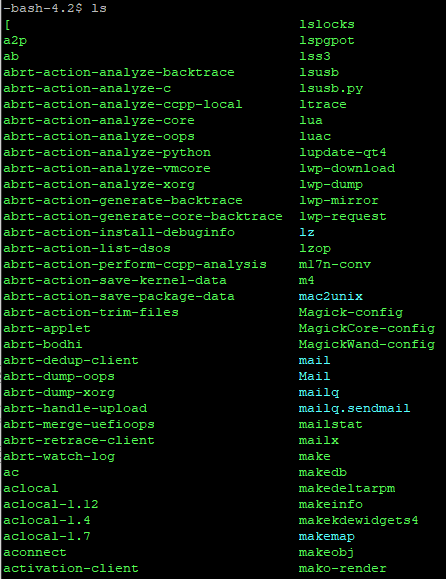
$ cd bin



cd / - 절대경로

를 통해 root로 가준 후, bin으로 하나 내려갑니다.

$ ls



…

bin에서 ls로 자식들을 확인하여 보니 수많은 파일이 있는 것을 확인할 수 있었습니다

$ ls bz\*



ls + ?\* 를 통해 ?로 시작되는 파일 명을 가진 파일들만이 출력되는 것을 확인할 수 있었습니다

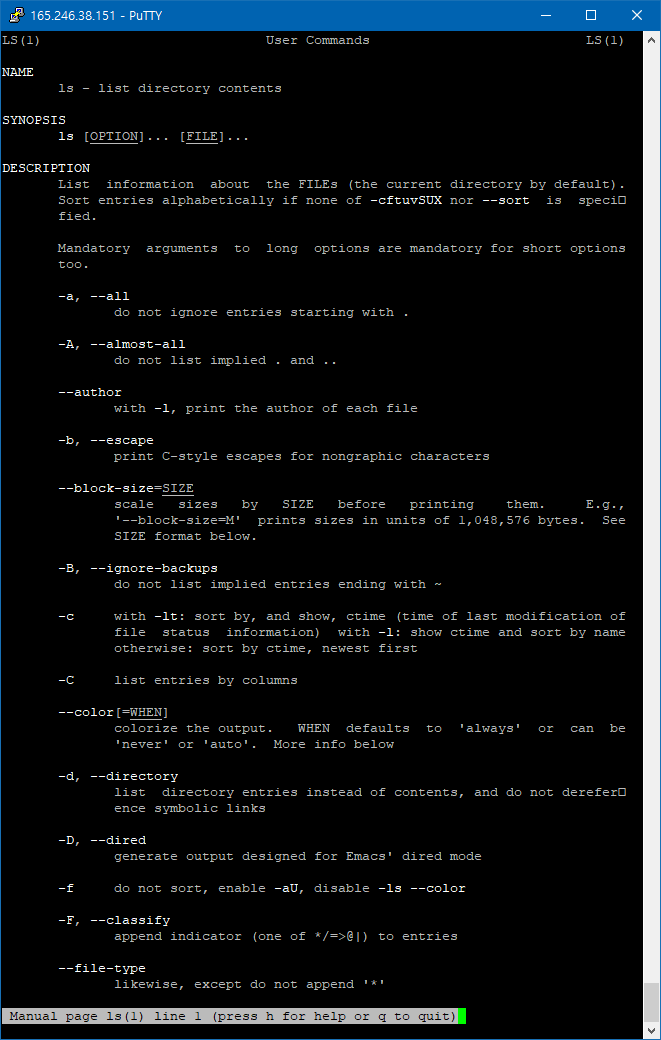
$ cd

$ pwd



cd 로 로그인 위치로 다시가준후, pwd로 위치 확인을 해줍니다.

$ man ls

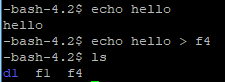


man ls를 통해 ls의 쓰임새를 상세히 확인할 수 있었습니다.

$ echo hello

$ echo hello > f4

$ ls



hello를 출력해준후,

echo > 를 통해 f4라는 파일을 생성하며 + 그안에 문자열을 입력해주었습니다.

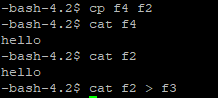
ls를 통해 잘 생성되었는지 확인해 주었습니다.

$ cp f4 f2

$ cat f4

$ cat f2

$ cat f2 > f3



cp f4 f2를 통해 f4의 내용을 f2에 복사해 주었고,

cat을 통해 확인하였습니다.

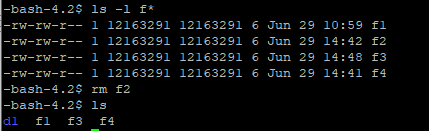
cat > 를 통해 f2의 내용을 f3에써주었습니다.

즉, 이는 echo > 와 비슷한 역할을 합니다.

$ ls –l f\*

$ rm f2

$ ls



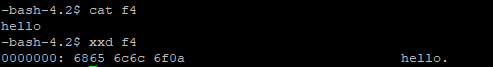
ls -l f\*를 통해 f로 시작되는 파일들의 상세 정보를 확인할 수있었고,

rm(remove) f2로 f2를 삭제했습니다.

ls로 잘 삭제 되었음을 확인하였습니다.

$ cat f4

$ xxd f4



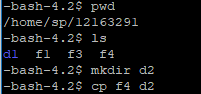
cat으로 파일의 내용을 확인하고,

xxd 로 파일을 16진수로 나타낼 수 있음을 확인하였습니다.

$ mkdir d2

$ cp f4 d2

$ cd d2



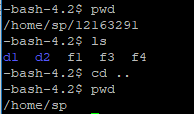
mkdir d2를 통해 directory d2를 하나 더 만들어 주었습니다

cp f4 d2를 통해 d2에 f4를 copy하여 만들어 주었습니다

$ pwd

$ ls

$ cd ..



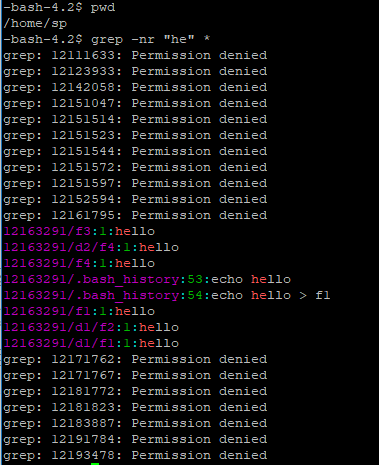
pwd 로 현 위치를 확인하였습니다.

ls로 로그인 위치에서 생긴 파일들의 목록을 확인하니 d2가 성공적으로 생성된 것을 확인할 수 있었습니다.

cd ..로 부모 node로 한칸 올라간 후, pwd로 위치를 확인하니

12163291의 부모인 sp에 위치한 것을 확인 할 수 있었습니다.

$ grep –nr "he" \*



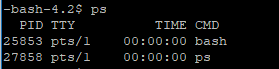
grep -nr “he” \* 를 통해 he로 시작하는 내용의 텍스트가 들어있는

모든 파일을 찾아주었습니다.

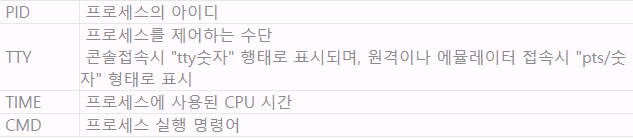
12163291 이 아닌이상 접근이 금지되어있으므로,

제 학번이 아닌 곳에는 접근을 시도할 때마다 접근금지문이 뜨게됩니다.

$ ps

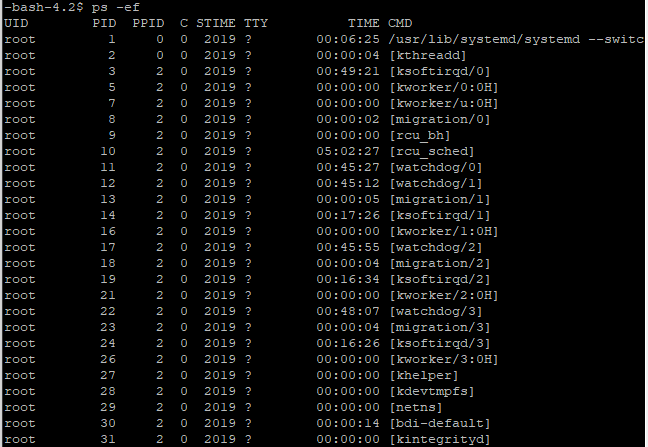


ps로 현 process id와 tty등 프로세스의 정보를 확인할 수 있었습니다.

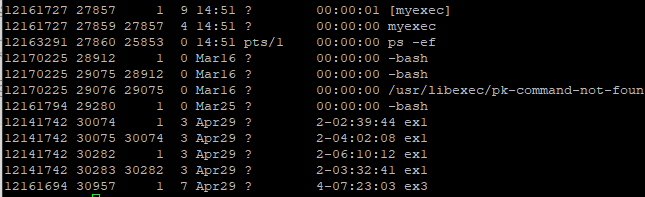


각 목록의 의미에 대한 표 입니다.

$ ps –ef

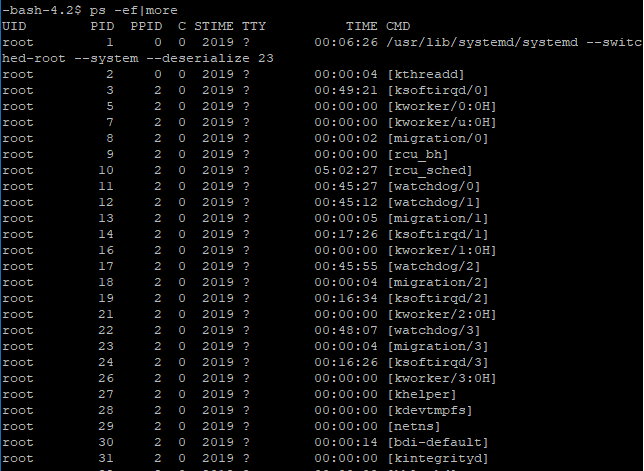


…

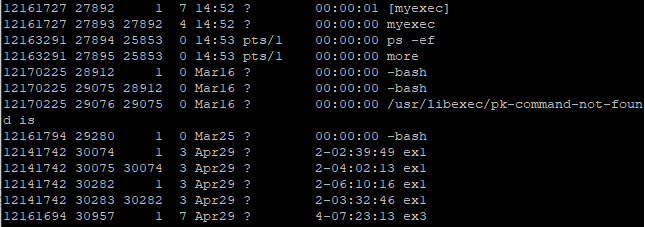


모든 프로세스에 대한 정보를 확인할 수 있습니다.

$ ps –ef | more



…



모든 프로세스에 대한 정보를 확인할 수 있습니다.

이때에는 줄잘림 없이 모든 내용이 제대로 출력됩니다!

+more로 인한 결과이리라 추측 됩니다.

5.3 Find out the usage of following commands. Use them in a Linux server and explain what happens.

\* 각명령어는 man 명령어를 통해 그 쓰임새를 알 수 있습니다.

chmod: 파일이나, 디렉토리의 권한을 변경할 수 있는 커맨드

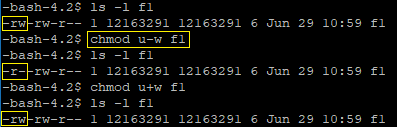
chmod의 옵션은

u=사용자 / g=그룹 / o=다른 사용자 / a=모든 사용자

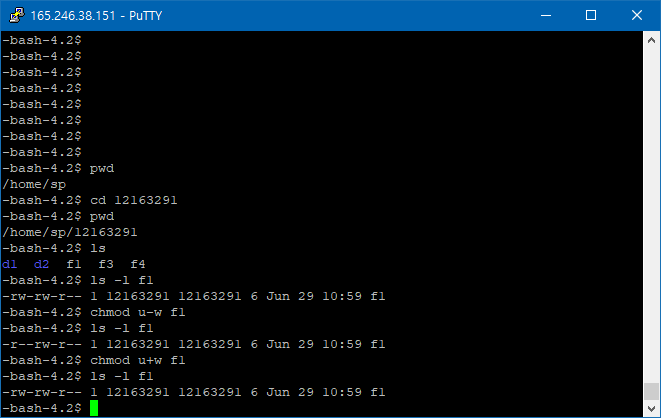
+ = 권한 부여/ - = 권한 제거 /

w = write / r = read / e = execute

로, 예를 들어 설명하자면, g+w는 그룹에 write권한을 부여해줍니다.



clear : 화면을 깨끗하게 지워줍니다.



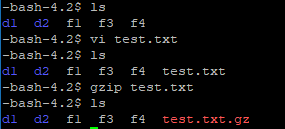


clear을 하고 난뒤의 화면입니다.

실제로 코드가 사라진 것은 아니고, 시각적으로만 clear를 해준 것입니다.

gzip: 파일을 압축하여주는 커맨드입니다.

gzip은 단일 파일 압축만이 가능합니다. 또한, 압축전의 원본 파일은 삭제되어 사라집니다.



vi test.txt를 통해 txt파일을 임의로 생성하여,

gzip 커맨드를 사용해 압축을 진행해주었습니다.

앞서 설명하였듯이, 원본 test.txt는 사라진 것을 확인할 수 있습니다.

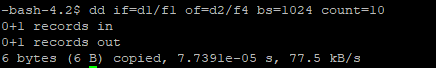
date: 날짜를 출력하는 커맨드입니다.



현재 2020년 06월 29일 4시 12분이므로, 그대로 출력된 것을 확인할 수 있습니다.

dd: 블록 단위로 파일을 복사하거나, 파일을 변환할 수 있는 명령어입니다.

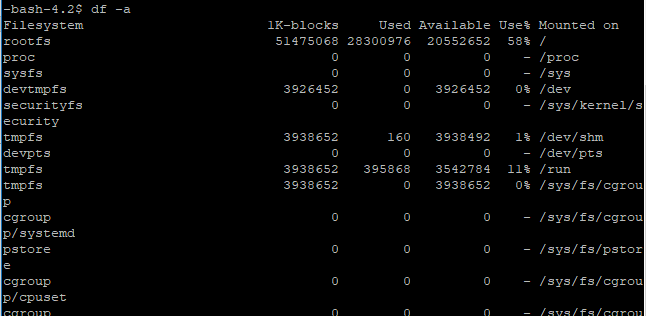
이는 예를들어 설명하겠습니다.



이를 통해 f1의 내용을 f4에 복사할 수 있게됩니다.

df: 시스템에 mount되어있는 하드디스크의 남은 용량을 확인할 때 사용하는 커맨드입니다.

기본 단위는 [1024Byte] 블록단위 입니다.



위 결과를 확인하면, 사용된 용량, 사용가능한 용량, 이의 use%, mount된 위치를 확인할 수 있는 것을 확인할 수 있습니다.

diff: 파일간의 차이점을 확인할 수 있는 명령어입니다.



위와 같이, diff 뒤에 3을 붙여주면, 3개의 파일까지 비교 가능합니다.



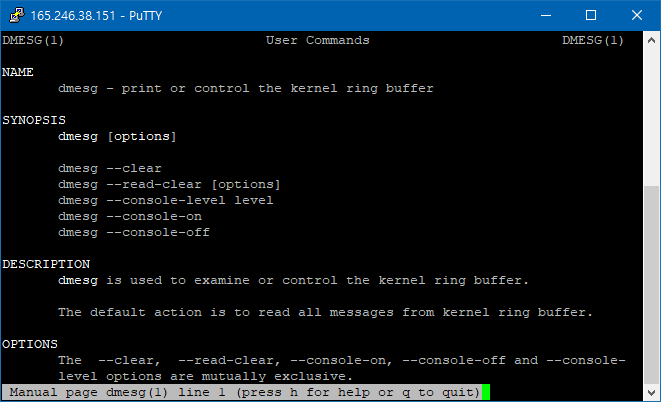
diff커맨드의 option은 위와 같습니다.

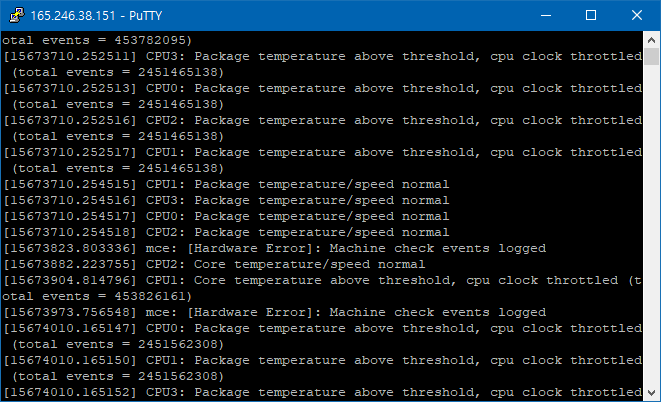


그에 따라, 현재 존재하고 있는 d1, d2를 -r 옵션을 통해 비교해보았습니다.

dmesg : 시스템 부팅 메세지를 확인하는 커맨드입니다. 또한 커널에서 출력되는 메세지를 일정 수준 기록하는 버퍼 역할을 수행하며, 커널 부팅 중에 에러가 났다면 어느 단계에서 에러가 났는지 범위를 좁히고 찾아내는데 도움이 됩니다.

설명을 읽어보았지만 감이 잘 잡히지 않아, man 명령어를 통해 확인해보았습니다.

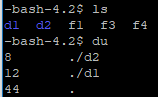




dmesg -c를 통해

모든 메시지를 삭제하고, 입력이후 새롭게 log를 쌓는 작업을 진행 하였습니다.

du : 디렉터리나 사용자별 디스크 사용량 확인시에 사용 ( 특정 디렉터리 별로 사용량을 알려줌) 하는 커맨드입니다.



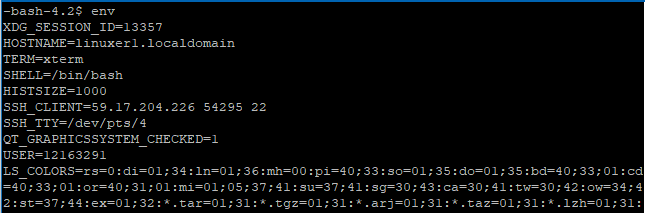
. = root/home/sh/12163291 입니다.

각 dir의 디스크 사용량을 확인할 수 있습니다.

env: 현재 지정되어 있는 환경 변수들을 출력하거나, 새로운 환경 변수를 설정하고 적용된 내용을 출력하는 커맨드 입니다.



옵션값은 위와 같습니다.

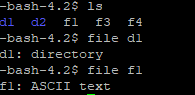


우선 compile은 해보았지만, env의 역할에 대해서는 더 공부해 보아야 할 것 같습니다.

exit : 프로그램을 종료할 수 있습니다.

그냥 putty 창이 종료됩니다.

file: 지정된 파일의 종류(타입)을 확인하는 커맨드입니다.



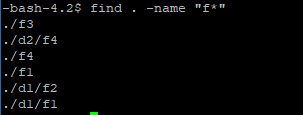
file + file이름

으로 각 파일의 형식을 확인할 수있습니다.

find: 리눅스에서 파일 및 디렉토리를 검색할 때 사용하는 커맨드입니다.



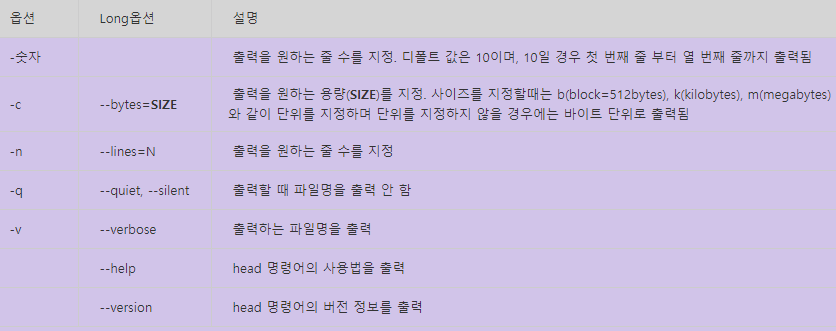
이와 같이 다양한 서치 옵션이 있습니다.



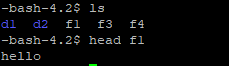
저는 . -name “f\*” 를 통해

파일’이름’이 f로시작되는 파일을 찾아보았습니다.

head: 어떤 파일의 앞부분을 보여주는 커맨드입니다.

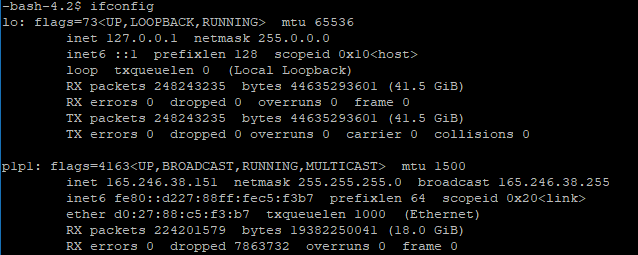


이 또한, 다양한 옵션이 존재합니다.



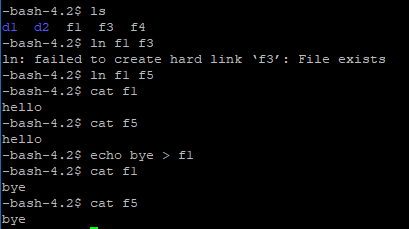
저는 f1의 head부분을 봤습니다.

ifconfig: 네트워크 인터페이스를 설정하거나 확인하는 명령어로 - IP주소, 서브넷마스크, MAC주소, 네트워크 상태 등을 확인, 설정할 수 있습니다. 현재 네트워크 구성 정보를 표시하고 네트워크 인터페이스에 IP 주소, 넷 마스크 또는 broadcast 주소를 설정하고 네트워크 인터페이스의 별칭을 만들고 하드웨어 주소를 설정하고 네트워크 인터페이스를 활성화 또는 비활성화하는 등 다양한 곳에 사용되는 커맨드입니다.



이와 같이 net의 정보가 출력됩니다.

ln: Link 의 약어로서 **리눅스** 파일시스템에서 링크파일을 만드는  **커맨더**입니다. 즉, 한 파일을 다른 파일 이름으로도 사용하고자 할 때 사용됩니다., - 링크된 파일 중 한 파일을 수정하면 다른 파일들도 수정됩니다. 연결되어있기 때문입니다.



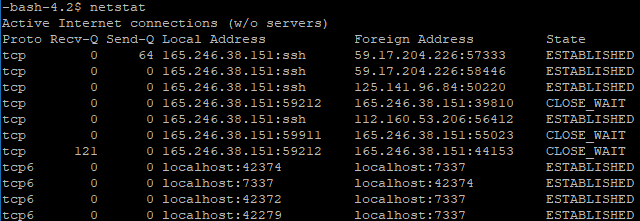
저는 f1를 f5에 이어주었습니다.

이미 있는 파일에는 이어줄 수 없습니다.

f1과 f5를 이어준 후, f1을 bye로 수정하니 f5도 동일하게 바뀌어있는 것을 확인할 수 있습니다.

mount: 우선, mount의 사전적인 의미는 – 물리적인 장치를 특정위치, 즉 디렉터리에 연결시켜 주는 것을 마운트라고 합니다. 섵불리 mount를 진행 시켜주어도 되나 확신이 들지 않아, 이는 실행해보지 않았습니다. mount에 대한 공부또한 더욱 필요할 것 같습니다.

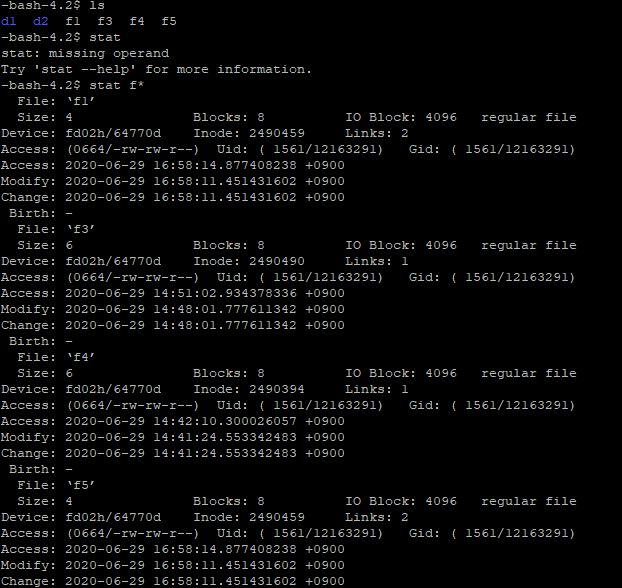
netstat: ‘net’ ‘stat’ 이라는 이름에서 유추해 볼 수 있듯이, 이는 모든 네트워크의 연결상태를 보여줍니다.



이와 같이, 네트워크의 local 주소들과 현재 상황(state)을 보여주는 것을 확인할 수 있습니다.

+ netstat -at 으로는 TCP 프로토콜만, -au로는 UDP 프로토콜만 출력할수 있는등의 옵션도 존재합니다.

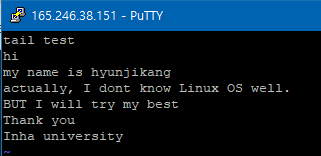
stat: 시스템 내 디렉토리나 파일의 다양한 정보를 확인할 수 있는 명령어입니다. ls와 비슷하지만 stat은 ls와 달리 요약되어있지않은, 상세한 정보입니다.



stat f\*

를 통해 f로 시작되는 파일들의 상세정보 – 수정시각등 을 확인할 수 있는 것을 확인할 수 있습니다.

tail: 파일의 내용을 뒤에서부터 출력해주는 명령어입니다. 파이프를 통해서 받은 입력을 출력하는것도 가능하며 여러가지 옵션과 함께 특정 프로세스나 파일의 로그를 볼 때 주로 사용합니다.



tail의 사용방법을 확인 하기 위해, tailtest.txt 파일을 생성해 주었습니다.



tail -n 2 파일이름 옵션을 사용하여, 위에 생성해둔 tailtest.txt파일내의

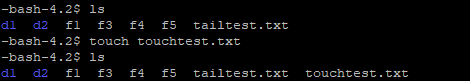
뒤에서 2줄의 내용을 출력해줍니다.

time: 어떤 파일을 컴파일 해주는데, 걸리는 시간을 구해주는 명령어입니다.



위와 같이, 컴파일을 할 때, 맨앞에 time 명령어를 넣어주면 사용할 수 있고, 그 결과 time은 컴파일을 할 떄에 걸리는 시간을 출력해줍니다.

touch: **리눅스**에서 파일을 생성하거나 갱신하는 **명령어**입니다. 새로운 파일을 만들 때는 존재하지 않는 파일명을 지정하면 지정된 파일명으로 파일이 생성되며 이미 존재하는 파일을 지정하면 파일의 수정시간이 업데이트 됩니다.



touch를 통해 touchtest.txt를 생성해주었습니다.

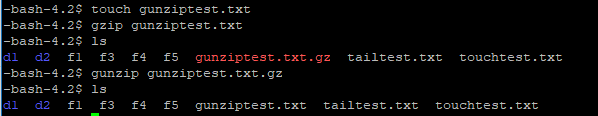
tty:  현재 사용중인 터미널의 파일명을 보여줍니다. - 현재 표준 입력에 연결되어 있는 터미널의 파일명을 보여줍니다.

이는 file tree내의 어느 위치에 있던 상관없이 동일한 값이 나오게 됩니다.

현재 연결된 ‘터미널’ 의 파일명을 보여주는 것 이기 때문입니다.

gunzip: 이는 앞서 **gzip** 형식으로 압축된 파일을 해제하여주는 **명령어**입니다.



우선 앞서 사용하였던

touch 명령어로 gunziptest.txt파일을 만들고,

gzip 명령어로 만든 텍스트파일을 압축해준 후,

gunzip 명령어로 원 상태로 되돌려 주었습니다.

whereis: 찾고자 하는 **명령어**의 실행 파일 절대 경로와 소스코드, 설정 파일 및 매뉴얼 페이지를 찾아 출력하는 **명령어**입니다.



touch라는 명령어는 위에 출력된 결과의 경로에 위치해 있습니다.

which: 특정명령어의 위치를 찾아주는 명령어입니다.



이를 통해 ls 명령어의 위치를 알 수 있습니다.

whoami: 현 system에 로그인 해 있는 사용자의 아이디를 출력하는 명령어 입니다.

