리눅스 명령어

VI 정리

:set nu => 숫자보임

0 => 해당 행의 맨앞으로 커서이동

$ => 해당 행의 맨뒤로 커서이동

dd => 행 삭제

yy => 행 복제

:1,10y => 1~10줄까지 복제

:1,$y => 전체 복제

Shift + V => 커서가 있는 라인 선택

+G => 전체 선택

p => 붙여넣기

/찾을문자 => 문자찾기

N => 이전 문자로 이동

n => 다음문자로 이동

H => 전체 텍스트의 가장 상단으로 커서이동

L => 전체 텍스트의 가장 하단으로 커서이동

M => 전체 텍스트의 중간으로 커서이동

<https://maeuminpaper.tistory.com/100>

vi로 보는데 한글깨질때..

:set encoding=utf-8

iptables 정리

포트 열려잇는지 확인

$nc -z 호스트주소 포트

Telnet 없을때.. 인터넷 연결안되거나 혹은 설치하면 안될때..

curl -v telnet://ip:port

포트 열기

$iptables -A INPUT -p tcp —dport 8888 -j ACCEPT

-A: append(방화벽 규칙을 추가)

INPUT : Inbound

-p tcp : tcp 규칙 추가할것

—dport : destination port

-j ACCEPT : Accept 규칙을 추가.. REJECT,DROP 이 있음..

<https://ienjoy.tistory.com/29>

<https://khie74.tistory.com/1169521441>

find grep awk

<https://jybaek.tistory.com/704>

find로 하위폴더중 depth를 지정해서 긁어올수있음

ex) 현재폴더에서 depth 2까지 file인 것들은 ABC라는 폴더 안에다가 이동시켜라

find ./ -maxdepth 2 -type f -exec mv ‘{}’ ABC/ \;

grep 정리

<https://recipes4dev.tistory.com/157>

<http://webterror.net/?p=1721>

grep 으로 검색했는데

“Binalry file 파일이름 matches” 나올때 해결방법..

=> -a 를 옵션에 넣어주면끝!

Binary가 파일에 포함되어있으면 이렇게 나온다고하는데.. -a를 이용해서 강제적으로 보여줄수있게 가능

<https://junho85.pe.kr/434>

du -h (해당폴더 사용용량.. 하위 디렉토리까지 다나옴)

df -h (파티션별 사용용량)

nohup을 사용하면 사용자가 터미널을 종료해도 프로그램이 계속 살아있게 된다.

1. Nohup

\* 정의 : 리눅스, 유닉스에서 쉘스크립트파일(\*.sh)을 데몬형태로 실행시키는 프로그램

\* Nohup은 리눅스에서 쉘스크립트파일을 데몬형태로 실행시키는 명령어이다.

- nohup으로 실행을 시키려면 실행파일 권한이 755이상으로 되어있어야 함

- 명령어 뒤에 '&'를 추가하면 백그라운드로 실행됨

- nohup 을 통해 프로그램을 실행시키면 nohup.log 라는 로그 파일 생성

$nohup [실행파일]

$nohup [실행파일] & // 백그라운드 실행

2. 로그 안남기기

$nohup [실행파일] 1>/dev/null 2>&1 &

1. /dev/null 이 표현은 1의 결과를 /dev/null 이라는 파일 속에 넣는다.

/dev/null로 보내버리면 모든 출력을 없애버린다.

2. &1 이 표현은 2번 파일디스크립터를 1번에 지정된 형식과 동일하게 /dev/null로 지정한다.

& 은 프로그램을 백그라운드에서 실행하도록 하는 표현이다.

3. nohup 종료하기

1. "ps -ef | grep 쉘스크립트파일명" // 명령으로 데몬형식으로 실행

2. "kill -9 PID번호" // 명령으로 해당 프로세스 종료

출처: https://jasontody.tistory.com/113 [p]

Sudo : super user do.. 현재계정에서 잠시 루트 권한을 빌리는것

Su [user] : switch user 계정전환.. (환경변수는 이전 계정으로)

Su – [user] : 계정전환 + 환경변수 또한 전환

Iptables 정리 : <https://webdir.tistory.com/170>

tcpdump : 리눅스에서 송수신되는 데이터들 보고싶을때사용

-특정인터페이스로 통하는 데이터들도 볼수있고

-특정포트 or 주소로 나가거나

-특정포트 or 주소로 들어오거나

-등등 다양하게사용가능

https://moyaria.tistory.com/809

시스템로그확인방법 : https://m.blog.naver.com/PostView.nhn?blogId=kdi0373&logNo=220522832069&proxyReferer=https:%2F%2Fwww.google.com%2F

<https://yjshin.tistory.com/entry/%EC%8B%9C%EC%8A%A4%ED%85%9C%EB%B3%B4%EC%95%88-%EB%A6%AC%EB%88%85%EC%8A%A4-%EB%A1%9C%EA%B7%B8%ED%8C%8C%EC%9D%BC-%EB%B0%8F-%EC%84%A4%EC%A0%95%ED%8C%8C>

lsof : lsof는 list open files의 약자로 시스템에서 열려있는 파일에 대한 정보를 출력해주는 명령어다. (대략 '엘에스오브'라고 발음하는 것 같다) 리눅스와 유닉스는 일반 파일과 디렉토리, 소켓, 파이프, 블록 디바이스, 캐릭터 디바이스에 대한 관리를 파일 시스템을 통해서 할 수 있다. 따라서 특정 프로세스가 열고 있는 파일 디스크립터(fd)를 조회해보는 행위는 시스템을 모니터링하는데 굉장한 도움이 된다.

유닉스 계열의 운영체제에서는 프로세스가 열고 있는 파일을 지웠을 때, 파일 시스템에서 ls로 보이지는 않지만 reference count가 0이 아니기 때문에 실제로 지워지지 않고 남아있는 경우가 더러 있다. 이 때, 프로세스가 열고 있는 파일이 위치한 디스크를 정상적으로 언마운트할 수 없기 때문에 누가 쓰고 있는지 조회해야하는 경우가 있다. 이 때, lsof를 사용하면 쉽게 찾을 수 있다.

<https://dev.plusblog.co.kr/44>

lsof -P -n -i TCP

lsof -P -n -i TCP:포트번호

-P : /etc.services 에 등록되어있는 이름대신 포트번호로 출력

-n : 호스트 이름 대신에 IP address 출력

-i : 네트워크 소켓에 대한 정보를 출력

세부명령정리 굿

<https://blog.cafe24.com/1989>

java version 변경(centos7)

<https://holics1226.tistory.com/24>

<https://linuxhint.com/install_jdk12_centos7/>

ln(링크 정리)

<https://eunguru.tistory.com/90>

중복데이터 찾아내기

Uniq -c

<http://bahndal.egloos.com/576672>

/bin 과 /user/bin 은 도대체뭐고 차이가뭔지에 대한설명

-모든 사용자가 특정 프로그램을 사용할때 어떤 디렉토리에서든 사용할수있도록 bin폴더 안에넣어놓으면 사용가능함.. 환경변수 셋팅이랑 비슷한느낌..?(찾아보니 환경변수에 /bin /sbin /usr/bin 등등의 경로가 이미 지정되어있네) 하는 역할은 비슷하나 경로에따라 관리측면에서 차이가있음

<https://wookiist.tistory.com/10>

pipe(|) redirect(>) 차이 정리굿..

<https://twpower.github.io/133-difference-between-redirect-and-pipe>

trap 으로 SIGNAL을 catch하여 처리하는방법

-ctrl+c => SIGINT (사용자의 종료)

-정상종료 => SIGTERM

=> 둘다 catch 가능,,

=> trap 할때는 둘다 넣어줘야겟다..

SIGINT,SIGTERM 차이 정리 : https://2kindsofcs.tistory.com/53

trap 관련 정리 : <https://pinocc.tistory.com/153>

SIGNAL 관련 정리 : https://jhnyang.tistory.com/143

리눅스 폴더 구조 설명 굿 :

1. home   
 - 리눅스 사용자들의 개인 공간  
 - 보통 일반 사용자는 home 밑에 별도의 설정이 없는 경우 자신의 아이디로 된 디렉토리가 배정됨(로그인 시 처음 작업 디렉토리가)  
 - /root는 root 의 홈 디렉토리  
   
2. bin & bash   
 - sbin 은 시스템 관리를 위한 명령들 => root에게만 허용됨  
 - echo $PATH 로 확인  
  
3. lib  
 - 리눅스 상에서 자주 사용되는 라이브러리를 모아놓음  
 - Shared object file format(.so)  
 - 윈도우에서의 Dynamic Linked Library(DLL)을 모아놓은 폴더로 생각할 수 있음  
  
4. etc  
 - 주요 설정 파일들이 위치한다  
 - 주요 리스트   
  \* motd : 로그인 성공 시 처음에 띄워주는 문구  
  \* passwd : 리눅스 사용자에 대한 정보(예전에는 사용자의 비밀번호(!) 도 있었음)  
  \* shadow : 리눅스 사용자 패스워드 정보 ( root만 볼수 있음)  
  \* sudoers : sudo 권한을 가진 사용자와 권한에 대한 정보  
  \* fstab : 파일 시스템과 마운트 포인터  
  \* hosts : 도메인 네임  
  \* issue : 리눅스 배포반 및 버젂  
  \* skel/ : 새로운 사용자가 추가될 때 홈 디렉토리에 기본적으로 복사해줄 파일들을 포함한 디렉토리  
  \* init.d/ : 컴퓨터가 부팅될 때 실행될 파일들을 넣어둔 디렉토리  
  
5. proc  
 - RAM 위에 존재하며 커널에 의해 제어됨  
 - 현재 실행되는 프로세스와 실제로 사용되는 장치, 커널이 수집한 하드웨어 정보가 저장됨  
 - 사용자가 /proc이나 하위 파일에 접근할 때마다 커널에서 파일내용을 동적으로 만들어냄  
 - 각 프로세스는 고유의 ID를 가지고 있으며(숫자) 이 아이디를 가진 디렉토리 밑에 각종 정보를 저장함  
  \* cmdline : 프로세스를 시작한 명령행 내용  
  \* status : 프로세스의 내부 상태 정보  
  \* cwd : 프로세스의 현재 작업 디렉토리  
 - /proc/meminfo : 총 메모리 사용 현황  
 - /proc/stat : 시스템의 상태에 곾한 정보  
 - /proc/uptime : 시스탬이 부팅된 후 흐른 시간  
 - /proc/version : 현재 실행되는 커널 버전  
  
6. dev  
 - 실제로 하드디스크에 존재하지는 않는 정보를 저장함  
 - 모든 하드웨어( hdd,cdrom, 마우스, 가상 장치,etc…) 를 파일로 인식하며 이를 장치 파일이라 함  
 - psaux PS/2 마우스 장치  
 - tty : teletypewritter, 사용자와 시스템 사이를 중계하는 역할  
 - 사용자와 시스템 사이의 입출력은 모두 여기를 통해 이루어짐  
 - 서버 시스템에 직접 연결된 장치에서 사용함  
 - pts: telnet, ssh 등을 이용해 원격으로 접속할 경우 (pseudo TTY slave)  
 - 유사 장치 : 실제 장치와 곾렦이 없음  
 - null : 이 장치로 들어갂 데이터는 모두 사라짐  
 - zero : 이 장치에서는 항상 null을 반환   
  
  
7. mnt & media  
 - 마운트: 한 파일 시스템의 루트 디렉토리를 다른 디렉토리에 붙임으로써 디렉토리를 사용할 수 있게 만들어주며, 모든 파일 시스템들을 마치 그들이 속해있는 파일시스템의 서브 디렉토리인 것처럼 사용 가능하게 만드는 것  
 - 이제는 많은 저장 매체들이 자동으로 마운트 되기 때문에 주로 /media(특히 우분투) 를 사용하는 경우가 많지만 mnt 를 사용하는 것은 자유  
  
8. tmp  
 - 실행 중인 프로세스들의 임시 파일들을 저장  
 - 종료 시 이 위치에 있는 파일들은 모두 삭제됨  
  
9. boot  
 - 부팅시에 필요한 필수 파일들이 저장됨  
 - lilo, grub 등의 부트로더와 커널이 위치함  
 - 부트로더: 컴퓨터를 켰을 때 가장 먼저 실행되는 프로그램  
  
10. var   
 - 리눅스 상에서 자주 변경되는 데이터들만을 모아 놓음  
 - /var/lib: 일반적인 시스템 운용시 계속 갱신되는 파일들을 위한 공간  
 - /var/local: /usr/local 아래에 설치된 프로그램들의 다양한 데이터가 보간  
 - /var/lock : 잠금 파일(lock file)이 있는 곳이다.  
 - /var/log : 다양한 프로그램들의 로그 파일  
  -/var/log/wtmp : 시스템의 모든 로그인, 로그아웃 정보를 기록  
 - /var/log/messages: 커널과 시스템 프로그램들의 모든 메시지  
 - /var/run : 시스템의 현재 정보, 부팅 시 리셋  
 - /var/run/utmp : 현재 로그인한 사용자들에 대한 정보  
 - /var/spool : 대기 상태에 있는 작업들을 위한 디렉토리  
 - /var/tmp : /tmp에 있는 임시 파일들보다는 좀 더 오래 유지될 필요가 있는 임시 파일들, 부팅 시 지워지지 않는다.   
   
11. usr  
 - 리눅스 상에서 가장 큰 공간을 사용하는 디렉토리  
 - 어플리케이션 수준의 프로그램들이 여기에 있음  
 - usr 밑의 bin, lib,… 등은 사용자가 자유롭게 다룰 수 있으 며 루트 디렉토리의 같은 이름을 가진 디렉토리와 같은 역할  
을 한다. ( 대신 우선순위가 떨어진다.)  
  
 - usr/bin : 실행 가능한 명령어, 주로 콘솔이나 X에서 사용되는 명령어들  
 - usr/sbin : 시스템 곾리를 위한 명령어, 주로 서버 곾리용  
 - usr/lib : 프로그램과 하위 시스템을 위한 라이브러리  
 - /usr/man, /usr/info, /usr/doc :각각 매뉴얼 페이지, GNU Info 문서들, 그리고 기타 다른 문서  
 - usr/include : C 를 위한 헤더 파일  
 - usr/X11R6 : X window  
 - usr/src : 시스템에 빌드하는 프로그램의 소스  
 - usr/share : 읽기 전용 자료, 주로 매뉴얼이나 문서 자료  
 - usr/local : 리눅스에서 필수적인 파일이나 배포판에서 확장되는 파일들을 제외한 응용 프로그램 파일들을 저장  
 - /bin : 문서에 명시되어 있는 명령어 및 필수적인 명령어  
 - /usr/bin : 주로 콘솔과 X에서 사용되는 명령어  
 - /usr/local/bin : 그 외 어플리케이션에 의해 추가되는 명령어

ps. $PATH 환경변수에 들어있는 실행파일 폴더들

/sbin  
/bin  
/root/bin  
/usr/sbin  
/usr/kerberos/sbin  
/usr/kerberos/bin  
/usr/local/sbin  
/usr/local/bin  
/usr/bin

출처: [https://blackpigstudio.tistory.com/entry/리눅스-폴더구조-및-용도](https://blackpigstudio.tistory.com/entry/%EB%A6%AC%EB%88%85%EC%8A%A4-%ED%8F%B4%EB%8D%94%EA%B5%AC%EC%A1%B0-%EB%B0%8F-%EC%9A%A9%EB%8F%84) [BlackPigStudio]

Awk 명령어 관련 정리 굿

<https://recipes4dev.tistory.com/171>

리눅스에서 process별 메모리 사용량 표시

- RSS로 확인가능

- RSS(Resident set size) : 물리 메모리를 실제 점유하고 있는 크기

- 단위는 KB임  
- 명령인수 포함

ps -eo user,pid,ppid,rss,size,vsize,pmem,pcpu,time,cmd --sort -rss | head -n 11

- 명령인수 미포함

ps -eo user,pid,ppid,rss,size,vsize,pmem,pcpu,time,comm --sort -rss | head -n 11

설명굿

<https://zetawiki.com/wiki/%EB%A6%AC%EB%88%85%EC%8A%A4_%EB%A9%94%EB%AA%A8%EB%A6%AC_%EC%82%AC%EC%9A%A9%EB%9F%89%EC%88%9C_%ED%94%84%EB%A1%9C%EC%84%B8%EC%8A%A4_%EB%B3%B4%EA%B8%B0>

원격지에 있는 데이터 가져오거나 올리는 명령어

SecureCopy(scp)

<https://wlsvud84.tistory.com/11>

파일을 읽을때는 vi 보다 less 이용하자 (vi는 파일크기만큼 메모리차지함.. 반면less는 랜덤엑세스 방식으로 파일을 읽기 때문에 명령어 입력 시 파일이 즉시 열리게 된다.  
그리고 메모리 사용량이 높지도 않다.

)

less 로 문서 열었을때 사용가능한 명령어

PAGE UP 또는 b : 한 페이지 위로

PAGE DOWN 또는 Space bar : 한 페이지 아래로

위 방향키 : 한 줄 위로

아래 방향키 : 한 줄 아래로

1G 또는 g : 텍스트 파일의 처음 부분으로 이동

G : 텍스트 파일의 마지막 부분으로 이동

/문자열 : 입력된 문자열 찾기

n : 이전 검색어의 다음 찾기

h : 도움말 보기

q : 프로그램 종료

<http://1004lucifer.blogspot.com/2016/07/linux-vi-less_29.html>

http://www.incodom.kr/Linux/%EA%B8%B0%EB%B3%B8%EB%AA%85%EB%A0%B9%EC%96%B4/less

리눅스/맥 커맨드라인 셸 입문자를 위한 생존 가이드

<https://www.44bits.io/ko/post/linux-and-mac-command-line-survival-guide-for-beginner>

if 없이 한줄로 명령어 분기처리하기

[ condition ] && true || false

http://soopsaram.com/documentudy/2021/10/10/shell-script-without-if-else/#:~:text=IF%EC%97%86%EC%9D%B4%20%EC%BB%A4%EB%A7%A8%EB%93%9C%20%EB%B6%84%EA%B8%B0%20%EC%B2%98%EB%A6%AC,-cmd1%EC%9D%B4%20%EC%A0%95%EC%83%81%EC%A0%81%EC%9D%B8&text=%ED%95%98%EC%A7%80%EB%A7%8C%20%26%26%20%EC%99%80%20%7C%7C%20%EB%A5%BC,%EC%BD%94%EB%93%9C%EA%B0%80%20%ED%9B%A8%EC%8B%A0%20%EA%B0%84%EA%B2%B0%ED%95%B4%EC%A7%84%EB%8B%A4.&text=%EB%A7%8C%EC%95%BD%20cmd1%EC%9D%B4%20%EC%84%B1%EA%B3%B5%ED%96%88%EC%9D%84,%ED%95%9C%EC%A4%84%EB%A1%9C%20%EC%B2%98%EB%A6%AC%20%EA%B0%80%EB%8A%A5%ED%95%98%EB%8B%A4.

nobdy 계정이란 뭐고 왜 사용되는지..?

<https://askubuntu.com/questions/329714/what-is-the-purpose-of-the-nobody-user>

alternatives 명령어

이를 사용하면 링크를 생성하는데, 링크의 대상이 되는 녀석들을 옵션으로 여러개 저장해놓았다가 필요할때마다 바꿔가며 쓸수있도록해준다.. 좋네

<https://keichee.tistory.com/100>

netstat 의미설명 굿

<https://blog.voidmainvoid.net/201#:~:text=netstat(network%20statistics)%EB%8A%94%20%EC%A0%84%EC%86%A1,%EB%B3%B4%EC%97%AC%EC%A3%BC%EB%8A%94%20%EB%AA%85%EB%A0%B9%20%EC%A4%84%20%EB%8F%84%EA%B5%AC%EC%9D%B4%EB%8B%A4>.

grep으로 하위디렉토리 검색

$ find . -name "\*" | xargs grep -n "찾고자 하는 문자열"

$ find . -name "\*.[ch]" | xargs grep -n "찾고자 하는 문자열"

xargs 는 앞선 명령어의 결과를 뒤의 명령어와 이어주는역할

https://skylit.tistory.com/81