Linux >

Linux Cheat Sheet

Contents

BASH

syntax, use 'echo' + keyword

```
$$ - PID of current shell
$0 - show shell name
$! - PID of last background cmd
$? - exist status of last cmd
$_ - previously created dir (mkdir foo && cd $_)
$@ - show all command's parameters
$# - show # of arguments passed to command
!! - run previous command
-eq - math equal (int)
-ne - math not equal (int)
-lt - math less than
-le - math less than or equal
-gt - math greater than
-ge - math greater than or equal
```

Bash tricks

create 25 new files from one command, use: {1..X}

```
touch myfile{1..25}
```

Run a specific cmd from history

```
history
120 cat /var/log/messages
121 vi /etc/hosts

!120
## will show /var/log/messages
```

configure Bash alias:

```
vi ~/.bashrc
```

```
-z - string is 0 length (null)
-n - string is not 0 length (not null)
-nt - newer than (file or object time)
(-r,-w,-x) - if object is readable,writable,exec
```

locate any related file, binary or executable anywhere on system
locate java

show environment path to executable
```
which java
```

show environment
```
env
printenv
echo $PATH
```

calculate value
```
echo $((35+15))
50
```

Key/Value pairs (associative Array)
read in config file, check array of key/val pairs to make sure parameters are set

```
$config = "/etc/file.conf"
declare -A myList=( [first]=
                    [last]=
                    [age]=
                  )

for param in "${!myList[@]}"; do
    value=$(grep ^$param $config)
    var[$param]=$value

     if [ -z ${myList[$param]} ];
     then
        echo "param $param is not set"; exit 1
     fi
done
```

Dictionary/Hash in bash (parse IP + Port hash, netcat to each IP and port)

```
  readonly connections=(

  'A,205.209.202.37,7755'
  'B,205.209.202.1,8899'
  'C,205.209.202.21,4578'

  )

  function nctest(){
    local name ip port
    for fields in ${connections[@]}
    do
      IFS=$',' read -r typ name ip port <<< $fields

      conn=$(nc -zv -w 2 $ip $port 2>&1 | grep 'Connection
  refused' )

      if [[ -z "${conn}" ]]
      then
        echo "[$name] nc $ip $port OK"
      else
        echo "[$name] nc $ip $port REFUSED"
      fi

      echo "--------------"
    done
  }

  nctest
```

Command Alias
edit ~/.bashrc
```
sshx() {
  ssh your.name@$1".corp.com"
}

sshx nysql01
```

add commands alias
alias ls='ls -lta --color=auto

get file extension
```
file=superman.jpg
name=${file%.*}   # superman
ext=${file#.*}  # jpg
```

delete all files that dont match an extension
```
rm !(*.foo|*.bar|*.baz)
```

strip off the last character from a string,
```
var="Banana"
echo ${var%?}  // Banan
```

run python inside Bash with arguments

```
  function print_hello {
  NAME="${1}" python - <<END
  import os
  print("Hello there %s" % os.environ['NAME'])
  END
  }

  print_hello Joe
```

delete files from search
find . -name '*.pyc' -delete

Debugging
enable line by line processing output in script
```
set -x
```

verbose output only if debug flag is set
```
debug=1
test $debug -gt 0 && echo "var is $var"
```

Test Command
test if file exists, if not, exit with error
```
test -f "${config}" || { echo "${config} not present,
exiting.."; exit 1; }
```

check if parameters are set and not empty, exits out w error if not set
```
err_msg="[ERROR] parameter is not set or empty value:"
myParam=${1:?"$err_msg myParam"}
```

get location of script from the script itself (pwd doesnt work here)
```
dirname "$0"
```

get variable from a json dump using python
```
URL=$(echo ${URL} | python -c 'import sys,json; print
json.load(sys.stdin)["url"]')
```

colorize Bash prompt (insert into ~/.bashrc)
```
export PS1="[\[\e[31m\]\u\[\e[m\]\[\e[33m\]@\h\[\e[m\]:\W]$
"
```

Root PS1
```
export PS1="[\[\e[30;41m\]\u\[\e[m\]\[\e[33m\]@\h\
[\e[m\]:\W]$ "
```

get # of characters in variable
```
var="milkshake"
echo ${#var}
9
```

convert uppercase files to lowercase
```
rename 'y/A-Z/a-z/' *
```

Case statement (check input params)

```
  case $key in
  -u| -username | --username)
     UNAME="$2"
   shift
   ;;
  -pw| -password | --password)
     PASSWORD="$2"
   shift
```

JQ - json parser

show all values
```
curl xxxx -X GET | jq .
```

show specific key
```
echo $json | jq '.values'
```

parse array key for specific value
```
echo $json | jq '.values[].title'
```

Convert YAML to JSON - 1 liner
```
python -c 'import sys, yaml, json; json.dump(yaml.load(sys.stdin),
sys.stdout, indent=4)' < file.yaml > file.json
```

Check if Word is in a String
```
[[ "$string" == *"$word"* ]] || echo "word not in string"
```

Get Date in specific format
```
echo $(date +%Y%m%d_%H%M%S)
```

redirect std output to both file and screen
```
program [arguments...] 2>&1 | tee outfile
```

extract filename from a path
```
echo /somedir/blah/postgresql96-9.6.5.x86_64.rpm | awk '{match($1,
"[^/]*$", a)}END{print a[0]}'

postgresql96-9.6.5.x86_64.rpm
```

Insert string after a delimeter, save in place (insert "dog" after "cat")
```
tmpfile=$(mktemp)
awk '/cat/ { print; print "dog"; next}1' pets.txt > $tmpfile && mv
-f $tmpfile pets.txt
```

add timestamp to a tail of log file
```
tail -f /var/log/messages | while read ; do echo "$(date +%T.%N)
$REPLY" ; done
```

create a random large 200MB file,
```
dd if=/dev/urandom of=file.txt bs=2075200 count=100
```

create 500gb file
```
dd if=/dev/zero of=/opt/testfile bs=1G count=500
```

copy all ssh keys for every user from 1 host to another
```
host1> for i in $(ls /home);do rsync -azP /home/$i/.ssh/id_rsa*
host2:/home/$i/.ssh/ ;done
```

```
  ;;
 -p| -profile | --profile)
    PROFILE="$2"
  shift
  ;;
 *)
  echo "Unknown Option"
 exit 1
```

Conditionals:

for loop
```
for loop in {1..50};do echo processing $loop; sleep 2; done
processing 1
processing 2
etc
```

or use a 'seq' operand
```
for i in $(seq 1 5); do echo $i; done
1
2
3..etc
```

loop thru directories and grep something from config files
```
for i in $(ls); do grep 'something' $i/*.conf ; done
```

Grep

grep either or
```
netstat -an | grep '8000\|8050'
```

search for a pattern in all files
```
grep -RnisI "My cat*" /var/log/*
```

grep 5 lines above and below result
```
cat file.txt | grep -A 5 'blah'
```

Get value between 2 delimiters,
```
grep ExecStart bitbucket.service | awk -v FS="
(bitbucket/|/bin)" '{print $2}')
```

remove all blank lines from a file
```
grep . file1 > file2
```

VIM

delete all lines from file
```
:1,$d
```

search for all instances of string 'horse'
```
escape key
/horse
press 'n' to move to next occurence
```

vi a file on remote server
```
vi scp://user@<hostname>//etc/hosts
```

SSH

file permissions
```
/home/user = 700
/home/user/.ssh = 700
/home/user/.ssh/id_rsa = 600
/home/user/.ssh/id_rsa.pub = 644
/home/user/.ssh/authorized_keys = 600
/home/user/.ssh/known_hosts = 644
```

troubleshoot auth errors
on target (where youre trying to ssh into), start SSH on different port, debug mode
```
/usr/sbin/sshd -d -p 2222
```

on client, connect to target
```
ssh user@target -p 2222 -vvv
```

SSH Shuttle
yum install sshuttle

route all connections to 172.31.23.156 via "server B"

```
sshuttle -r user@<server B IP> 172.31.23.156
```

all connections will now be going via remote IP, encrypted
to route ALL connections, use 0/0
sshuttle -r user@serverB 0/0

proxy connections for a specific website,via jump host serverB, send to background
```
serverA> nohup sshuttle -r serverB `dig +short
www.somesite.com | sed "/[^0-9\.]/d" | xargs -n1 -I '$'
echo -n '$/32 '` 2>&1 &
```

pass a custom SSH key to sshuttle
```
sshuttle --dns user@host <IP range> --ssh-cmd 'ssh -i
/home/user/priv_key'
```

use SSH as a web proxy
```
ssh -D 8080 username@proxyHost
```

set browsers proxy option to 127.0.0.1:8080, all browsing requests will go via proxyHost

check what pub key matches the priv key
```
ssh-keygen -y -e -f ~/.ssh/id_rsa
```

add a new SSH key and copy the public key to remote known_hosts file
```
ssh-keygen -t rsa
cat ~/.ssh/id_rsa.pub | ssh user@hostname 'cat >>
.ssh/authorized_keys'
```

run a command on remote host
```
ssh servername cmdname
```

connect to an unreachable server B (port 2345) via SSH hop over reachable server A
```
ssh user@serverA -L 6789:serverB:2345 -f -N  (localhost:6789 =
serverB:2345)
```

Port Tunneling via SSH
(port 1200 is unreachable from server A, connect to it via localhost:1300 via SSH to server B

```
user@serverA> ssh -L 1300:localhost:1200 serverB -fN
```

setup SSH Sockets
```
mkdir ~/.ssh/sockets
vim ~/.ssh/config
```

```
  UseRoaming no
  TCPKeepAlive yes
  ServerAliveInterval 15
  ServerAliveCountMax 6
  Host *
  Compression yes
  ControlMaster auto
  ControlPath ~/.ssh/sockets/%r@%h:%p
  ControlPersist yes
  ControlPersist 600

  Host nycweb1
  Hostname 192.168.10.2
  User root
  IdentityFile ~/.ssh/id_rsa
```

show fingerprint of a public key file, useful to track down /var/log/secure messages to see who logged in

```
ssh-keygen -lf /home/user/.ssh/authorized_keys | grep
<fingerprint> (looks like SHA256:zZUd2W..etc)
```

Memory / Diagnostics

Debian - CPU and Mem
```
lshw -html > /tmp/specs.html
```

Fedora - CPU and Mem
```
cat /proc/cpuinfo
cat /proc/meminfo
lspci -v
```

Show memory usage
```
free -m
```

Show all current users logged in
```
who
w
```

Send msg to all logged-in users
```
wall -n "hello"
```

Show all loaded modules
lsmod

Show processes by memory usage
```
ps aux | awk '{print $6/1024 " MB\t\t" $11}' | sort -n
```

Monitor I/O usage
vmstat 1 20 // runs vmstat every 1 sec, 20 times

show USB info
```
lsusb -v
```

show size of folder
```
du -sh
```

Swap

clear swap space
```
swapoff -a  (wait till clears)
swapon -a
```

Journalctl

tail a log for a process
```
journalctl -u httpd -f
```

show last 100 lines for a process
```
journalctl -u httpd --no-pager -n100
```

DMIDecode

show bios
```
dmidecode -t bios
```

system info
```
dmidecode -t system
```

chassis
```
dmidecode -t chassis
```

memory, processor, slot
```
dmidecode -t memory
dmidecode -t processor
dmidecode -t slot
```

serial #
```
dmidecode -s
```

insert, remove mod
insmod fat
rmmod fat

Show current runlevel
runlevel

Show all device and hardware information log
dmesg

show IRQ drivers being used
```
cat /proc/interrupts
```

show DMA channels being used (comms between I/O ports)
```
cat /proc/dma
```

show I/O ports being used
```
cat /proc/ioports
```

Stress Testing
```
yum install stress-ng
```

run stress on 2 CPUs
```
stress-ng --cpu 2 --timeout 10s --metrics-brief
```

force Out of memory kill
```
stress-ng --vm 5 --vm-bytes 95% --vm-method all --verify -t 1m -v
```

Stress I/O load, run 5 workers that will continually R/W to temp file
```
stress-ng -d 5
```

Run application with memory limit
systemd-run --user -p MemoryLimit=3G google-chrome

Kill frozen process
```
Alt + PrintScreen + f
```

Network

IP command

show all interfaces	`ip a`
show specific interface	`ip addr show dev em1`
assign address to interface	`ip addr add 192.168.5.2 dev em1`
show only active interfaces	`ip link ls up`
bring up an interface	`ip link set dev em1 up`
disable an interface	`ip link set dev em1 down`

rename inteface w/o network restart	`ip link set dev em1 down`
	`ip link set em1 name eth1`
	`ip link set eth1 up`

delete interface	`ip link delete em4`
bring up an interface	`ip link set em1 up`

Check Traceroute and Ping at same time, live stream
```
mtr www.google.com
```

```
                                        Packets
Pings
 Host                                                 Loss%   Snt
Last   Avg  Best  Wrst StdDev
 1. 10.24.11.1                                       0.0%    12
17.6  15.6  13.2  19.5   1.9
 2. 209.95.50.1.static.midphase.com                  0.0%    12
11.9  18.8  11.9  38.0   6.6
 3. 173.244.223.117.static.midphase.com              0.0%    12
12.9  28.1  12.9 111.0  27.2
 4. 173.244.202.29.static.midphase.com               0.0%    12
17.7  27.8  12.1 119.3  29.2
```

Check Port communication

change MTU on interface	`ip link set em1 mtu 9000`
see all routes	`ip route` or `route -n`
get route for an IP	`ip route get 192.168.1.2`
delete route	`ip route del 192.168.1.2`
add a new route via gateway	`ip route add 192.168.1.2 via 192.168.1.1 dev em1`
add default route	`ip route add default ia 192.168.1.1 dev em1`
show all tunnels	`ip tunnel`

Trace # of hops for HTTP request
```
traceroute 123.123.21.2
```

check kernel actions during network service start
```
dmesg
```

Check if port 120 is open and listening
```
netstat -an | grep 120
```

check user, PID listening on port 8080
```
ss -ap4 | grep 8080
```

TCP Dump

show all interfaces tcpdump can listen on
```
tcpdump -D
```

listen on specific interface
```
tcpdump -i eth0
```

listen on all ifaces
```
tcpdump -i any
```

listen on specific port or portrange
```
tcpdump portrange 3334  (3334-3380 range)
```

record packet capture into a .cap file
```
tcpdump -w capture.cap
```

read contents of a .cap file
```
tcpdump -r capture.cap
```

display only IP address and ports instead of hostnames
```
tcpdump -n
```

display only where destination IP is 192.168.5.1 (for source use -n src)
```
tcpdump -n dst host 192.168.5.1
```

capture TCP packets where port is between 1 and 1023
```
tcpdump -n tcp dst portrange 1-1023
```

capture packets where destination host is 192.168.5.1 and port is 5049
```
tcpdump -n "dst host 192.168.5.1 and dst port 5049"
```

Check MAC address mapping to IP
```
arp
```

IP Routing Table
```
route
```

add a new route,
```
ip route add 118.100.1.173 via 192.168.38.17 dev p1p2 metric 200
```

add new route permanently
```
vim /etc/sysconfig/network-scripts/route-p1p2
201.224.250.40 via 192.168.38.33 metric 200
```

delete a route
```
ip route del 118.100.1.173
```

modify existing route

find process thats holding a certain port #
```
netstat -tulpn | grep 5000
```

Netcat

Chat client
```
On Server - start NC session
hostA: nc -l 9933

on Client, connect to NC session
hostB: nc hostA 9933

can type messages between servers like chat client
```

start a Netcat Bash session (ghetto SSH)
```
serverA> nc -l 5000 -e /bin/bash
serverB> nc serverA 5000
```

Netcat Ghetto web server
```
while true ; do nc -l -p 1500 -c 'echo -e "HTTP/1.1 200 OK\n\n $(date)"' ; done
```

Scan a range of IPs for an open port,
```
for i in {1..25};do nc -zv 208.224.251.$i 8003 -w 2 ;done
```

Spin up a webserver with custom port, check that you can connect to port
```
serverA> python -m SimpleHTTPServer 8331
serverB> nc serverA 8331
```

connect on a UDP port
```
nc -u <hostname> <port> -vv
```

transfer files between 2 hosts
```
hostA> netcat -l 4444 > /tmp/file1
hostB> echo "cats suck dogs rule" > myfile
hostB> nc hostA 4444 < myfile
hostA> cat /tmp/file1
cats suck dogs rule
```

NPING (part of nmap pkg)

send TCP packets over port 22, 80 and 443
```
nping --tcp nycweb01 -p 80,443,22
```

send UDP packet

To spin up webserver on specific network interface,
```
python -c 'import BaseHTTPServer as bhs, SimpleHTTPServer as shs; bhs.HTTPServer(("192.168.200.99", 8331), shs.SimpleHTTPRequestHandler).serve_forever()'
```

check ports using nmap
```
nmap localhost
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
89/tcp   open  su-mit-tg
```

NMAP
check subnet for open ports
```
nmap -sP -PS22,3389 192.168.30.1/24
```

DNS

Check DNS routing
```
host github.com
github.com has address 192.30.253.113
```

```
ip route del 40.2.2.0/24 via 30.1.2.2
ip route add 40.2.2.0/24 via 30.1.2.2 metric 1234
```

kill all connections on port 21
```
tcpkill -i eth0 port 21
```

add TCP permissions to TCP analyzer tools so non-root users can create sockets and access network interfaces
```
setcap cap_net_raw,cap_net_admin=eip /usr/bin/tcpreplay
setcap cap_net_raw,cap_net_admin=eip /usr/bin/tcpdump
```

check bandwidth usage
```
yum install iperf
iperf -c <ip of target> -p 22

Client connecting to 208.224.251.3, TCP port 22
TCP window size: 90.0 KByte (default)
------------------------------------------------------------
[  3] local 172.31.23.96 port 48908 connected with
208.224.251.3 port 22
write failed: Connection reset by peer
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.0 sec   130 KBytes  68.9 Mbits/sec
```

```
  github.com has address 192.30.253.112
  github.com mail is handled by 10 ALT3.ASPMX.L.GOOGLE.com.
```

Dig into DNS query
```
dig www.domain.com
```

check all DNS name servers
```
cat /etc/resolv.conf
```

get your public IP from google
```
dig +short myip.opendns.com @resolver1.opendns.com
124.245.66.135
```

or this
```
curl -4 icanhazip.com
```

Check all open network connections
```
lsof -i
```

Get true Timezone
```
curl https://ipapi.co/timezone
```

Multicast

see what MC groups are present
```
ip maddr
```

Network Utilities

hping3 - like ping but can connect to ports and use TCP
iftop - iface network activity top
ss - better version of netstat (ss -ap4)
iptraf - interface and network cmd line gui tool (very good)

IPTABLES

show all rules
```
iptables -L -n -v
```

show all FORWARD rules
```
iptables -L FORWARD --line-numbers
```

delete a rule
```
iptables -D FORWARD <line number>
```

check existing NAT rules
```
iptables -t nat -v -L  POSTROUTING --line-number
iptables -t nat -v -L  PREROUTING --line-number
```

forward any request from ServerA port 80 to ServerB port 80
on server A:
```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j
DNAT --to-destination <IP of serverB>:80
iptables -t nat -A POSTROUTING -p tcp -j
MASQUERADE
```

change outgoing packets IP header
```
iptables -t nat -A POSTROUTING -d <destination IP>
--dport <PORT> -j SNAT --to-source <IP you want to
change to>
```

forward an OUTGOING packet for a specific port (going from host A), to another host (host B)
host A>
```
iptables -t nat -A OUTPUT -p tcp --dport 8331 -j
DNAT --to-destination 10.182.26.8:8331
```

allow all connections from an IP
```
iptables -A INPUT -s 59.50.131.179 -j ACCEPT
```

forward a packet going to a specific hostname and port to another hostname:port

save all IPTABLES rules permanently
```
iptables-save > /etc/sysconfig/iptables
```

restore from file
```
iptables-restore < /tmp/backup.iptables
```

add Debug log to prerouting rule #3 (tail syslog)
```
iptables -t nat -I PREROUTING 3 -j LOG
```

allow SSH port 22 only from address 190.120.30.3, block all others
```
iptables -I INPUT -p tcp '!' -s 190.120.30.3 --dport 22 -j REJECT
```

allow SSH port for specific address
```
iptables -A INPUT -p tcp -s 190.120.30.3 --dport 22 -j ACCEPT
```

block port
```
iptables -A OUTPUT -p tcp --dport 2500 -j DROP
```

allow a port
```
iptables -A INPUT -p tcp --dport 2500 -j ACCEPT
```

allow an IP address
```
iptables -A INPUT -p tcp -s 192.168.3.5 -j ACCEPT
iptables -A OUTPUT -p tcp -d  192.168.3.5 -j ACCEPT
```

block an IP address
```
iptables -A INPUT -s 192.130.2.4 -j DROP
```

block range of IPs
```
iptables -A INPUT -s 192.168.2.0/24 -j DROP
```

allow range of ports (1200 and 5000-6000)
```
iptables -A INPUT -p tcp --match multiport --dports 1200,5000:6000 -m
conntrack -j ACCEPT
```

redirect port to another port on same host

```
iptables -t nat -A PREROUTING -p tcp -d 18.224.251.4 --dport 22 -j DNAT --to-destination 192.168.10.22:22
```

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 25 -j REDIRECT --to-port 2525
```

create a custom CHAIN

```
iptables -N My-Custom-Rules
```

Troubleshooting

Packages / Libs

show installed software

Debian distro

```
dpkg -l
apt-cache search [pkg name]
```

Fedora distro

```
yum list installed
rpm -qa | grep [pkg name]
yum search [pkg name]
```

RPM install package

```
rpm -i pkg.rpm
rpm -i mypkg.rpm --force  (force install)
rpm -i mypkg.rpm --nodeps (ignore dependencies)
```

what RPM does a file belong to?

```
rpm -qf /usr/bin/mysqlaccess
```

show files inside installed RPM package

```
rpm -ql package-name
```

show files inside local uninstalled RPM package

```
rpm -qpl local-file.rpm
```

Show libraries for a program

```
ldd /bin/ls
```

refresh YUM cache

```
yum clean expire-cache
yum clean all
```

show dependency for a package

```
yum -q deplist $pkg
```

see install/upgrade history

```
yum history
```

get info on specific yum transaction

```
yum history info <# of transaction>
```

rollback yum patch

```
yum history undo <# of transaction>
```

rebuild cached library list or add new libs

```
vi /etc/ld.so.conf
ldconfig
```

Install package including its dependencies, example 'mysql'

```
yum deplist mysql | awk '/provider:/ {print $2}' | sort -u | xargs yum -y install
```

show installed packages by disk space usage (Centos)

```
rpm -qa --queryformat '%10{size} - %-25{name} \t %{version}\n' | sort -n
```

Process / Init / CPU

get uptime of a process

```
ps -p $$ -o etime=
```

where $$ is PID, result is in format dd-hh:mm:ss

find PID of a process (add to .bashrc)

```
function pid() { ps -fU $USER | grep $1 | grep -v "grep" | grep -v "ps -fU"  ;}
```

Run process in background (use & to push to background)

```
./run_script.sh &
```

Get current PID

```
$$
```

Run command with a process "niceness" or priority (-20 highest priority, 19 lowest)

```
nice -18 cat /etc/hosts
```

Check new incoming connections on port, live

```
ss -nap | grep 4433
```

Change a running program's priority (change to priority 7, PID 168390 for all processes running by users 'root' and 'joe'

```
renice 10 168390 -u root joe
```

Systemctl

kill all processes by name (with confirmation)
```
pkill -f $name
```

kill process by Port
```
fuser -k 5100/tcp
```

kill process by owner name
```
killall -u username
```

find process by name, kill all
```
ps -ef | grep "vault server" | grep -v grep | awk '{print $2}'
| xargs kill -9
```

show 4 way scrollable process tree
```
ps awwfux | less -S
```

show all processes and children
```
ps -ef --forest
```

show # of processes per user
```
ps hax -o user | sort | uniq -c | sort -r
```

kill a process running on Port 8331
```
kill -9 $(lsof -i :8331 | awk '{l=$2} END {print l}')
```

NTP

check offset of time between 2 servers,
```
[23:38 root@web1:~ ]# ntpdate -q web2
server 10.112.42.8, stratum 2, offset 0.005212, delay 0.02580
13 Aug 23:39:01 ntpdate[17325]: adjust time server 10.182.48.8
offset 0.005212 sec
```

offset of less than 5/1000s of a second

check offset against timeserver
```
ntpq -p
```

run in debug
```
ntpdate -dv <name of timeserver>
```

Hide processes and PIDs for non-root users

edit /etc/fstab
```
proc   /proc  proc    defaults,hidepid=2 0 0
```

remount
```
mount -o remount,rw,hidepid=2 /proc
```

to add an exception for a group/user (let this group see other PIDs), add 'gid' & remount
```
proc   /proc   proc    defaults,hidepid=2,gid=joe  0 0
```

Isolate CPUs for specific processeses

grubby --default-kernel
/boot/vmlinuz-3.10.0-862.14.4.el7.x86_64

grubby --info=/boot/vmlinuz-3.10.0-862.14.4.el7.x86_64
args="ro no_timer_check console=tty0 console=ttyS0,115200n8
net.ifnames=0 biosdevname=0 elevator=noop crashkernel=auto
LANG=en_US.UTF-8"

get current islated cores
cat /sys/devices/system/cpu/isolated

add cpu isolation
grubby --update-kernel=/boot/vmlinuz-3.10.0-862.14.4.el7.x86_64 --
args=isolcpus=2,3

reboot host to pickup changes

remove isolation
grubby --remove-args="isolcpus=2,3" --update-kernel=<kernel
name>

show all enabled services
```
systemctl list-unit-files | grep enabled
```

show all running services
```
systemctl list-units --type=service --state=running
```

start / stop / status / refresh / reload / enable / disable / show
```
systemctl start httpd.service
```

analyze bad startup script
```
systemd-analyze verify monit.service
```

refresh sysctl
```
systemctl daemon-reload
```

I/O
monitor high disk IO
```
* * * * * root /usr/sbin/iotop -botqqqk --iter=60 | grep -P
"\d\d\.\d\d K/s" >> /var/log/iotop
```

Limit CPU usage for a process #2240 to 50% of CPU and also its child procs
```
cpulimit -pid 2240 -l 50 -i
```

Taskset and NUMACTL

start a process on only 1st CPU core
```
taskset -c 0 /bin/nginx
```

for multiple CPU affinity
```
nohup taskset -c 0,1,2,5 /bin/program
```

get range of CPUs on which process can run on (affinity)
```
taskset -cp <PID>
```

get CPU on which a PID is running on
```
ps -mo pid,tid,fname,user,psr -p <PID>
```

pin processes to specific CPUs that are isolated (by default, numa does not allow pin to isolated CPUs, must use ALL option)
```
cat /proc/cmdline
isolcpus=2,3
nohup numactl --all -C 2,3 /bin/myprogram
```

find User and Parent PID of a zombie process thats holding up a port

#1 get the iNODE
```
root@min1# netstat -ltpnae | awk 'NR==2 || /:18100/'
Proto Recv-Q Send-Q Local Address           Foreign
Address         State       User  Inode  PID/Program name
tcp        1      0 0.0.0.0:18100
0.0.0.0:*               LISTEN      1000  24444060
-
tcp        1      0 192.168.37.5:18100
208.224.250.11:1046     CLOSE_WAIT  0     0
-
```

#2 search by iNODE
```
root@min1# lsof | awk 'NR==1 || /24444060/'
COMMAND         PID    TID        USER   FD      TYPE
DEVICE   SIZE/OFF       NODE NAME
trading_engine 138517 138529      joe   50u     IPv4
24444060        0t0        TCP *:18100 (LISTEN)
```

run program in background, no output
```
nohup programName 2>&1 &
```

User / Group / Sudo

USERS

create new user
```
adduser eric
```

add user to Group
```
usermod -aG mygroup eric
```

remove user from Group
```
gpasswd -d <user> <group>
```

add user to multiple groups
```
usermod -aG group1,group2,group3 eric
```

change UID for user
```
usermod -u 2550 eric
```

change GID for user
```
groupmod -g 2550 eric
```

lock a user account
```
passwd -l eric
```

unlock user account
```
passwd -uf eric
```

delete a user's password
```
passwd --delete eric
```

change user's shell
```
usermod --shell /bin/bash eric
```

remove user from group
```
gpasswd -d joe wheel
```

create a nologin user (no home dir)
```
useradd -r joe
or
adduser -r -s /bin/nologin jsmith
```

create user Joe with custom home dir, custom ID 999, custome group ID 555, add to 2 groups (corp, webadmins)
```
useradd -d /var/home/joe -u 999 -g 555 -G corp,webadmin joe
```

via Perl
```
adduser --home /var/home/joe -u 999 -g 555 -G corp,web joe
```

GROUPS

add new group
```
groupadd mygroup
```

modify group ID
```
groupmod -g 999 mygroup
```

change group name
```
groupmod -n newgroup oldgroup
```

show what cores a process is running on
```
for i in $(pgrep <name of process>); do ps -mo
pid,tid,fname,user,psr -p $i;done
```

File / Dir

Rsync

sync files from one Dir1 to Dir2
```
rsync -azP dir1/ dir2    ## -z flag is compression
```

-azP flag is used to compress file (z), and P for partial, it will only rsync deltas instead of starting all over from scratch

RSYNC file to a remote system's /tmp dir
```
rsync -azP file1 root@remotesystem:/tmp
```

rsync and exclude logs, png
```
rsync -azP --exclude={*.log,*.png} server1:/tmp/dir /tmp
```

Pull file from a remote system to a local /tmp dir
```
rsync -azP root@remotesystem:/opt/file1 /tmp
```

```
If Rsync not found, use path
 --rsync-path=/usr/bin/rsync
```

```
Rsync using a hop server (A > B > C)
assuming you can ssh joe@A > joe@B
and can ssh from joe@B > joe@C
```

Logrotate

place all logrotate confs in /etc/logrotate.d

```
/var/log/httpd/*log {
    rotate 3  # how many
rotated files to keep left
over
    size 10MB  # rotate if
log exceeds this
    daily  # rotate on daily
basis unless size max
criteria is met first
    maxage 20 # delete old
rotate files over 20 days
    compress  # gzip
compress rotated files
    missingok
    notifempty
    sharedscripts
    postrotate
        /sbin/service httpd
graceful 1>/dev/null 2>&1 ||
true
    endscript
}
```

make sure to

```
rsync -azP -e "ssh -A joe@B ssh" file1 joe@C:/tmp

will rsync local file1 via B, into C


if Rsync versions dont match up, can also do this, (rsyncs file on C to localhost via B)
rsync -azP -e 'ssh -o "ProxyCommand ssh -A joe@B nc %h %p"' joe@C:/tmp/xferfile .


rsync - set mod and ownership on incoming files/dirs,

  hostA> ls -la /home/joe
  drwxrwsr-x. 3 joe   groupA  21 Sep 16  2018 tmp/

  hostB> rsync -azP --chmod 644 --chown=mary:accounting hostA:/home/joe/tmp .
  hostB> ls -la
  drw-r--r--  mary  accounting  /tmp



 Sort file

sort -d filename  ## alphabetically
sort -r filename  ## reverse order
sort -n filename  ## numeric sort
sort -M filename  ## sort by month date

SSHFS
sudo sshfs -o allow_other,defer_permissions root@xxx.xxx.xxx.xxx:/ /mnt/droplet


copy all files to destination except for whatever is in .gitignore
cp -r !($(cat .gitignore)) /tmp/dest

mount NFS share
yum install nfs-utils nfs-utils-lib
service nfs start

remove first 500 lines of a file, in place (shrink a log file)
sed -i -e 1,500d file.log

reduce log file to 200b
truncate -s 200 file.log

User & Group Permissions
give 'sysadmin' Group 777 permission to a dir /opt/test
chmod g+rwx /opt/test

change group ownership for symlink (recurse down)
chgrp -Rh mygroup /home/user/dir

add execute bit for group on all folders
find . type -d | xargs chmod g+x

change group ownership of a dir
chgrp sysadmins /opt/test

Get ACL on a directory
getfacl /opt/test

give Sysadmins group 777 to /opt/test
setfacl -m group:sysadmins:rwx /opt/test

to set recursively down,
setfacl -Rm u:joe:rwx /home/mary

remove ACL
setfacl -x user:antony /opt/test

give r/w access to /home/user1 and preserve SSH security
chmod 750 /home/user1
setfacl -m user:user2:rw /home/user1

remove all ACLs from file or dir
setfacl -b /home/user1
```

Searching

find all files larger than 100M
```
find /home -xdev -type f -size +100M | xargs du -sh | sort -hr
```

Find 10 largest files
```
find . -type f -print0 | xargs -0 du | sort -n | tail -10 | cut -f2 | xargs -I{} du -sh {}
```

another way
```
find /home -type f -exec du -Sh {} + | sort -rh | head -n 5
```

find all files created in last 120 minutes
```
find / -cmin 120
```

Find 10 largest dirs
```
find . -type d -print0 | xargs -0 du | sort -n | tail -10 | cut -f2 | xargs -I{} du -sh {}
```

find 25 largest files in current dir and its subdirs
```
find . -type f -exec ls -al {} \; | sort -nr -k5 | head -n 25
```

find duplicate files, (based on MD5 hash)
```
find -type f -exec md5sum '{}' ';' | sort | uniq --all-repeated=separate -w 33
```

find specific user's files
```
find . -user <username> -print
```

recursively remove all empty subdirs
```
find . -depth -type d -empty -exec rmdir {} \;
```

find all hard links to a file
```
find /path/to/dir -xdev -samefile <name of file>
```

find the latest modified files (recursively)
```
find . -type f -exec stat --format '%Y :%y %n' "{}" \; | sort -nr | cut -d: -f2- | head
```

find files modified or created in last 2 days
```
find /dir -newermt "2 days ago" -ls
```

Show top 10 largest open files
```
lsof / | awk '{ if($7 > 1048576) print $7/1048576 "MB" " " $9 " " $1 }' | sort -n -u | tail
```

show 10 largest files in a directory
```
du -a /opt/blah | sort -n -r | head -n 10
```

list by size(-S), human readable(-h), all(-a), reverse date order (-r), list (-l), date (-t)

set a default ACL for a directory (all new files or dirs created in this directory will inherit ACL permissions)
```
setfacl -d -m u::rwx,g::rwx,o::r- /opt/testdir
setfacl -Rdm u:joe:rwx /opt/somedir
```

ensure all files and dirs created by user, inherit the Group permission of parent directory (SUID bit) - this example gives Joe rwx, gives group "employees" only Read (directories get set with X in order for group members to 'ls' to them), all others have no access to this folder or subfolders

```
1. chgrp -Rh employees /home/joe
2. setfacl -d -Rm u::rwX,g::rX,o::-  /home/joe
3. chmod -R g+s /home/joe  (set S bit to inherit parent permissions for all new
subfolders)
4. chmod -R g-w /home/joe  (removes write perms for group inside joe's home folder)
```

find files older than 300 days, display them
```
find /tmp -type f -mtime +300 -
print | xargs ls -lha
```

now delete them
```
find /tmp -type f -mtime +300 -
print | xargs rm
```

Find and Search
```
find -name filename ## any file
```

Find in specific dir
```
find /tmp -name myfile
```

Find file in specific location larger than 20MB
```
find /tmp -size +20M
```

Find files larger than 20MB and older than 360 days, delete them
```
find /tmp -type f -size +20M -
mtime +300 -print | xargs rm
```

get last element
echo /my/dir/name/backups/someFile.tar | awk -F"/" {'print $(NF)'}
someFile.tar

or another way,
basename /my/dir/name/backups/someFile.tar // someFile.tar

compare contents of 2 directories
diff <(cd </path/to/dir1> && find | sort) <(cd </path/to/dir2> && find | sort)

Freeze (lock) a directory or file from being modified (ACL, permissions, ownership,etc) - only root can unlock this. NOTE - this also prevents creating new files, this "freezes" the dir completely.
chattr +i <dir name> (locks dir)
chattr -i <dir name> (unlocks)

General Commands

mount ISO
```
mount -t iso9660 -o loop /home/tecmint/Fedora-18-i386-DVD.iso /mnt/iso/
```

unmount ISO
```
umount /mnt/iso
```

Compression

compress using bz2
```
tar cvfj  mydir.tar.bz2  /home/mydir
```

untar tar.bz2 file
```
tar -xvf file.tar.bz2
```

compress a file XZ format (best compression)
```
tar -cvpJf mydir.tar.xz /home/user/mydir
```

check JSON formats for multiple files
install jsonlint and check format
```
npm install jsonlint -g
$ for i in $(ls | grep *.json); do jsonlint $i;
done
```

Cron

show all crons for a user
```
crontab -l -u <username>
```

edit crons for your user
```
crontab -e
```

execute cron manually
```
run-parts /var/spool/cron
```

Untar a XZ file
```
tar xf filename.tar.xz
```

tar a file or dir into tar.gz
```
tar zcvf name.tar.gz file1 dir1 dir2
```

untar and unzip
```
tar -xvzf file.tar.gz
```

untar .tgz
```
tar xzvf file.tgz
```

see whats inside a tar
```
tar -tvf mydir.tar
```

untar single file from tar.gz (for bz2, replace tar.gz with tar.bz2)
```
tar --extract --file=mydir.tar.gz file1
```

untar multiple files using wildcard
```
tar -zxvf mydirs.tar.gz --wildcards '*.php'
```

lrzip

create a symlink
ln -s <path to binary> <target name>
ln -s /opt/myProgram /usr/local/bin

download entire website down to local level (and convert links to local) Wget Mirror
```
wget -mk www.google.com
```

download a file using curl
```
curl -O -u<USERNAME>:<API_KEY> -X GET
https://api.bintray.com/packages/orgname/repo_name/pkg_name/logs/downloads-03-
12-2016.csv.gz
```

Audit

show status of audit system
```
auditctl -s
```

show all audit rules
```
auditctl -l
```

clear all rules
```
auditctl -D
```

monitor file for any changes
```
auditctl -w /etc/filename -p wa -k myfile_changes
```

see any changes done to file
```
ausearch -k myfile_changes
```

save audit rules permanently
```
add to /etc/audit/rules.d/audit.rules
-w /etc/filename -p wa -k myfile_changs
```

check user actions by user name, from yesterday to now
```
ausearch -ua joe -ts yesterday -te now -i
```

search by type of event
```
ausearch -ua joe -m SYSCALL (or EXECVE)
```

search by time range
```
ausearch -ua joe --start 09/09/2019 '12:04:00' --end 09/12/2019
'12:22:00'
```

search by parsing a specific log file
```
ausearch -ua joe --input /tmp/audit.log
```

get list of failed login attempts by user and IP where theyre coming from
```
last -f /var/log/btmp
```

Test Email
```
yum install mailx
mail -s "test email" user@company.com <
/dev/null
```

Centos Xauthority (graphical gui)
```
yum install -y xorg-x11-server-Xorg xorg-x11-
xauth xorg-x11-apps
grep -i X11Forwarding /etc/ssh/sshd_config
(should be set to Yes)
```

ssh to box
```
ssh -X name@box
xclock (test)
```

SYSCTL

show all current values
```
sysctl -a
```

write new value
```
sysctl -w vm.swappiness=2
```

load values from file
```
sysctl -p /etc/sysctl.conf
```

monitor a command (run command repeatedly)
```
watch -n 5 free -h (runs free -h every 5 sec)
```

Encrypt a File
```
openssl aes-256-cbc -salt -in
myFileUnencrypted.txt -out
myFileEncrypted.txt.enc -k myPASSWORD
```

Unencrypt File
```
openssl enc -aes-256-cbc -in
MyFileEncrypted.txt.enc -out
myFileUnencrypted.txt
<type in password>
```

```
}
```

Disk / Partitioning / Filesystem

list all partitions

```
fdisk -l
Disk /dev/sda: 11.3 GB, 11286446080 bytes, 22043840 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000591a7

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     1026047      512000   83  Linux
/dev/sda2         1026048    22042623    10508288   8e  Linux LVM
```

Get ID and type of disk

```
blkid
```

Additional Disk checks

http://www.foxhop.net/local-or-san-device-in-linux

check disk for errors

`fsck` (only works for certain filesystems)

check what kind of filesystem type

```
df -T
```

show volume groups

```
vgdisplay
```

extend volume group

```
lvextend -r -L+25GB /dev/lvol/name
```

Mount a NetApp device as a local filesystem

```
mount -t nfs -o _netdev,rw,hard,intr,nosuid,dev,bg,nfsvers=3 netappNas01:/netbackup /netbackup
```

add to /etc/fstab,

```
netappNas01:/netbackup    /netbackup    nfs    _netdev,rw,hard,intr,nosuid,dev,bg,nfsvers=3 0 0
```

check if Disks are local or mounted SAN

```
ls /dev/disk/by-path/
```

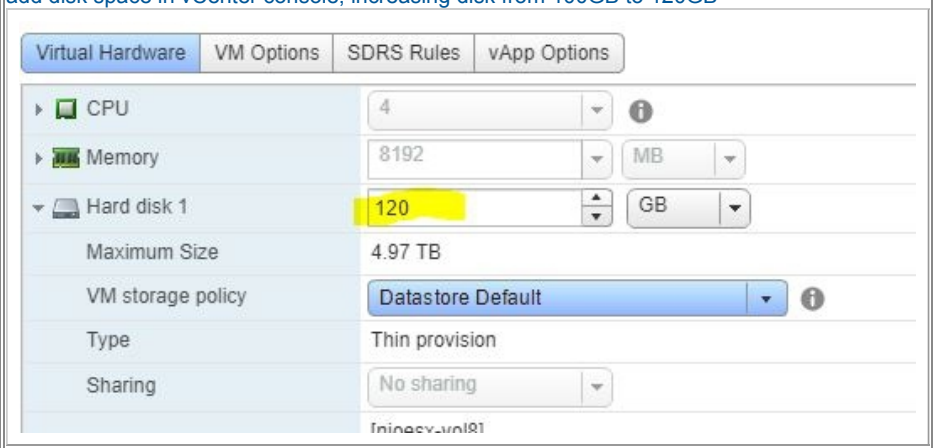
(SANs will have an IP next to path)

Increase partition space via vCenter GUI

Problem: current /opt only has 75G of available space, need to add another 20G

```
df -h
/dev/mapper/vg0-opt    80G  1.9G   75G   3% /opt
```

add disk space in vCenter console, increasing disk from 100GB to 120GB



on Centos box check name of scsi device,

```
ls /sys/class/scsi_device/
0:0:0:0
```

Resize a logical partition

Expand partition

1. add space to Hard Disk on VM in vCenter or VirtualBox
2. check all partitions, need to resize /opt its 30% full,

```
[root@mrxsplunkidx02 joe]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/vg0-root  7.8G  1.1G  6.3G  15% /
devtmpfs              1.9G     0  1.9G   0% /dev
tmpfs                 1.9G     0  1.9G   0% /dev/shm
tmpfs                 1.9G  8.6M  1.9G   1% /run
tmpfs                 1.9G     0  1.9G   0% /sys/fs/cgroup
/dev/sda1             976M  110M  799M  13% /boot
/dev/mapper/vg0-home  2.0G  7.0M  1.8G   1% /home
/dev/mapper/vg0-opt   4.8G  1.4G  3.3G  30% /opt
```

3. check logical space

```
lvs
appl vg0 -wi-ao---- 10.00g
home vg0 -wi-ao----  2.00g
opt  vg0 -wi-ao----  5.00g
root vg0 -wi-ao----  8.00g
swap vg0 -wi-ao----  2.00g
```

check available HD space,

```
vgdisplay
Free  PE / Size       191 / 5.97 GiB
```

4. Need to add another 5 Gigs to /opt

```
lvextend -r -L +5G /dev/mapper/vg0-opt
```

to extend ALL remaining free space,

```
lvextend -l +100%FREE /dev/mapper/vg0-opt
```

5. check the File System type of /opt

```
mount | grep opt
/dev/mapper/vg0-opt on /opt type ext4 (rw,relatime,data=ordered)
```

6. extend physical space

```
resize2fs /dev/mapper/vg0-opt
```

7. check space again, its now 15% full

```
df -h
/dev/mapper/vg0-opt   9.8G  1.4G  8.0G  15% /opt
```

check logical volume again,

```
lvs
opt  vg0 -wi-ao---- 10.00g
```

Shrink Partition

need to shrink partition /appl from 2GB to 1GB

rescan scsci bus
```
echo 1 > /sys/class/scsi_device/0\:0\:0\:0/device/rescan
```

check to see if extra space is visible,
```
fdisk -l
Disk /dev/sda: 128.8 GB
```

fdisk /dev/sda
type 'p' - prints out all partitions
type 'n' - create new partition
type 'p' - to make new partition

```
Command (m for help): p

Disk /dev/sda: 128.8 GB, 128849018880 bytes, 251658240 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x0001125e

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     2099199     1048576   83  Linux
/dev/sda2         2099200     2508799      204800    6  FAT16
/dev/sda3         2508800   209715199   103603200   8e  Linux LVM

Command (m for help): n
Partition type:
   p   primary (3 primary, 0 extended, 1 free)
   e   extended
Select (default e): p
Selected partition 4
First sector (209715200-251658239, default 209715200):
```

select the next available sector (default), select default Last Sector

```
First sector (209715200-251658239, default 209715200):
Using default value 209715200
Last sector, +sectors or +size{K,M,G} (209715200-251658239, default 251658239):
Using default value 251658239
Partition 4 of type Linux and of size 20 GiB is set
```

type 'w' to save changes
reboot the VM
once rebooted, type 'fdisk -l', a new partition is added

```
/dev/sda1   *        2048     2099199     1048576   83  Linux
/dev/sda2         2099200     2508799      204800    6  FAT16
/dev/sda3         2508800   209715199   103603200   8e  Linux LVM
/dev/sda4       209715200   251658239    20971520   83  Linux
```

now extend your /dev/mapper/vg0-opt

```
> vgs
VG  #PV #LV #SN Attr   VSize  VFree
  vg0   1   6   0 wz--n- 98.78g    0

> vgextend vg0 /dev/sda4
Volume group "vg0" successfully extended
```

Check to see available PE space (shows 20G of available space)
```
> vgdisplay
Free  PE / Size       639 / <19.97 GiB
```

now resize to full available space, will show 94G of available space

```
> lvextend -l +100%FREE /dev/mapper/vg0-opt
> resize2fs /dev/mapper/vg0-opt
> df -h
/dev/mapper/vg0-opt   100G  1.9G   94G   2% /opt
```

```
lvs

appl vg0 -wi-ao---- 2.00g
docker vg0 -wi-ao---- 10.00g
home vg0 -wi-ao---- 60.00g
opt vg0 -wi-ao---- 5.00g
root vg0 -wi-ao---- 8.00g
swap vg0 -wi-ao---- 2.00g
tmp vg0 -wi-ao---- 3.00g
var vg0 -wi-ao---- 3.00g
```

unmount it
```
umount -v /appl
```

get filesystem name
```
df -h
/dev/mapper/vg0-appl   /appl
```

check for file system error
```
e2fsck -ff /dev/mapper/vg0-appl
```
(must pass all 5 stages)

reduce FS by 1GB
```
resize2fs /dev/mapper/vg0-appl 1G
```

reduce the logical volume
```
lvreduce -L -1G /dev/mapper/vg0-appl
```

mount /appl back on
```
mount /dev/mapper/vg0-appl /appl
```

check size of partition
```
lvdisplay /appl

--- Logical volume ---
LV Path                /dev/vg0/appl
LV Name appl
VG Name vg0
LV UUID Aim8Q2-gxp2-jnT0-OcS2-d3To-n5Nd-
IJmvxo
LV Write Access read/write
LV Creation host, time xxxx, 2018-02-23
11:52:48 -0500
LV Status available # open 1
LV Size 1.00 GiB Current LE 32 Segments 1
Allocation inherit
Read ahead sectors auto - currently set to
8192
Block device 253:6
```

Mount an EC2 volume as /home
attach volume to instance
on ec2:
> lsblk
should be listed as nvme1n1 or similar name

get ID of volume
> blkid (get UUID)

> vi /etc/fstab
UUID=<insert ID> /home xfs defaults 0 0

remount
> mount -a

Regex / Awk / Sed

find all lines starting with #
```
^#.*$
```

find blank line
```
^\s*$
```

remove all duplicate entries from a file
```
awk '!x[$0]++' filename
```

Replace string in a file (write directly to file -i)
```
sed -i -e "s/${prev_version}/${version}/g" bitbucket.service
```

Replace anything between 2 delimeters "!!" with word "super"
```
sed -e 's/!.*!/super/g' /etc/file
```

remove whitespace
```
sed -i "s/ //g" file
```

remove 2nd line from top, from file
```
sed -i '2,$d' file
```

```
strip double quotes
echo '"string"' | tr -d '"'
```

Troubleshooting

slow / frozen system

check if procs are in uninterrupted sleep state (waiting for IO and causing slowness)
`ps aux` (check STAT column, will show procs that are in uninterrupted sleep)

check paging faults
`sar -B 2 5` will generate paging report, check majflt column, major faults per second, if high #, means system is out of RAM

enable SysRQ to kill procs that are in 'uninterrupted sleep' state. SysRQ will respond even in frozen state (assuming command line is responsive)

1. enable sysrq
2. kill D state procs

<END>