# BashRC,Bash\_profile,etc/profile

.**bashrc** file is placed at user’s home location, to see this file go to ~/.bashrc or /home/<username>

.bashrc file is used for nologin shell, this file is belongs to specific user

If we added anything in .bashrc file it will be applicable for particular user and once we open new terminal .bachrc get executed

But if we added something in .bashrc file and we want to apply it on same terminal we need to execute command **source ~/.bashrc** or **. ~/.bashrc**

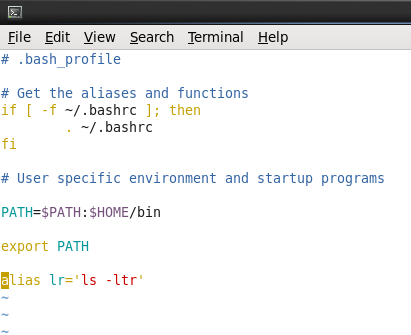
**.bash\_profile** also located at home (all file started with period. are hidden files)

. **bash\_profile** profile used for login shell, every users have his own bash\_profile file this stores user’s related environment variables and settings

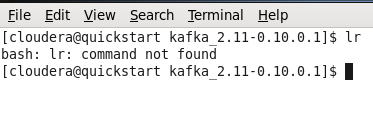
This file get executed when we do login into terminal

Lets take one example

Open a new terminal and add **alias lr='ls -ltr'** in **.bash\_profile** file



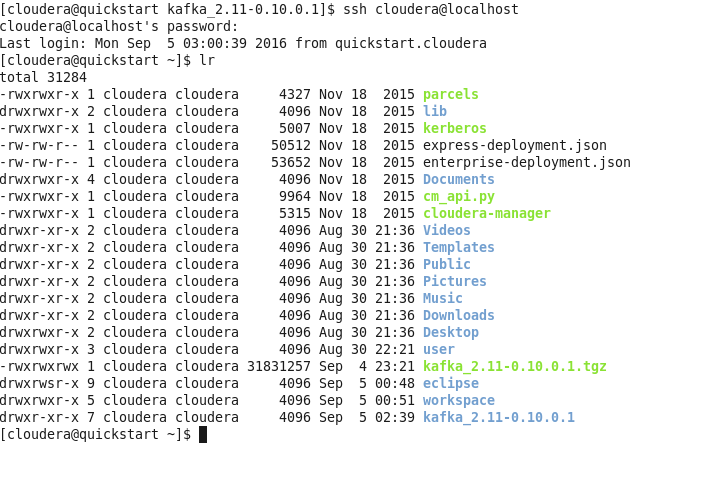
Now check command lr without login



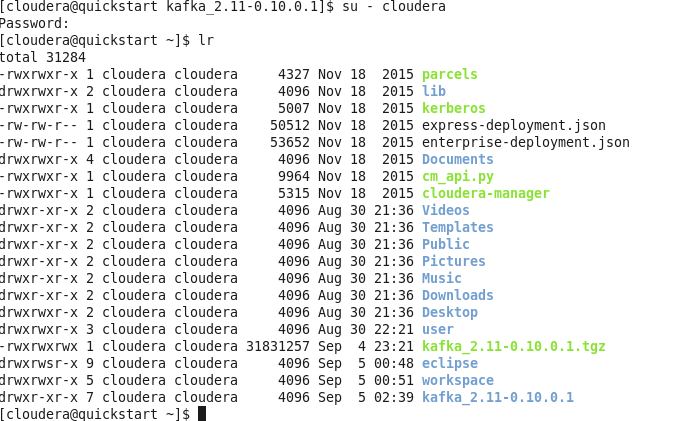
lr command not found because we haven’t logged in

now login first ,once you login bash\_profile file get executed automatically

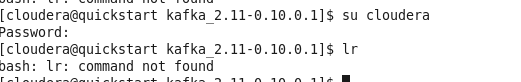
after you logged in it will recognize lr command, see below



Alternatively you can use **su –** command



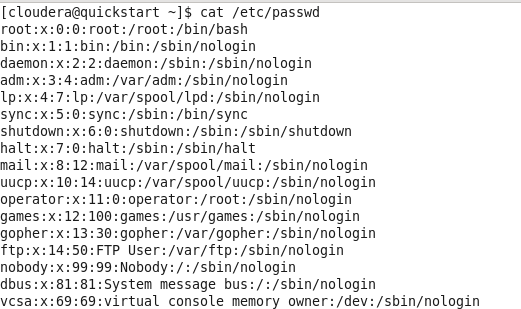
Here we need to remember that if we run only su command , .bash\_profile will not get executed



**/etc/profile** is system profile any changes in this file will reflect to everyone, once we login with any account this profile will get executed; this is like global environment variables file

echo $0 to know the shell type

**/etc/passwd** file holds all information regarding user



It is separated by colon

**cloudera:**x:501:501::/home/cloudera:/bin/bash

First word is user name

cloudera**:**x:501:501::/home/cloudera:/bin/bash

Second is password x means the password is stored in **/etc/shadow (please check /etc/gshadow also )** file in encrypted format

cloudera**:**x:501:501::/home/cloudera:/bin/bash

Third and fourth are user id and group id respectively ( you can check by executing id command), below 500 is for system accounts

cloudera**:**x:501:501::/home/cloudera:/bin/bash

next is group name in this case it is blank

cloudera**:**x:501:501::**/home/cloudera**:/bin/bash

next is home folder for this user

cloudera**:**x:501:501::/home/cloudera:**/bin/bash**

Last is the shell the user belongs to

you can modify this file to modify the information for particular group , e.g. you can change the group of the user by changing fourth parameter

but better to use **usermod** command to modify the user

you can see group information in /etc/group file

**useradd <username>** is for creating new user, it add the same group name as user name in /etc/group file so you can change it later or at the time of adding user you can specify the group , see **man useradd** command

|  |
| --- |
| useradd -g Shalaj –G admin –d /home/shalaj shalaj1  here we created shalaj1 user having primary group as Shalaj , secondary group as admin and home directory is /home/shalaj |

**passwd <username>** to change the password

**groupadd** command is for creating new group

**groupdel** is for deleting existing group

**groups** command is used for see the group name the current user belongs to

we can go to root by **su –** command , it asks for password

Note: in cloudera vm you need to execute **sudo su** – to move to root account as cloduera has sudo access for all

**/etc/skel** is directory having all the templates like bash\_profile, bashrc , whenever we creates new user new files created according to these template files

**chown** to change the group and owner of the file

|  |
| --- |
| chown <user > : <group> <file\_name/directory> |

**chgrp** to change the group of the file

|  |
| --- |
| chgrp -R <group> <file\_name>/<directory name>  -R for recursive change inside directory |

|  |
| --- |
| [root@quickstart ~]# touch /home/shalaj/1  [root@quickstart ~]# su - shalaj  [shalaj@quickstart ~]$ ls -ltr  total 0  -rw-r--r-- 1 root root 0 Sep 5 23:22 1 // user and group ownership is root and root  [shalaj@quickstart ~]$ vi 1 //cant update it is readonly for user shalaj  [shalaj@quickstart ~]$ exit  logout  [root@quickstart ~]# chown shalaj /home/shalaj/1 // change ownership to user shalaj  [root@quickstart ~]# su - shalaj  bash rc for user shalaj  bash profile for user shalaj  [shalaj@quickstart ~]$ ls -ltr  total 0  -rw-r--r-- 1 shalaj root 0 Sep 5 23:22 1  [shalaj@quickstart ~]$ vi 1 // now Shalaj user can update it as for user the permission is rw-  [shalaj@quickstart ~]$ |

# Switch User/Substitute User/Super user (SU) command

su command is used for switching the user without logout

|  |
| --- |
| su <username>  it will ask for password for specified user name  if we skip user name it will login as root  use exit key word to exit back from this account  If we use **su - <username>** command to switching the user it go to home directory of user and if we use **su <username>** command the directory will not get changed |

# Substitute User do

|  |
| --- |
| sudo command is used for executing some command without switching user  sudo –u <username> <command>  user should have sudo access in **/etc/sudoers** file  or sudo <command> to run a command as super user  **sudo visudo -f /etc/sudoers** is to check the sudo access (do not edit sudoers file with any editor)  [**visudo**](http://manpages.ubuntu.com/manpages/precise/en/man8/visudo.8.html) - Edits the sudoers file. This uses a special executable that checks the syntax of the file to ensure you are not locked out of sudo due to a corrupted sudoers file. If you attempt to save an invalid configuration it will stop you, and return you to the editor to fix it.  if you add some user in admin group it will access all command as root user  sudo useradd Shalaj admin    Cloudera has ALL access in ALL machine it can run ALL command with (ALL) accounts  We can use **sudo su** command combined , in this case it will not ask for root password , it will ask for your account password and you will switch to root user |

# GZ files

To compress in gz file

|  |
| --- |
| gzip <name of the file> |

To compress multiple files in single file

|  |
| --- |
| gzip -c file1 file2 > foo.gz |

To uncompress file

|  |
| --- |
| gunzip <name of the gz file> |

# Change javac path in Centos

First check javac version using below command



Now if we want to update it with different version

Use below command

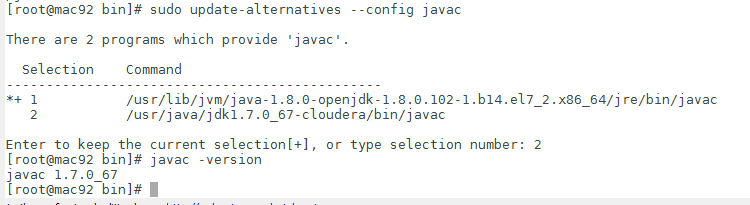
|  |
| --- |
| sudo update-alternatives --install "/usr/bin/javac" "javac" "/usr/java/jdk1.7.0\_67-cloudera/bin/javac" 1 |

Here we are updating soft link that are present under “/usr/bin/javac” and we are giving target path as “/usr/java/jdk1.7.0\_67-cloudera/bin/javac”

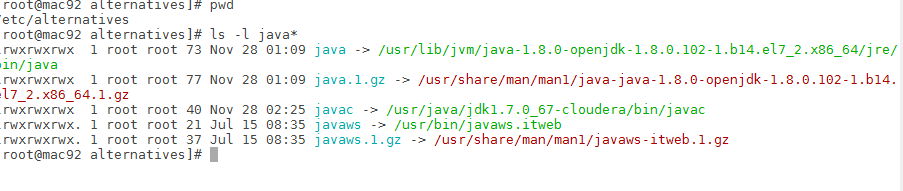
Now update alternative for javac using following command

|  |
| --- |
| sudo update-alternatives --config javac |

Now if you again check javac version it will be updated to java 1.7 , the same way you can update java version also



You can check the soft link by looking into /etc/alternatives folder



# SSH password less connection

We can set password less connection between two machines using ssh for that we need to generate private and public keys, there are certain command and utilities to generate private and public keys

On linux box command is

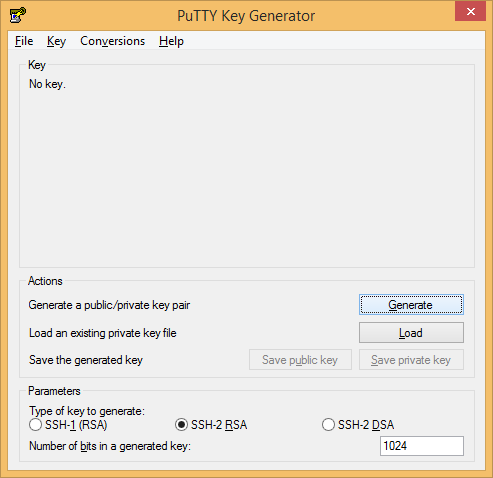
|  |
| --- |
| ssh-keygen -t rsa   * It cerate id\_rsa file under .ssh directory for private key * And id\_rsa.pub file for public key |

We need to save private key in our local machine and we need to add public key in remote machine so that the connection between local and remote machine would be password less

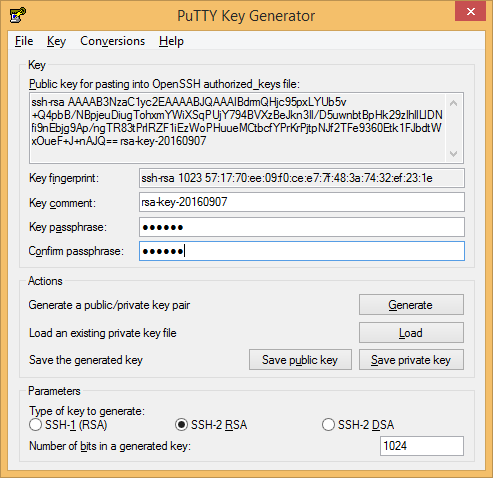
We need to create one directory namely **.ssh** in remote machine with permission 700, and under .ssh we need to create file **authorized\_keys** with permission 600, and paste the contents of public key in this file.

Now we can do ssh from local to remote machine without password, this passwordless communication is required when we need to run some automated script from one machine to another machine.

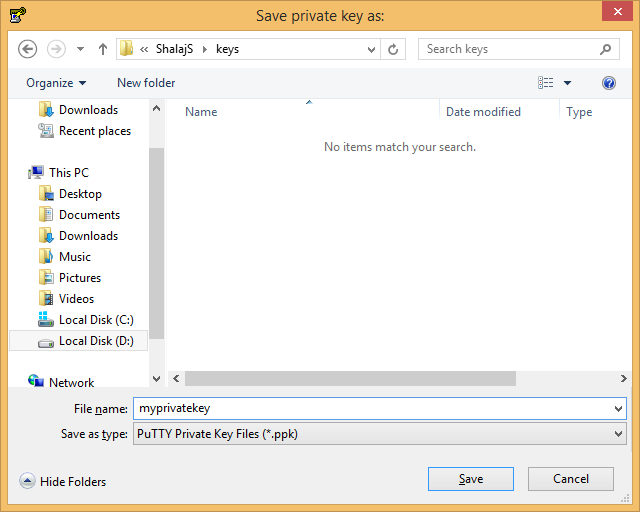
In windows we can generate private and public key using tools called puttygen



Click on Generate , it will create public and private keys



Set the key passphrase for your private key and Save Private key in some location



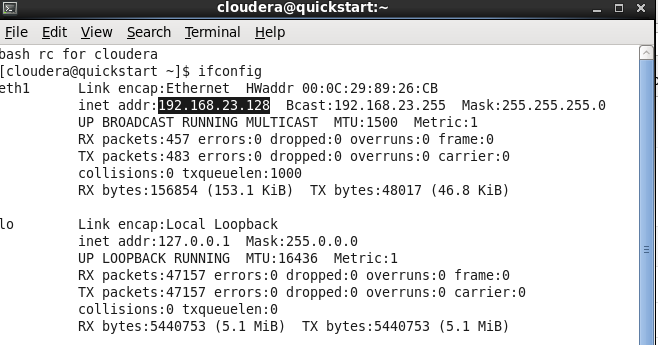
Copy the content of public key

|  |
| --- |
| ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAIBdrmQHjc95pxLYUb5v+Q4pbB/NBpjeuDiugTohxmYWiXSqPUjY794BVXzBeJkn3Il/D5uwnbtBpHk29zIhllLlDNfi9nEbjg9Ap/ngTR83tPrlRZF1iEzWoPHuueMCtbcfYPrKrPjtpNJf2TFe9360Etk1FJbdtWxOueF+J+nAJQ== rsa-key-20160907 |

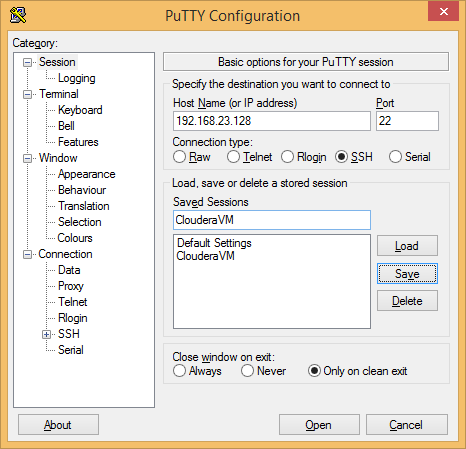
Now close the putty gen, now you have private key saved in file and public key in clipboard

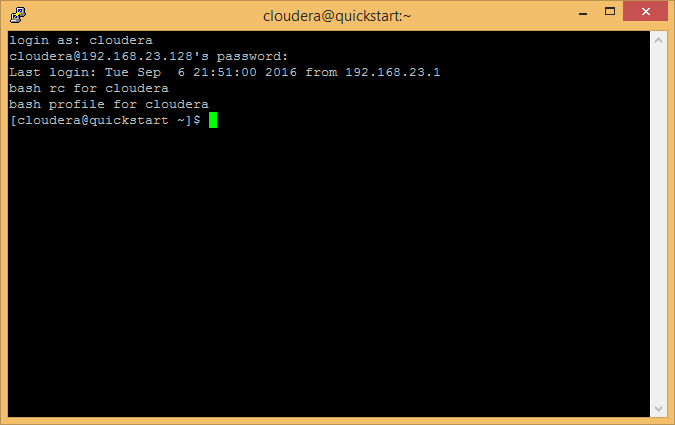
Now copy this public key in remote machine

IP of my cloudera vm is 192.168.23.128 , you can check it by using ifconfig command on linux



So we can login into this machine from your local window machine using putty or any other utility

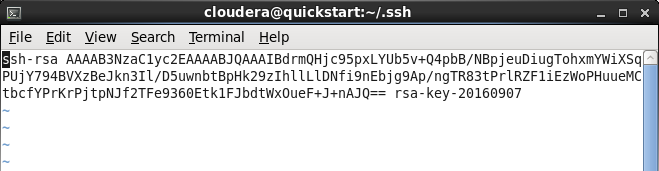


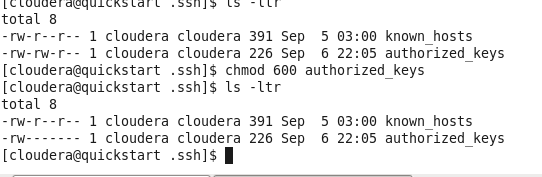


So here we have provided username and password for cloudera machine to connect with it.

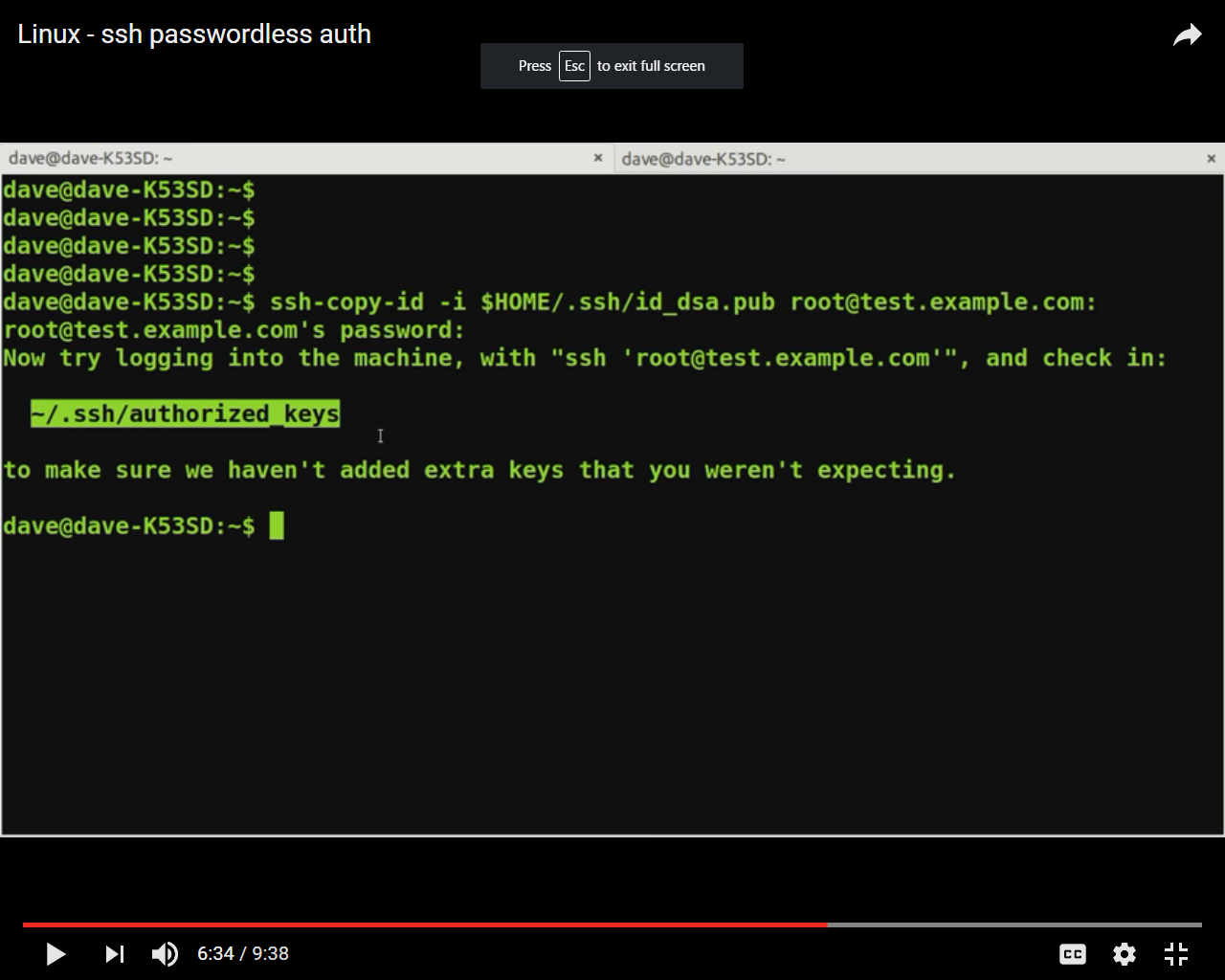
Now I need to login into it without giving password

First create directory .ssh and file authorized\_keys under this directory and paste the content of public key





There is a command to directly copy the public key content from one linux machine to other

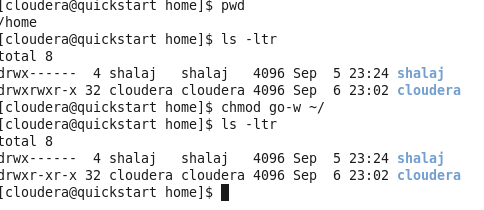


If you found below issue in /var/log/secure file at remote machine

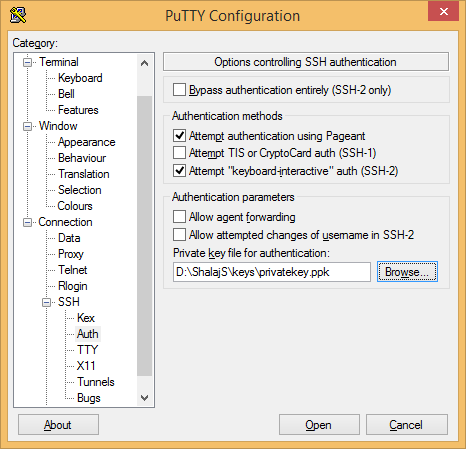
|  |
| --- |
| Sep 6 23:17:22 quickstart sshd[8506]: Authentication refused: bad ownership or modes for directory /home/cloudera |

It means your home folder permission is not proper to do the ssh

Run chmod go-w ~/ command to change your homefolder permission

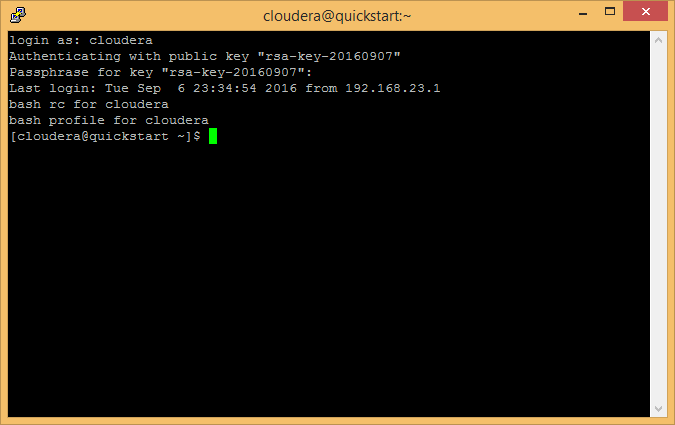


now open putty

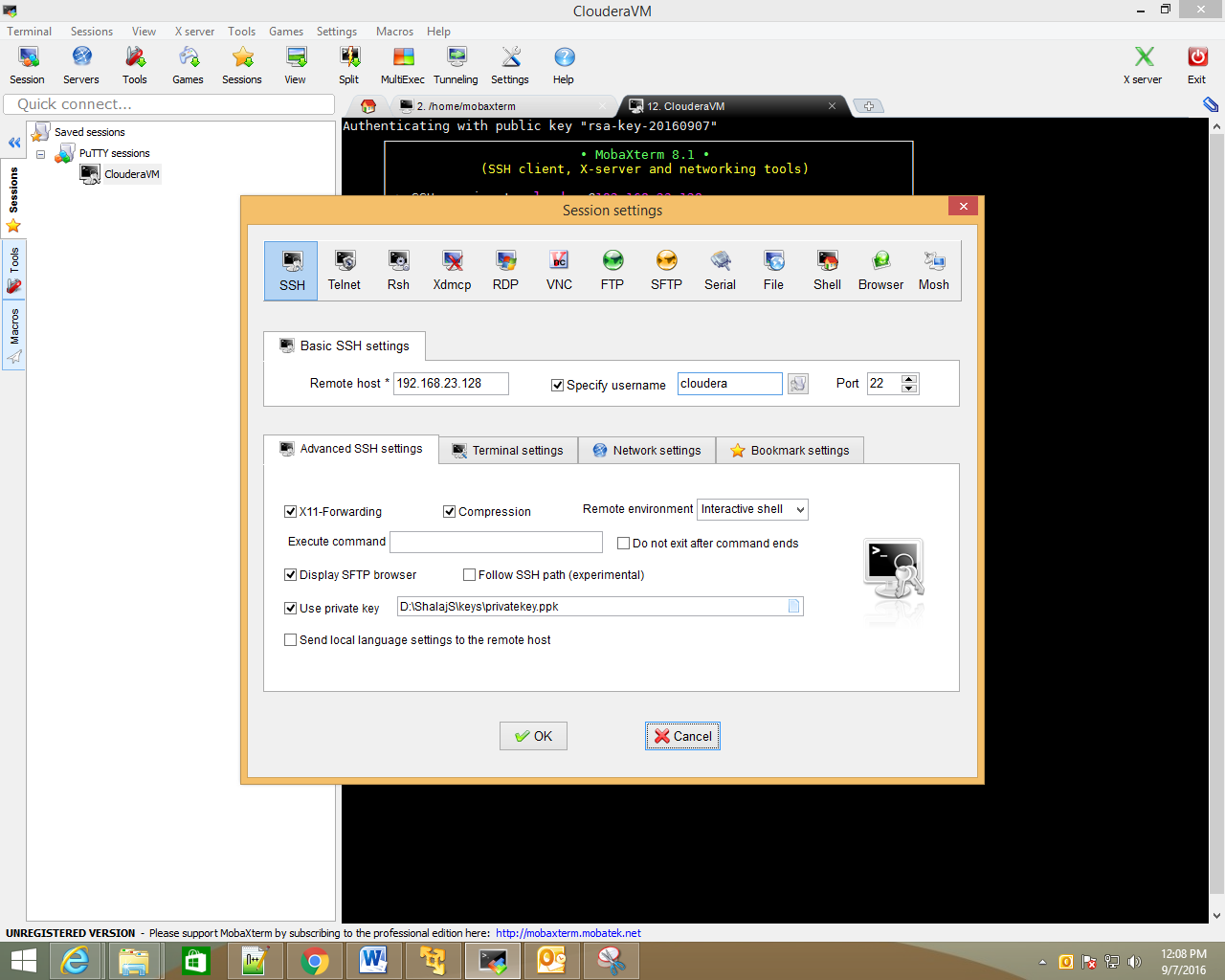


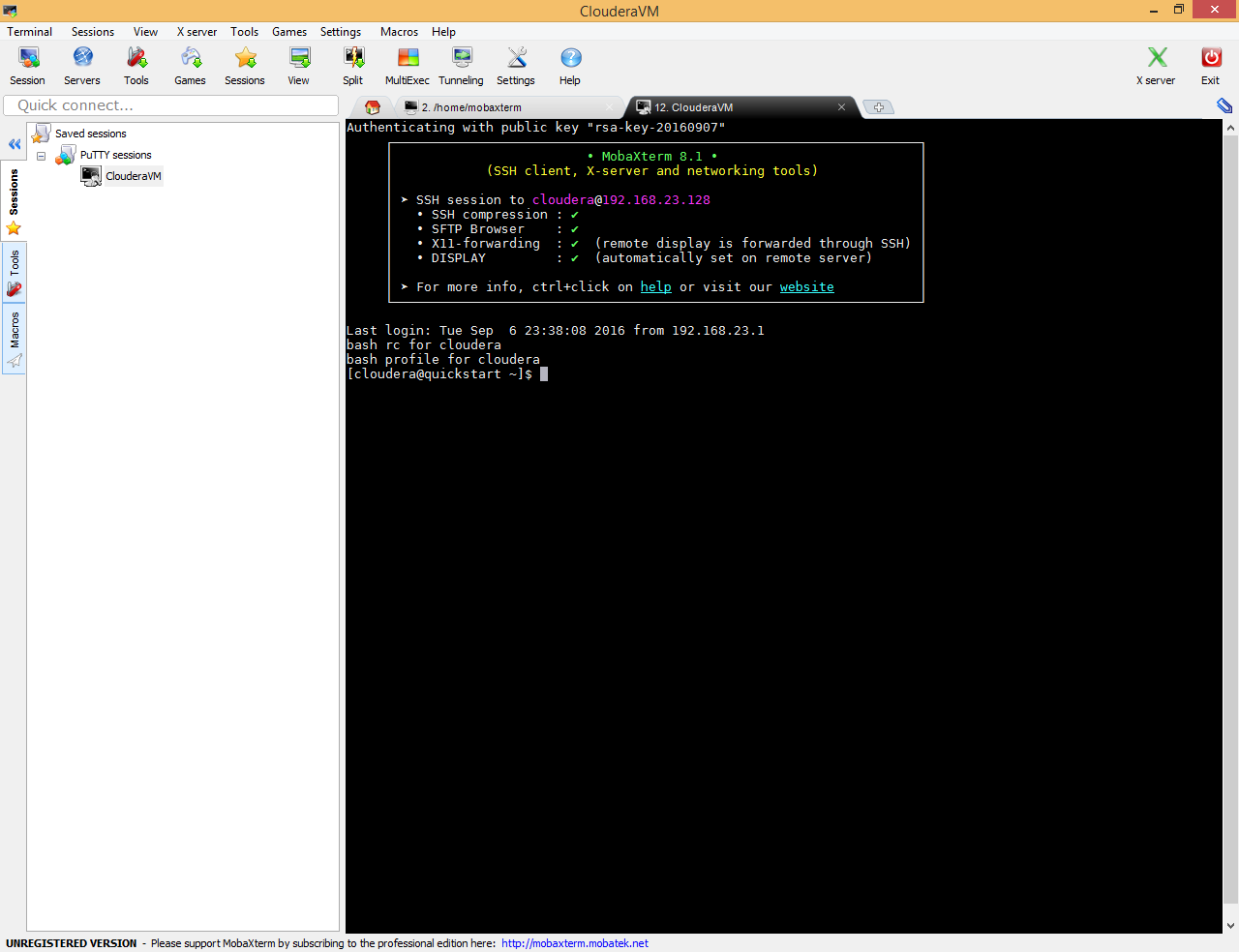
Select private key in SSh >>Auth

Now login to cloudera machine , now you can see instead of password it asks for paasphrase of private key



Same setting you can apply in **MobaXterm**



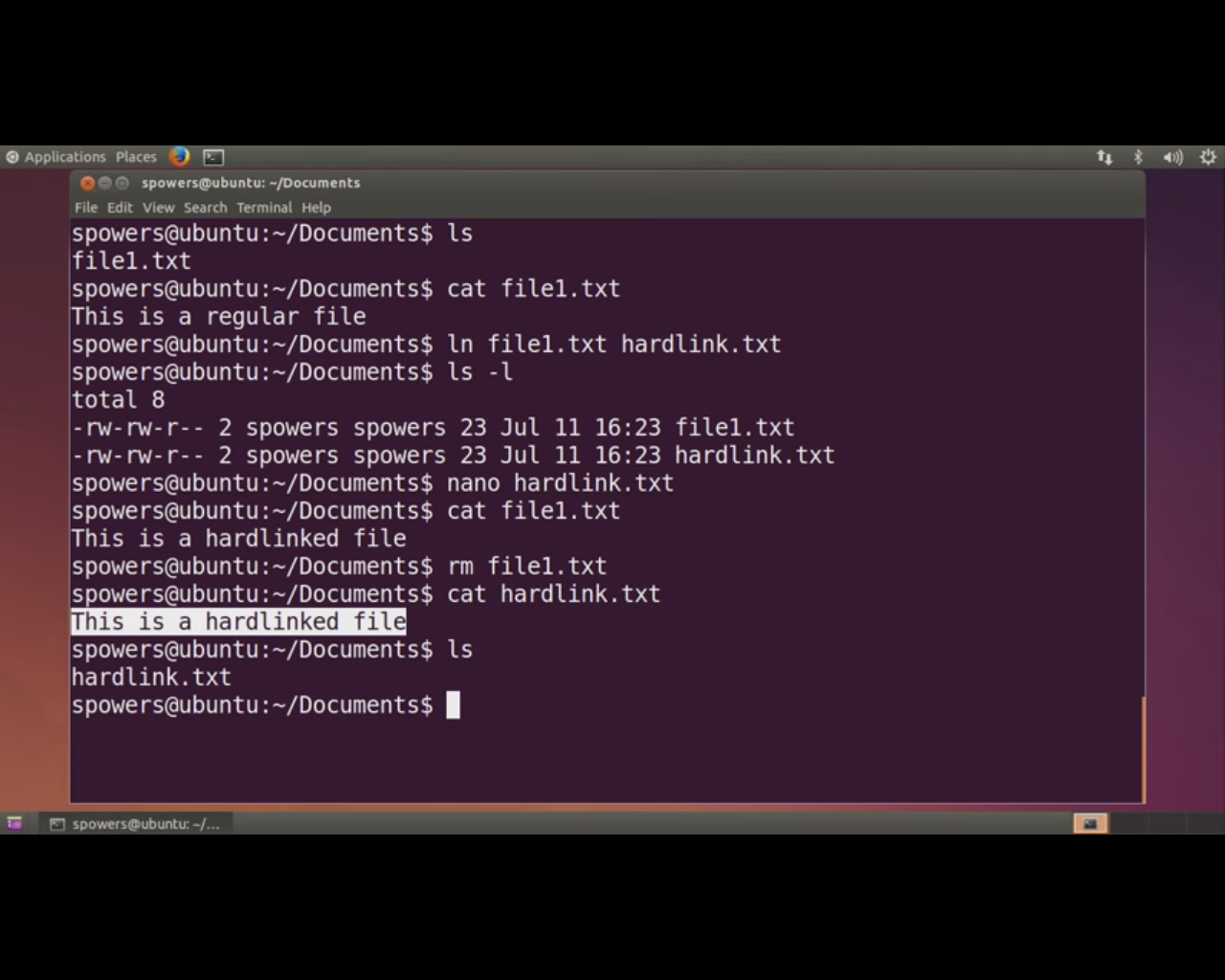


# Hard link and soft(Symbolic link) in linux

Hard link point to the same memory location in disk but it takes separate space to store the file and if we change the content of one file , the content of other file will get changed automatically but if we remove one file the other file persist

Command to create hardlink is

|  |
| --- |
| ln <original\_file> <hardlinkFile> |



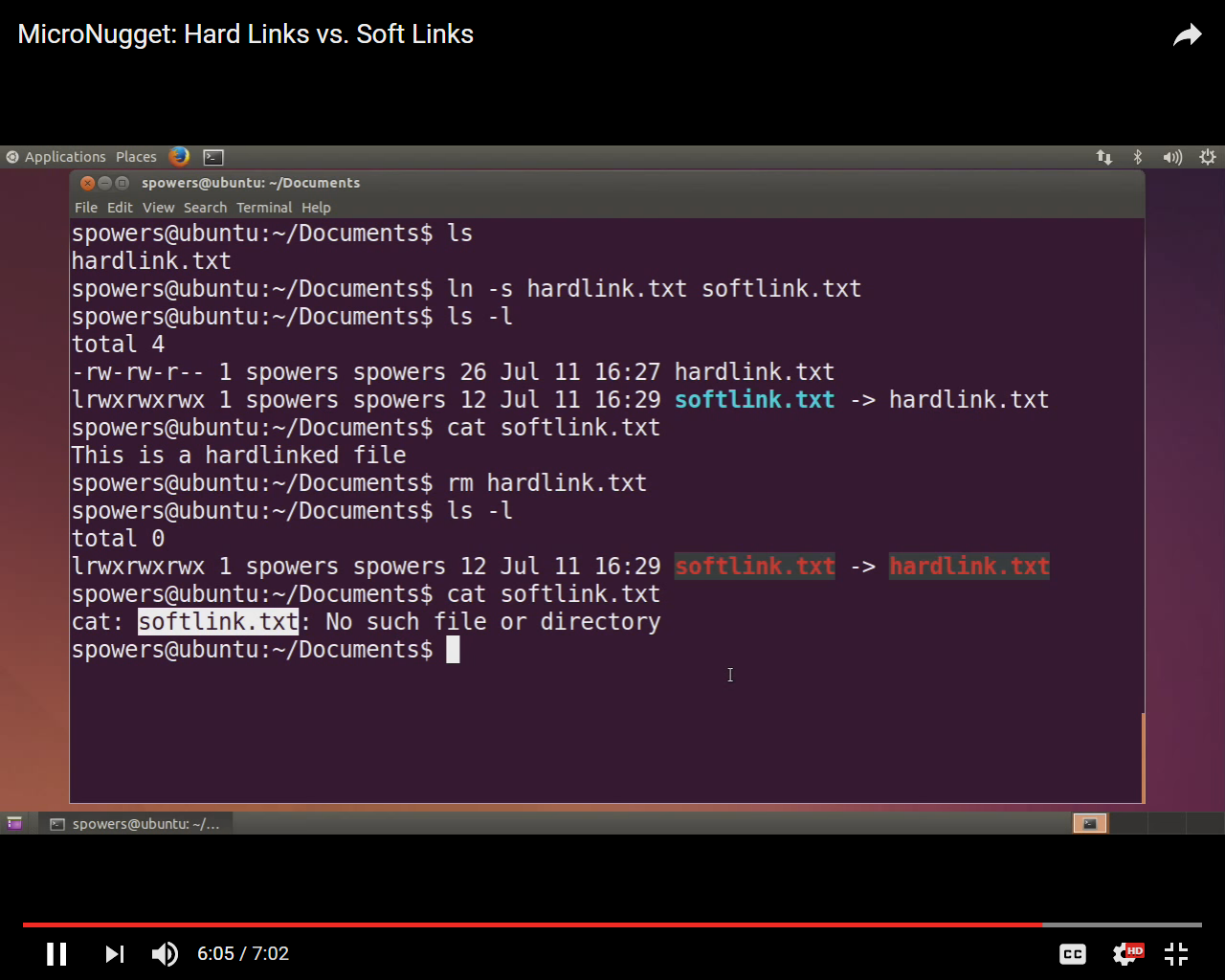
Soft link on the other hand point to the original file and it doesn’t take much space . its like shortcut that we creates on windows and if we delete original file , softlink file has no existence

Command to create softlink file is

|  |
| --- |
| ln –s <original\_file> <softlinkfile> |

Command to update symbolink link

|  |
| --- |
| ln -sfn {path/to/file-name} {link-name} |



Generally we use softlink in Hadoop eco system when we need to have the same file in more than one eco system ,so instead of copy paste the file we create soft link so that if later there is requirement to change in one file we don’t need to change it to other file

# Swappiness

check swappiness value

|  |
| --- |
| [root@mac122 log]# sysctl vm.swappiness  vm.swappiness = 60 |

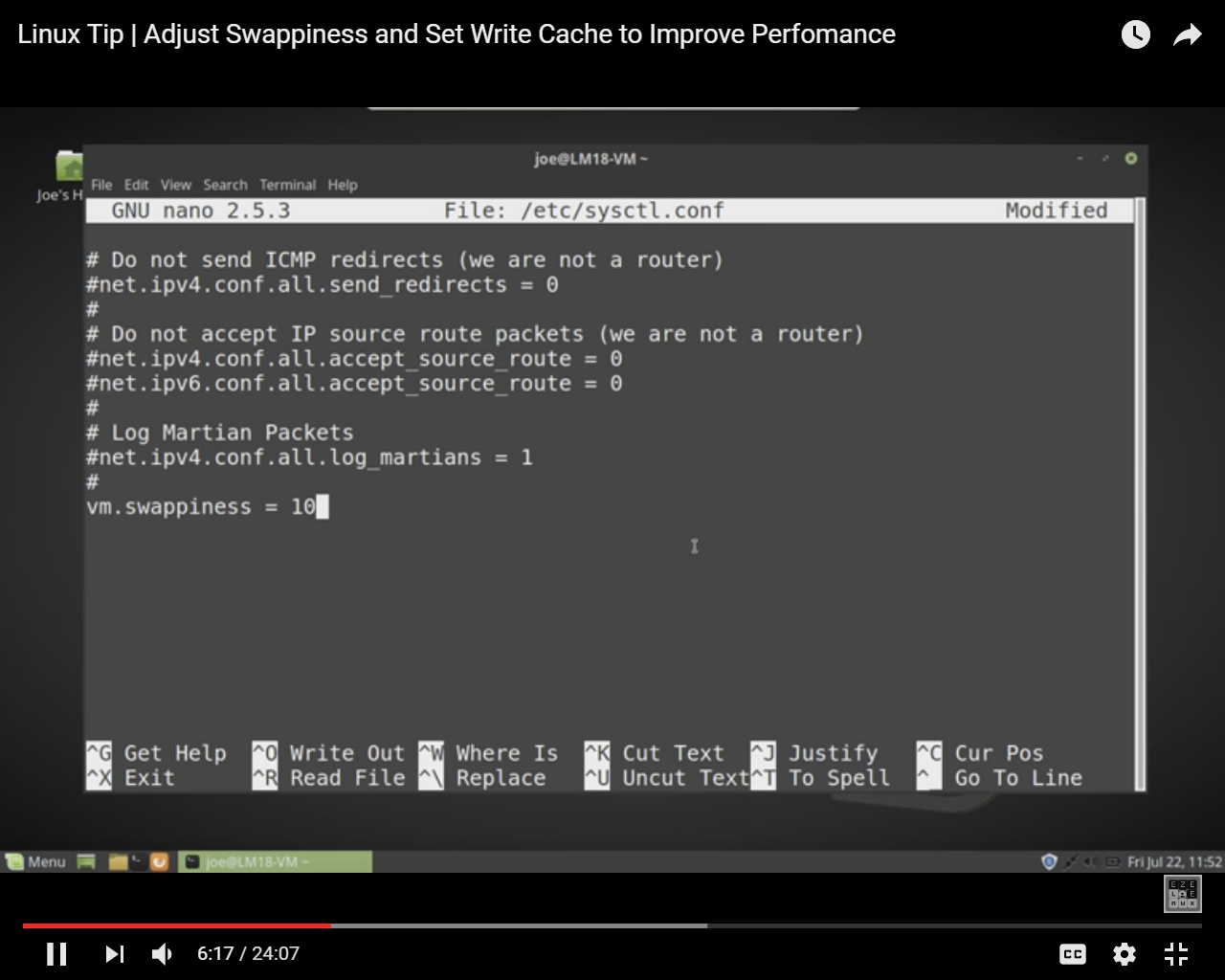
OR

|  |
| --- |
| [root@mac53 etc]# cat /proc/sys/vm/swappiness  60 |

It means it’s going to start swapping out (move to disk)when 40% of physical memory is used

You can change swappiness value by editing /etc/sysctl.conf file

Add line add the end of file vm.swappiness = <value> like vm.swappiness = 10



Reboot system

To reduce the swappiness set vm.swappiness =0

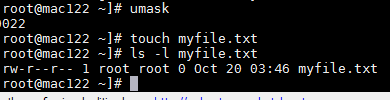
# Linux file permission umask

Umask stands for User file creation MASK

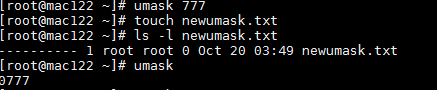
When we create some file or folder permission given according to umask value

You can check value by issuing command umask at terminal

|  |  |  |
| --- | --- | --- |
| **mask digit** | **resulting default file permissions** | **resulting default directory permissions** |
| **0** | rw | rwx |
| **1** | rw | rwx |
| **2** | r | rx |
| **3** | r | r |
| **4** | w | wx |
| **5** | w | w |
| **6** | x | x |
| **7** | (no permission allowed) | (no permission allowed) |



You can change umask by issuing command umask <new value>

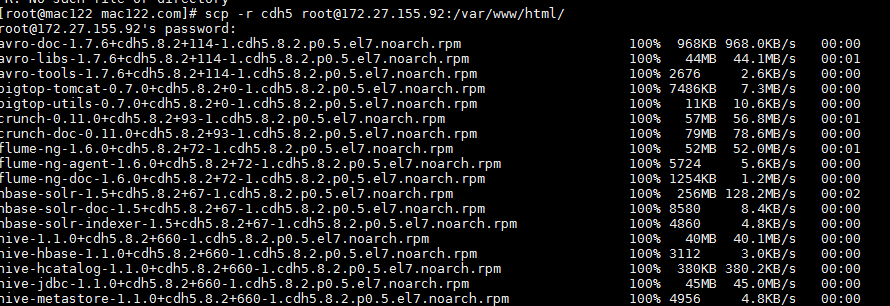


First digit for owner , second for group and third for others

# SCP Command

Here we are moving cdh5 folder with all files from mac122 to mac92 under /var/www/html folder

|  |
| --- |
| [root@mac122 mac122.com]# scp -r cdh5 root@172.27.155.92:/var/www/html/ |



Here we are at machine mac122 and copying cdh5 folder from mac53 which resides under /var/www/mac53.com

|  |
| --- |
| scp -r root@172.27.155.53:/var/www/mac53.com/cdh5 ./ |

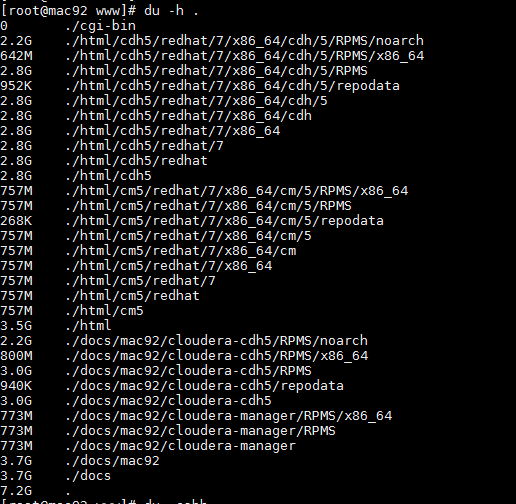
# Find the size of folder

du -csbh .



To find out each folder size under one directory

du -h .



# Tee Command

In computing, **tee** is a **command** in **command**-line interpreters (shells) using standard streams which reads standard input and writes it to both standard output and one or more files, effectively duplicating its input. It is primarily used in conjunction with pipes and filters.

By default tee command overwrites the file. You can instruct tee command to append to the file using the option –a as shown below.

|  |
| --- |
| $ ls | tee –a file |

It writes to a file and shows the output at console as well

You can also write the output to multiple files as shown below.

|  |
| --- |
| $ ls | tee file1 file2 file3 |

We can move the standard output to file using command below

|  |
| --- |
| $ <linux command> | tee –a file 2>&1 |

|  |
| --- |
| **2** refer to the second file descriptor of the process, i.e. stderr.  **>** means redirection.  **&1** means the target of the redirection should be the same location as the first file descriptor, i.e. file |

|  |
| --- |
| $ <linux command> | tee –a /dev/null 2>&1 |

So this command first redirects stdout to **/dev/null** and then redirects stderr there as well. This effectively silences all output (regular or error) from the any linux command.

**/dev/null** redirects the command standard output to the null device, which is a special device which discards the information written to it

# Determine Current Shell

1. **echo $0** - will print the program name... which in the case of shell is the actual shell
2. **ps -ef | grep $$ | grep -v grep** - will look for current process ID in the list of running processes. Current process being shell, it will include shell. This is not 100% reliable as you might have OTHER processes whose ps listing includes the same number as shell's process ID, especially if that ID is a small # (e.g. if shell's PID is "5", you may find processes called "java5" or "perl5" in the same grep output!). Which presents the second problem to "ps" approach, on top of the shell name being not always reliable.
3. **echo $SHELL** The path to the current shell is in SHELL variable for any shell. The caveat for the last one is that if you launch a shell explicitly as a subprocess (e.g. it's not your login shell) you will get you login shell's value instead - if that's a possibility, use the ps or $0 approach.

# IO Redirection

Input and output in the Linux environment is distributed across three streams. These streams are:

* standard input (stdin)
* standard output (stdout)
* standard error (stderr)

The streams are also numbered:

* stdin (0)
* stdout (1)
* stderr (2)

During standard interactions between the user and the terminal, standard input is transmitted through the user's keyboard. Standard output and standard error are displayed on the user's terminal as text. Collectively, the three streams are referred to as the standard streams.

**Standard Input**

The standard input stream typically carries data from a user to a program. Programs that expect standard input usually receive input from a device, such as a keyboard. Standard input is terminated by reaching EOF (end-of-file). As described by its name, EOF indicates that there is no more data to be read.

To see standard input in action, run the cat program. Cat stands for concatenate, which means to link or combine something. It is commonly used to combine the contents of two files. When run on its own, cat opens a looping prompt.

cat

After opening cat, type a series of numbers as it is running.

|  |
| --- |
| 1  2  3  ctrl-d |

When you type a number and press enter, you are sending standard input to the running cat program, which is expecting said input. In turn, the cat program is sending your input back to the terminal display as standard output.

EOF can be input by the user by pressing ctrl-d. After the cat program receives EOF, it stops.

**Standard Output**

Standard output writes the data that is generated by a program. When the standard output stream is not redirected, it will output text to the terminal. Try the following example:

|  |
| --- |
| echo Sent to the terminal through standard output  or  echo “Sent to the terminal through standard output” |

When used without any additional options, the echo command displays any argument that is passed to it on the command line. An argument is something that is received by a program.

Run echo without any arguments:

|  |
| --- |
| echo |

It will return an empty line, since there are no arguments.

**Standard Error**

Standard error writes the errors generated by a program that has failed at some point in its execution. Like standard output, the default destination for this stream is the terminal display.

When a program's standard error stream is piped to a second program, the piped data (consisting of program errors) is simultaneously sent to the terminal as well.

Let's see a basic example of standard error using the ls command. ls lists a directory's contents.

When run without an argument, ls lists the contents within the current directory. If ls is run with a directory as an argument, it will list the contents of the provided directory.

|  |
| --- |
| ls % |

Since % is not an existing directory, this will send the following text to standard error:

ls: cannot access %: No such file or directory

**Stream Redirection**

Linux includes redirection commands for each stream. These commands write standard output to a file. If a non-existent file is targetted (either by a single-bracket or double-bracket command), a new file with that name will be created prior to writing.

Commands with a single bracket overwrite the destination's existing contents.

Overwrite

|  |
| --- |
| > - standard output  < - standard input  2> - standard error |

Commands with a double bracket do not overwrite the destination's existing contents.

Append

|  |
| --- |
| >> - standard output  << - standard input  2>> - standard error |

Let's see an example:

|  |
| --- |
| cat > write\_to\_me.txt  a  b  c  ctrl-d |

Here, cat is being used to write to a file, which is created as a result of the loop.

View the contents of writetome.txt using cat:

cat write\_to\_me.txt

It should have the following contents:

|  |
| --- |
| a  b  c |

Redirect cat to writetome.txt again, and enter three numbers.

|  |
| --- |
| cat > write\_to\_me.txt  1  2  3  ctrl-d |

When you use cat to view writetome.txt, you will see the following:

|  |
| --- |
| 1  2  3 |

The prior contents are no longer there, as the file was overwritten by the single-bracket command.

Do one more cat redirection, this time using double brackets:

|  |
| --- |
| cat >> write\_to\_me.txt  a  b  c  ctrl-d |

Open writetome.txt again, and you will see this:

|  |
| --- |
| 1  2  3  a  b  c |

The file now contains text from both uses of cat, as the second one did not override the first one.

# Run a Command in background

|  |
| --- |
| #nohup <some command> & |

This will run the command in background and append the log into nohup.out file

If you want to create new log file instead of nohup.out use below command

|  |
| --- |
| #nohup some\_command > log\_filename 2>errorlog\_filename & |

If you want to append both out and error log in same file use

|  |
| --- |
| #nohup some\_command > log\_filename 2>&1 & |

Example

|  |
| --- |
| [root@mac92 ~]# nohup kafka-server-start /etc/kafka/server.properties > kafkaserver.out 2>&1 &  [1] 28600 |

|  |
| --- |
| [root@mac92 ~]# tail -f kafkaserver.out  [2017-02-14 01:03:29,938] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,39] in 0 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,42] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,42] in 0 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,45] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,940] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,45] in 1 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,940] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,48] (kafka.coordinator.GroupMetadataManager) |

To check the process id of this command

|  |
| --- |
| [root@mac92 ~]# ps ax | grep kafka  28600 pts/3 Sl 0:09 java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -XX:+DisableExplicitGC -Djava.awt.headless=true -Xloggc:/var/log/kafka/kafkaServer-gc.log -verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -Dkafka.logs.dir=/var/log/kafka -Dlog4j.configuration=file:/etc/kafka/log4j.properties -cp :/usr/bin/../share/java/kafka/\*:/usr/bin/../share/java/confluent-support-metrics/\*:/usr/share/java/confluent-support-metrics/\* io.confluent.support.metrics.SupportedKafka /etc/kafka/server.properties  28693 pts/3 S+ 0:00 tail -f kafkaserver.out  32078 pts/4 S+ 0:00 grep --color=auto kafka  [root@mac92 ~]# |

Here we can see that process id is 28693

To kill this process we can use below command

|  |
| --- |
| [root@mac92 ~]# kill -9 28693 |

Check kafkaserver.out file again

|  |
| --- |
| [root@mac92 ~]# tail -f kafkaserver.out  [2017-02-14 01:03:29,938] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,39] in 0 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,42] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,42] in 0 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,939] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,45] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,940] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,45] in 1 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,940] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,48] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,941] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,48] in 1 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,941] INFO [Group Metadata Manager on Broker 0]: Loading offsets and group metadata from [\_\_consumer\_offsets,15] (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:29,949] INFO [Group Metadata Manager on Broker 0]: Finished loading offsets from [\_\_consumer\_offsets,15] in 8 milliseconds. (kafka.coordinator.GroupMetadataManager)  [2017-02-14 01:03:31,097] INFO [ReplicaFetcherManager on broker 0] Removed fetcher for partitions \_\_consumer\_offsets-22,\_\_consumer\_offsets-30,\_\_consumer\_offsets-8,\_\_consumer\_offsets-21,\_\_consumer\_offsets-4,\_\_consumer\_offsets-27,\_\_consumer\_offsets-7,\_\_consumer\_offsets-9,\_\_consumer\_offsets-46,\_\_consumer\_offsets-25,\_\_consumer\_offsets-35,\_\_consumer\_offsets-41,\_\_consumer\_offsets-33,\_\_consumer\_offsets-23,\_\_consumer\_offsets-49,\_schemas-0,AvroClicks-0,\_\_consumer\_offsets-47,\_\_consumer\_offsets-16,test-0,\_\_consumer\_offsets-28,\_\_consumer\_offsets-31,\_\_consumer\_offsets-36,\_\_consumer\_offsets-42,\_\_consumer\_offsets-3,\_\_consumer\_offsets-18,\_\_consumer\_offsets-37,\_\_consumer\_offsets-15,\_\_consumer\_offsets-24,\_\_consumer\_offsets-38,\_\_consumer\_offsets-17,\_\_consumer\_offsets-48,\_\_confluent.support.metrics-0,\_\_consumer\_offsets-19,\_\_consumer\_offsets-11,\_\_consumer\_offsets-13,\_\_consumer\_offsets-2,\_\_consumer\_offsets-43,\_\_consumer\_offsets-6,\_\_consumer\_offsets-14,\_\_consumer\_offsets-20,\_\_consumer\_offsets-0,\_\_consumer\_offsets-44,\_\_consumer\_offsets-39,\_\_consumer\_offsets-12,\_\_consumer\_offsets-45,\_\_consumer\_offsets-1,\_\_consumer\_offsets-5,\_\_consumer\_offsets-26,\_\_consumer\_offsets-29,\_\_consumer\_offsets-34,\_\_consumer\_offsets-10,\_\_consumer\_offsets-32,\_\_consumer\_offsets-40 (kafka.server.ReplicaFetcherManager)  [2017-02-14 01:03:34,194] INFO Monitored broker is now ready (io.confluent.support.metrics.MetricsReporter)  [2017-02-14 01:03:34,194] INFO Starting metrics collection from monitored broker... (io.confluent.support.metrics.MetricsReporter)  Killed |

We can see that this process has been killed

# SAR Command

SAR stands for System Activity Report, as its name suggest sar command is used to collect, report & save CPU, Memory, I/O usage in Unix like operating system. SAR command produces the reports on the fly and can also save the reports in the log files as well.

SAR is one of the utility inside sysstat. You can easily download and install it in your machine very easily through YUM, rpm or apt-get according to your OS.

## Install and Configure Sysstat

First, make sure the latest version of sar is available on your system. Install it using any one of the following methods depending on your distribution.

sudo apt-get install sysstat

(or)

yum install sysstat

(or)

rpm -ivh sysstat-10.0.0-1.i586.rpm

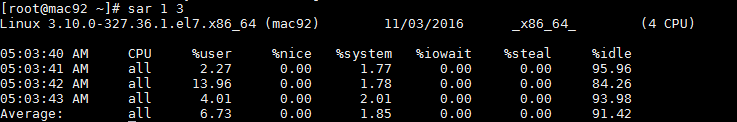
To check the version of SAR use

|  |
| --- |
| sar -V |



Finally, make sure sar works. For example, the following gives the system CPU statistics 3 times (with 1 second interval).

|  |
| --- |
| sar 1 3 |



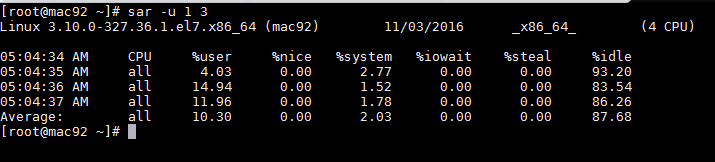
All the sar command will have the following as the 1st line in its output.

|  |
| --- |
| [root@mac92 ~]# sar -u 1 3  Linux 3.10.0-327.36.1.el7.x86\_64 (mac92) 11/03/2016 \_x86\_64\_ (4 CPU) |

* Linux 3.10.0-327.36.1.el7.x86\_64 – Linux kernel version of the system.
* (mac92) – The hostname where the sar data was collected.
* 11/03/2016 – The date when the sar data was collected.
* \_x86\_64\_ – The system architecture
* (4 CPU) – Number of CPUs available on this system. On multi core systems, this indicates the total number of cores.

## CPU Usage of ALL CPUs (sar -u)

This gives the cumulative real-time CPU usage of all CPUs. Option “1 3”- reports for every 1 second, total of 3 times. Most likely you’ll focus on the last field “%idle” to see the CPU load



Following are few variations:

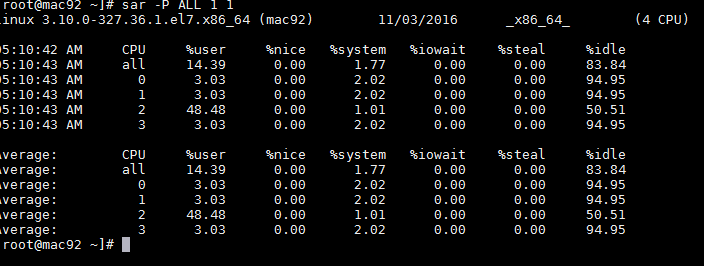
* **sar -u** Displays CPU usage for the current day that was collected until that point.
* **sar -u 1 3** Displays real time CPU usage every 1 second for 3 times.
* **sar -u ALL** Same as “sar -u” but displays additional fields.
* **sar -u ALL 1 3** Same as “sar -u 1 3” but displays additional fields.
* **sar -u -f /var/log/sa/sa10** Displays CPU usage for the 10day of the month from the sa10 file.

## CPU Usage of Individual CPU or Core (sar -P)

If you have 4 Cores on the machine and would like to see what the individual cores are doing, do the following.

“-P ALL” indicates that it should displays statistics for ALL the individual Cores.

In the following example under “CPU” column 0, 1, 2, and 3 indicates the corresponding CPU core numbers.

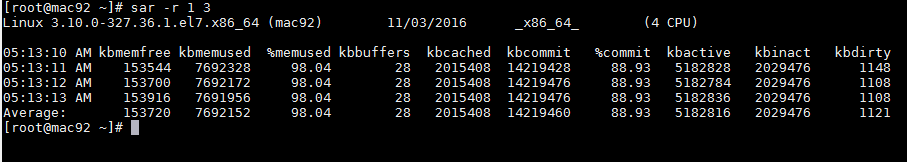


Following are few variations:

* **sar -P ALL** Displays CPU usage broken down by all cores for the current day.
* **sar -P ALL 1 3** Displays real time CPU usage for ALL cores every 1 second for 3 times (broken down by all cores).
* **sar -P 1** Displays CPU usage for core number 1 for the current day.
* **sar -P 1 1 3** Displays real time CPU usage for core number 1, every 1 second for 3 times.
* **sar -P ALL -f /var/log/sa/sa10** Displays CPU usage broken down by all cores for the 10day day of the month from sa10 file.

## Memory Free and Used (sar -r)

This reports the memory statistics. Option - “1 3” reports for every 1 second a total of 3 times. Most likely you’ll focus on “kbmemfree” and “kbmemused” for free and used memory.

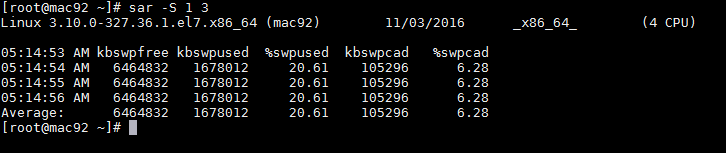


Following are few variations:

* sar -r
* sar -r 1 3
* sar -r -f /var/log/sa/sa10

## Swap Space Used (sar -S)

This reports the swap statistics. Option - “1 3” reports for every 1 second a total of 3 times. If the “kbswpused” and “%swpused” are at 0, then your system is not swapping.

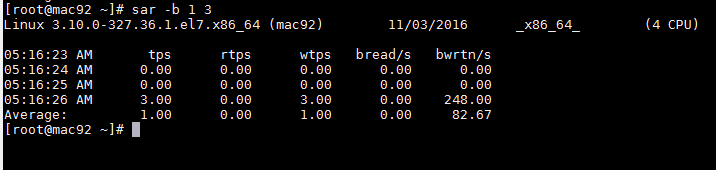


## Overall I/O Activities (sar -b)

This reports I/O statistics. Option - “1 3” reports for every 1 second a total of 3 times.

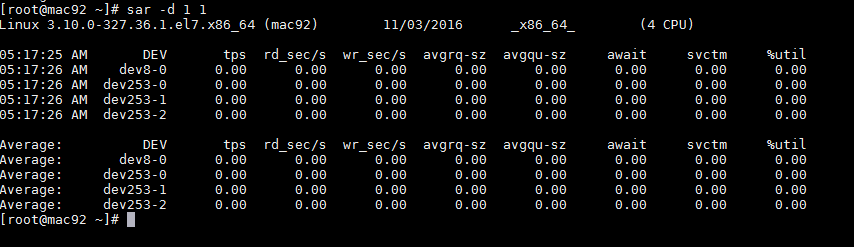
Following fields are displays in the example below.

* tps – Transactions per second (this includes both read and write)
* rtps – Read transactions per second
* wtps – Write transactions per second
* bread/s – Bytes read per second
* bwrtn/s – Bytes written per second



## Individual Block Device I/O Activities (sar -d)

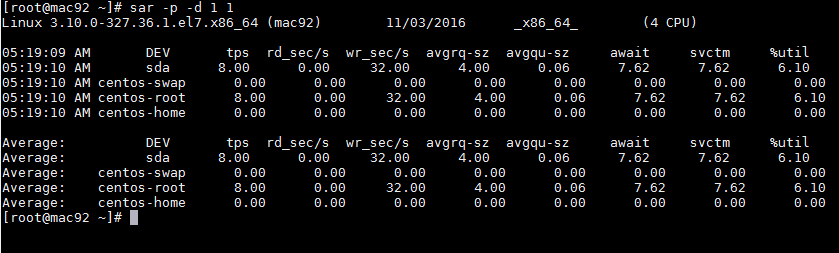
To identify the activities by the individual block devices (i.e a specific mount point, or LUN, or partition), use “sar -d”



In the above example “DEV” indicates the specific block device.

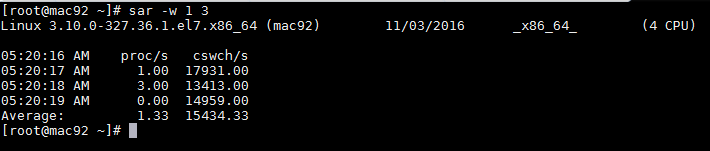
For example: “dev253-1” means a block device with 253 as major number, and 1 as minor number.

The device name (DEV column) can display the actual device name (for example: sda, sda1, sdb1 etc.,), if you use the -p option (pretty print) as shown below.



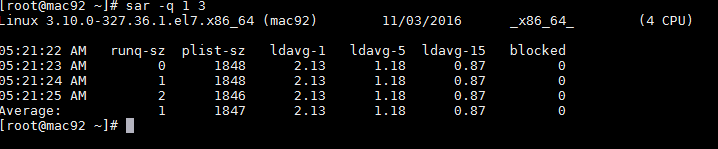
## Display context switch per second (sar -w)

This reports the total number of processes created per second and total number of context switches per second. Option - “1 3” reports for every 1 second a total of 3 times.



## Reports run queue and load average (sar -q)

This reports the run queue size and load average of last 1 minute, 5 minutes, and 15 minutes. Option - “1 3” reports for every 1 second a total of 3 times.



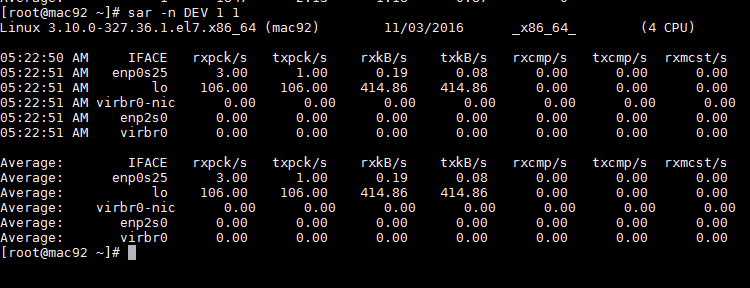
## Report network statistics (sar -n)

This reports various network statistics. For example: number of packets received (transmitted) through the network card, statistics of packet failure etc.,.Option - “1 3” reports for every 1 seconds a total of 3 times.

sar -n KEYWORD

KEYWORD can be one of the following:

* DEV – Displays network devices vital statistics for eth0, eth1, etc.,
* EDEV – Display network device failure statistics
* NFS – Displays NFS client activities
* NFSD – Displays NFS server activities
* SOCK – Displays sockets in use for IPv4
* IP – Displays IPv4 network traffic
* EIP – Displays IPv4 network errors
* ICMP – Displays ICMPv4 network traffic
* EICMP – Displays ICMPv4 network errors
* TCP – Displays TCPv4 network traffic
* ETCP – Displays TCPv4 network errors
* UDP – Displays UDPv4 network traffic
* SOCK6, IP6, EIP6, ICMP6, UDP6 are for IPv6
* ALL – This displays all of the above information. The output will be very long.



## Report Sar Data Using Start Time (sar -s)

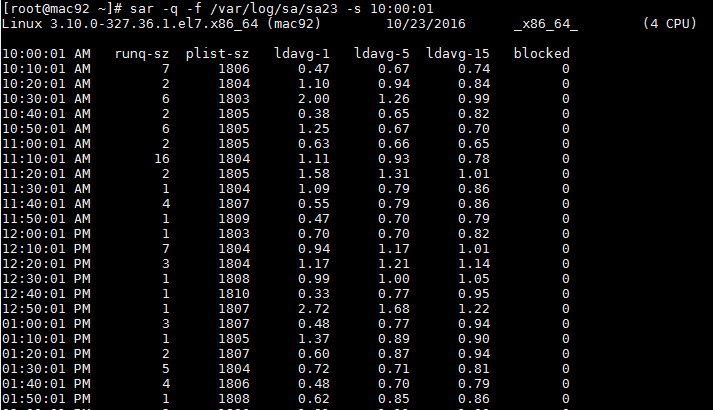
When you view historic sar data from the /var/log/sa/saXX file using “sar -f” option, it displays all the sar data for that specific day starting from 12:00 a.m for that day.

Using “-s hh:mi:ss” option, you can specify the start time. For example, if you specify “sar -s 10:00:00”, it will display the sar data starting from 10 a.m (instead of starting from midnight) as shown below.

You can combine -s option with other sar option.

For example, to report the load average on 26th of this month starting from 10 a.m in the morning, combine the -q and -s option as shown below.

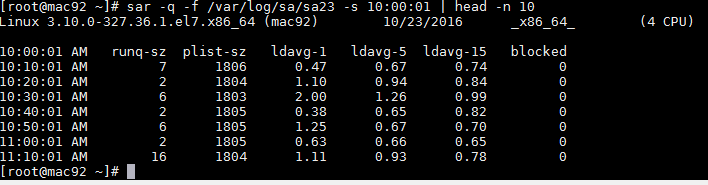
|  |
| --- |
| sar -q -f /var/log/sa/sa23 -s 10:00:01 |



There is no option to limit the end-time. You just have to get creative and use head command as shown below.

For example, starting from 10 a.m, if you want to see 7 entries, you have to pipe the above output to “head -n 10”.

|  |
| --- |
| sar -q -f /var/log/sa/sa23 -s 10:00:01 | head -n 10 |

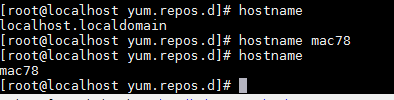


# Setting Up hostname

Make entry in /etc/hosts file for ip and hostname for all machines in network

|  |
| --- |
| 172.27.155.78 mac78  172.27.155.55 mac55  172.27.155.53 mac53  172.27.155.122 mac122 |

Update hostname of machine using hostname <name\_of\_host> command



Check host name by running hostnamectl command

|  |
| --- |
| [root@mac127 cloudera-scm-server]# hostnamectl  Static hostname: Vaibhav2  Transient hostname: mac127  Icon name: computer-desktop  Chassis: desktop  Machine ID: a59b056ce9db4e8b97089164752af8b8  Boot ID: 058089751f634a389b1b0f8e06952c98  Operating System: CentOS Linux 7 (Core)  CPE OS Name: cpe:/o:centos:centos:7  Kernel: Linux 3.10.0-327.el7.x86\_64  Architecture: x86-64 |

You can permanently change host name by command

hostnamectl set-hostname Your-New-Host-Name-Here

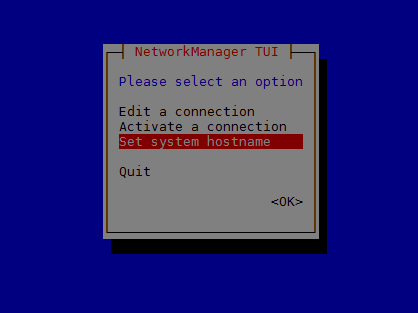
|  |
| --- |
| [root@mac127 cloudera-scm-server]# hostnamectl set-hostname mac127  [root@mac127 cloudera-scm-server]# hostnamectl  Static hostname: mac127  Icon name: computer-desktop  Chassis: desktop  Machine ID: a59b056ce9db4e8b97089164752af8b8  Boot ID: 058089751f634a389b1b0f8e06952c98  Operating System: CentOS Linux 7 (Core)  CPE OS Name: cpe:/o:centos:centos:7  Kernel: Linux 3.10.0-327.el7.x86\_64  Architecture: x86-64 |

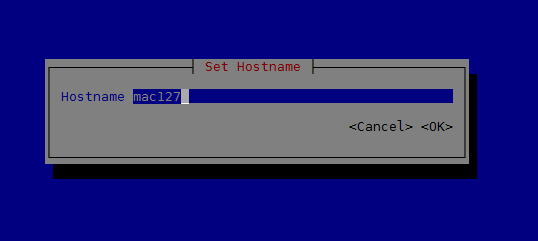
How do I delete a particular host name?

The syntax is:  
# hostnamectl set-hostname ""

Or you can use command nmtui

[root@mac127 cloudera-scm-server]# nmtui



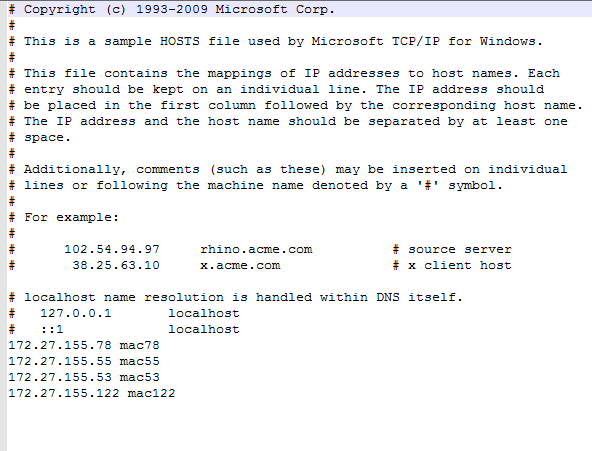


If you are not able to ping the linux machine by hostname from window machine, you need to add entries in hosts file

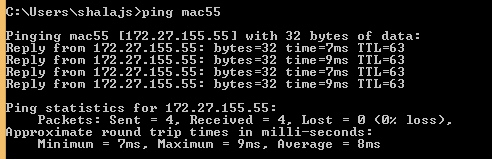


If you want to use hostname in window you need to update hosts file placed under C:\Windows\System32\drivers\etc

Make same entries as we did in linux hosts file

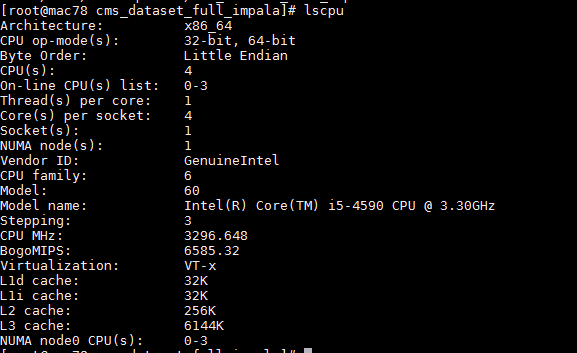


Now try to ping the machine using hostname



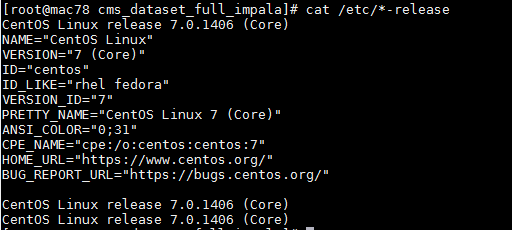
# Hardware/OS/Memory information

|  |
| --- |
| [root@mac92 ~]# lscpu |



OS information

|  |
| --- |
| [root@mac78 ]# cat /etc/\*-release |



Memory info

|  |
| --- |
| [root@mac92 etc]# **cat /proc/meminfo**  MemTotal: 7845908 kB  MemFree: 4295992 kB  MemAvailable: 5489456 kB  Buffers: 2660 kB  Cached: 1818480 kB  SwapCached: 0 kB  Active: 2338752 kB  Inactive: 812064 kB  Active(anon): 1545432 kB  Inactive(anon): 317660 kB  Active(file): 793320 kB  Inactive(file): 494404 kB  Unevictable: 0 kB  Mlocked: 0 kB  SwapTotal: 8142844 kB  SwapFree: 8142844 kB  Dirty: 32 kB  Writeback: 0 kB  AnonPages: 1329680 kB  Mapped: 170116 kB  Shmem: 533412 kB  Slab: 218888 kB  SReclaimable: 149368 kB  SUnreclaim: 69520 kB  KernelStack: 7568 kB  PageTables: 29072 kB  NFS\_Unstable: 0 kB  Bounce: 0 kB  WritebackTmp: 0 kB  CommitLimit: 12065796 kB  Committed\_AS: 3924364 kB  VmallocTotal: 34359738367 kB  VmallocUsed: 361796 kB  VmallocChunk: 34359369728 kB  HardwareCorrupted: 0 kB  AnonHugePages: 882688 kB  HugePages\_Total: 0  HugePages\_Free: 0  HugePages\_Rsvd: 0  HugePages\_Surp: 0  Hugepagesize: 2048 kB  DirectMap4k: 97008 kB  DirectMap2M: 8124416 kB |

# Find File that contains given pattern

|  |
| --- |
| [root@mac127 current]# find . -name \\*1073741874\\*  ./finalized/subdir0/subdir0/blk\_1073741874\_1050.meta  ./finalized/subdir0/subdir0/blk\_1073741874 |

Here we are trying to find a file in current directory (.) that contains 1073741874 in name of the file

You can use filename met characters (such as “\*”), but you should either put an escape character (“\”) in front of each of them or enclose them in quotes.

|  |
| --- |
| [root@mac127 current]# find . -name "\*1073741874\*"  ./finalized/subdir0/subdir0/blk\_1073741874\_1050.meta  ./finalized/subdir0/subdir0/blk\_1073741874 |

# Create MD5 files

An MD5 sum is a string of letters and numbers that acts like a fingerprint for a file. If two files have the same MD5 sum, the files are exactly alike - which is why MD5 "fingerprints" can verify whether or not your downloaded file got corrupted in transit.

|  |
| --- |
| **Create files and add those files under md5 file**  [root@mac92 md5]# ls -l  total 0  [root@mac92 md5]# cat > sample1.txt  this is sample 1 file  [root@mac92 md5]# cat > sample2.txt  this is sample 2 file  [root@mac92 md5]# cat > sample3.txt  this is sample 3 file  [root@mac92 md5]#  [root@mac92 md5]# md5sum sample1.txt sample2.txt sample3.txt > hash.md5  [root@mac92 md5]# **cat hash.md5**  a3fd769ef90f70882bfaa49fb2d35255 sample1.txt  6c22175288a0be219366e76e70fe9fb3 sample2.txt  652755190be42f37513c32b841d8ff02 sample3.txt |
| **Check if files are OK (calculated hash on md5 is equal to the file hash)**  [root@mac92 md5]# md5sum -c hash.md5  sample1.txt: OK  sample2.txt: OK  sample3.txt: OK  [root@mac92 md5]# cat >>sample3.txt  More content  [root@mac92 md5]# |
| **Update sample3.txt and check the hash again**  [root@mac92 md5]# cat >>sample3.txt  More content  [root@mac92 md5]# md5sum -c hash.md5  sample1.txt: OK  sample2.txt: OK  sample3.txt: FAILED  md5sum: WARNING: 1 computed checksum did NOT match  [root@mac92 md5]# |