Important Linux Administrative commands

Introduction

This course is only intended to give a basic overview of commands which will be required for taking next step towards Big Data Area specially Hadoop Configuration and Administration.

Copying Files

cp (copy)

cp *file1 file2* is the command which makes a copy of **file1** in the current working directory and calls it **file2**

What we are going to do now, is to take a file stored in an open access area of the file system, and use the **cp** command to copy it to your unixstuff directory.

First, **cd** to your unixstuff directory.

```
% cd ~/unixstuff
```

Then at the UNIX prompt, type,

```
% cp /vol/examples/tutorial/science.txt .
```

(Note: Don't forget the dot (**.**) at the end. Remember, in UNIX, the dot means the current directory.)

The above command means copy the file **science.txt** to the current directory, keeping the name the same.

(Note: The directory **/vol/examples/tutorial/** is an area to which everyone in the department has read and copy access. If you are from outside the University, you can grab a copy of the file here. Use 'File/Save As..' from the menu bar to save it into your **unixstuff** directory.)

Exercise 2a

Create a backup of your **science.txt** file by copying it to a file called **science.bak**

Moving files

mv (move)

mv *file1 file2* moves (or renames) **file1** to **file2**

To move a file from one place to another, use the **mv** command. This has the effect of moving rather than copying the file, so you end up with only one file rather than two.

It can also be used to rename a file, by moving the file to the same directory, but giving it a different name.

We are now going to move the file science.bak to your backup directory.

First, change directories to your unixstuff directory (can you remember how?). Then, inside the **unixstuff** directory, type

% mv science.bak backups/.

Type **ls** and **ls backups** to see if it has worked.

Removing files and directories

rm (remove), rmdir (remove directory)

To delete (remove) a file, use the **rm** command. As an example, we are going to create a copy of the **science.txt** file then delete it.

Inside your **unixstuff** directory, type

% cp science.txt tempfile.txt
% ls (to check if it has created the file)
% rm tempfile.txt
% ls (to check if it has deleted the file)

You can use the **rmdir** command to remove a directory (make sure it is empty first). Try to remove the **backups** directory. You will not be able to since UNIX will not let you remove a non-empty directory.

Exercise 2b

Create a directory called **tempstuff** using **mkdir** , then remove it using the **rmdir** command.

Displaying the contents of a file on the screen

clear (clear screen)

Before you start the next section, you may like to clear the terminal window of the previous commands so the output of the following commands can be clearly understood.

At the prompt, type

```
% clear
```

This will clear all text and leave you with the % prompt at the top of the window.

cat (concatenate)

The command **cat** can be used to display the contents of a file on the screen. Type:

```
% cat science.txt
```

As you can see, the file is longer than than the size of the window, so it scrolls past making it unreadable.

less

The command **less** writes the contents of a file onto the screen a page at a time. Type

% less science.txt

Press the **[space-bar]** if you want to see another page, type **[q]** if you want to quit reading. As you can see, **less** is used in preference to **cat** for long files.

head

The **head** command writes the first ten lines of a file to the screen.

First clear the screen then type

% head science.txt

Then type

% head -5 science.txt

What difference did the -5 do to the head command?

tail

The **tail** command writes the last ten lines of a file to the screen.

Clear the screen and type

% tail science.txt

How can you view the last 15 lines of the file?

Searching the contents of a file

Simple searching using less

Using **less**, you can search though a text file for a keyword (pattern). For example, to search through **science.txt** for the word 'science', type

```
% less science.txt
```

then, still in **less** (i.e. don't press [q] to quit), type a forward slash **[/]** followed by the word to search

```
/science
```

As you can see, **less** finds and highlights the keyword.
Type **[n]** to search for the next occurrence of the word.

grep (don't ask why it is called grep)

grep is one of many standard UNIX utilities. It searches files for specified words or patterns. First clear the screen, then type

```
% grep science science.txt
```

As you can see, **grep** has printed out each line containg the word science.

Or has it????

Try typing

```
% grep Science science.txt
```

The **grep** command is case sensitive; it distinguishes between Science and science.

To ignore upper/lower case distinctions, use the -i option, i.e. type

```
% grep -i science science.txt
```

To search for a phrase or pattern, you must enclose it in single quotes (the apostrophe symbol). For example to search for spinning top, type

% grep -i 'spinning top' science.txt

Some of the other options of grep are:

-v display those lines that do NOT match
-n precede each maching line with the line number
-c print only the total count of matched lines

Try some of them and see the different results. Don't forget, you can use more than one option at a time, for example, the number of lines without the words science or Science is

% grep -ivc science science.txt

wc (word count)

A handy little utility is the **wc** command, short for word count. To do a word count on **science.txt**, type

% wc -w science.txt

To find out how many lines the file has, type

% wc -l science.txt

Redirecting the Input

We use the < symbol to redirect the input of a command.

The command **sort** alphabetically or numerically sorts a list. Type

% sort

Then type in the names of some vegetables. Press **[Return]** after each one.

```
carrot
beetroot
artichoke
^D (control d to stop)
```

The output will be

```
artichoke
beetroot
carrot
```

Using < you can redirect the input to come from a file rather than the keyboard. For example, to sort the list of fruit, type

```
% sort < biglist
```

and the sorted list will be output to the screen.

To output the sorted list to a file, type,

```
% sort < biglist > slist
```

Use **cat** to read the contents of the file **slist**

Processes and Jobs

A process is an executing program identified by a unique PID (process identifier). To see information about your processes, with their associated PID and status, type

```
% ps
```

A process may be in the foreground, in the background, or be suspended. In general the shell does not return the UNIX prompt until the current process has finished executing.

Some processes take a long time to run and hold up the terminal. Backgrounding a long process has the effect that the UNIX prompt is returned immediately, and other tasks can be carried out while the original process continues executing.

Running background processes

To background a process, type an **&** at the end of the command line. For example, the command **sleep** waits a given number of seconds before continuing. Type

% sleep 10

This will wait 10 seconds before returning the command prompt %. Until the command prompt is returned, you can do nothing except wait.

To run **sleep** in the background, type

% sleep 10 &

[1] 6259

The **&** runs the job in the background and returns the prompt straight away, allowing you do run other programs while waiting for that one to finish.

The first line in the above example is typed in by the user; the next line, indicating job number and PID, is returned by the machine. The user is be notified of a job number (numbered from 1) enclosed in square brackets, together with a PID and is notified when a background process is finished. Backgrounding is useful for jobs which will take a long time to complete.

Backgrounding a current foreground process

At the prompt, type

% sleep 100

You can suspend the process running in the foreground by holding down the **[control]** key and typing **[z]** (written as **^Z**) Then to put it in the background, type

% bg

Note: do not background programs that require user interaction e.g. pine

Listing suspended and background processes

When a process is running, backgrounded or suspended, it will be entered onto a list along with a job number. To examine this list, type

% jobs

An example of a job list could be

[1] Suspended sleep 100
[2] Running netscape
[3] Running nedit

To restart (foreground) a suspended processes, type

% fg %jobnumber

For example, to restart **sleep 100**, type

% fg %1

Typing **fg** with no job number foregrounds the last suspended process.

Killing a process

kill (terminate or signal a process)

It is sometimes necessary to kill a process (for example, when an executing program is in an infinite loop)

To kill a job running in the foreground, type **^C** (control c). For example, run

% sleep 100
^C

To kill a suspended or background process, type

% kill %jobnumber

For example, run

```
% sleep 100 &
% jobs
```

If it is job number 4, type

```
% kill %4
```

To check whether this has worked, examine the job list again to see if the process has been removed.

ps (process status)

Alternatively, processes can be killed by finding their process numbers (PIDs) and using **kill *PID_number***

```
% sleep 100 &
% ps

PID TT S TIME COMMAND
20077 pts/5 S 0:05 sleep 100
21563 pts/5 T 0:00 netscape
21873 pts/5 S 0:25 nedit
```

To kill off the process **sleep 100**, type

```
% kill 20077
```

and then type **ps** again to see if it has been removed from the list.

If a process refuses to be killed, uses the **-9** option, i.e. type

```
% kill -9 20077
```

> Note: It is not possible to kill off other users' processes!!!
> Unless you are a system administrator

Create User

```
# adduser -g wheel -s /bin/bash -d /home/hadoop Hadoop
```

Check IP address

```
# ip addr show
```

Check CPU

```
# more /proc/cpuinfo | grep 'core id' | wc –l
# lscpu
[root@nakparsdev-1-vm-01 ~]# nproc --all
8
```

Memory Check:

```
# more /proc/meminfo | grep 'Mem'
```

Set All the Host Names

To set all the host names on a system, enter the following command as root:

```
# hostnamectl set-hostname datasciencelab
```

Mount CD-ROM for making RPM Package:

```
# ls -al /dev/cdrom*
lrwxrwxrwx 1 root root 3 Nov  9 13:06 /dev/cdrom -> sr0

# mkdir /media/cdrom
```

```
# mount -t iso9660 /dev/scd0 /media/cdrom
 mount: block device /dev/sr0 is write-protected, mounting read- only
# df -Th
```

While un-mounting:

```
# umount /dev/hdc
# eject
```

f you are unable to unmount your previously mounted CDROM, make sure that you are not in the directory where your CDROM is mounted or that some other application is not using it.

If you are still having problems to unmount your CDROM medium you can use fuser command to kill all related processes using your device.

```
# fuser -mk /dev/hdc
# eject
```

Make Yum repository from RHEL 7 CDROM

```
# cp -rp /media/cdrom/* /rhel7repo/
```

Navigate to /etc/yum.repos.d/ directory .

Create a new file with below contents with extension of ".repo" .

```
[root@UnixArena-RHEL7 yum.repos.d]# cat unixarena-rhel7.repo
[rhel7_dvd]
gpgcheck = 0
enabled = 1
baseurl = file:///var/ftp/pub/rhel7repo
name = nakredhat-repo
[root@UnixArena-RHEL7 yum.repos.d]#
```

For second

```
[root@UnixArena-RHEL7 yum.repos.d]# cat unixarena-rhel7.repo
[rhel7_dvd]
gpgcheck = 0
enabled = 1
baseurl = ftp://172.26.7.195/pub/rhel7repo/
name = nakredhat-repo
[root@UnixArena-RHEL7 yum.repos.d]#
```

5. Run “yum clean all”

```
[root@UnixArena-RHEL7 yum.repos.d]# yum clean all
Loaded plugins: langpacks, product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
Cleaning repos: rhel7_dvd
Cleaning up everything
[root@UnixArena-RHEL7 yum.repos.d]#
```

6.Test the new yum repository by installing new package.

```
[root@UnixArena-RHEL7 yum.repos.d]# yum install telnet
```

Adding RHEL Extended Package:

```
[root@nakparsdev-2-vm-01 yum.repos.d]# more nakrhelepl7.repo
[rhel7repo]
gpgcheck = 0
enabled = 1
baseurl = https://dl.fedoraproject.org/pub/epel/7/x86_64/
name = nak-repo-epl
[root@nakparsdev-2-vm-01 yum.repos.d]#
```

Note: If you find any **certification problem (peer certification expired)** the temporary solution is to use http rather than https

Install FTP Server

yum install vsftpd* -y

service vsftpd restart

chkconfig vsftpd on

yum install ftp-*

mv few files under /var/ftp/pub for remote machine access.

service vsftpd restart

Check from browser:

ftp://172.26.7.195/pub/rhel7repo/Packages/

Understanding DNS Resolution

```
# more /etc/hosts
# more /etc/resolv.conf
# nslookup
```

IP Troubleshooting Basics

```
# ping -s 128 192.168.43.214
# traceroute 192.168.43.214
# tcpdump host 192.168.43.214
# arp –v
```

System Shut Down or Restart

```
# init 6 #Restart
# init 0 #Power off
# shutdown –h now #Shutdown now
# shutdown –h +10 #Shutdown after 10 minutes
```

shutdown –r now
reboot –h now

Control default system run level

```
#  systemctl get-default
runlevel5.target
```

```
# systemctl list-units --type=target
```

The output will look like below.

```
UNIT                   LOAD   ACTIVE SUB    DESCRIPTION
basic.target           loaded active active Basic System
cryptsetup.target      loaded active active Encrypted Volumes
getty.target           loaded active active Login Prompts
graphical.target       loaded active active Graphical Interface
local-fs-pre.target    loaded active active Local File Systems (Pre)
local-fs.target        loaded active active Local File Systems
multi-user.target      loaded active active Multi-User System
network.target         loaded active active Network
nfs.target             loaded active active Network File System Server
paths.target           loaded active active Paths
remote-fs.target       loaded active active Remote File Systems
slices.target          loaded active active Slices
sockets.target         loaded active active Sockets
swap.target            loaded active active Swap
sysinit.target         loaded active active System Initialization
timers.target          loaded active active Timers

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.
```

Change default to runlevel 3 (nothing but a multi-user.target).

```
# systemctl set-default multi-user.target
```

Confirm the default runlevel.

```
# systemctl get-default
multi-user.target
```

Reboot and check it out.

```
# reboot
```