

Linux command line tools

24/10/17

Prepared for the Centre of System Genomics

Created by Bobbie Shaban

Centre for System Genomics

This document outlines the material for the tutorial. This tutorial will also be recorded and uploaded to the Genomic Databases Resource Hub (COGENT) as a webinar for download: <https://blogs.unimelb.edu.au/system-genomics/>

This tutorial assumes that you have an account. If you don’t have an account please contact your group leader to give you access.

# 1. Subject

1. Linux command line tools

# 2. Glossary

**chmod:** A utility that changes file permissions.

**chown**: the low-level software that supports a computer's basic functions, such as scheduling tasks and controlling peripherals.

**find**: the low-level software that supports a computer's basic functions, such as scheduling tasks and controlling peripherals.

**rsync**: the low-level software that supports a computer's basic functions, such as scheduling tasks and controlling peripherals.

**pwd:** Displays the name of the current working directory – aka "print working directory."

**grep:** Grep is a command used for searching one or more files for a given character string or pattern. It can also be used to replace the character string with another one.

**cat:** Abbreviated from the word "concatenate," which means to link things together, cat is used in Linux to link file contents and output them for viewing or printing.

**tar:** The tar program provides the ability to create tar archives. Short for “Tape Archiver”

**alias:** Allows substitution of a small or more familiar name in place of a long string.

**vimdiff:** Opens two files in vim and compares them

**group:** A Unix permission classification

# 3. Tutorial

Group

Unix groups can be used to share files with CS department users. Each user on the CS public machines is associated with a list containing at least one group, and each file or directory on the CS public machines is associated with exactly one group. This is usually referred to as group membership and group ownership, respectively. That is, users are in groups and files are owned by a group.

## UNIX Commands for Working with Groups

|  |  |  |
| --- | --- | --- |
| **Command** | **Description** | **Example** |
| groups | See groups to which you belong with primary group first | groups |
| id | See current group as part of your id | id |
| newgrp | Start a shell in a different group | newgrp project1 |
| chmod | Change permissions for directories and files | chmod g+rwx myfile |
| chgrp | Change group ownership of directories and files | chgrp project1 myfile |
| ls | List file permissions | ls -l |

Examples.

What groups do you belong to?

|  |
| --- |
| [bshaban@snowy-sg1 ~]$ groupsSG0009 SG0004 SG0005 sysgen SGN0001 |

What’s your id?

|  |
| --- |
| [bshaban@snowy-sg1 ~]$ iduid=4997(bshaban) gid=3767(SG0009) groups=3767(SG0009),3659(SG0004),3670(SG0005),3698(sysgen),3714(SGN0001) |

My current group is (SG0009).

Main group is set in kaarage when you sign up. You have a kaarage account you can log into and change your main group if you have been added to more than one group. <https://my.vlsci.org.au/karaage/profile/login/?next=/karaage/projects/>

Chmod

In general, **chmod** commands take the form:

chmod *options* *permissions* *file name*

*permissions* defines the permissions for the owner of the file (the "user"), members of the group who owns the file (the "group"), and anyone else ("others"). There are two ways to represent these permissions: with symbols ([alphanumeric](https://www.computerhope.com/jargon/a/alphanum.htm) [characters](https://www.computerhope.com/jargon/c/charact.htm)), or with [octal](https://www.computerhope.com/jargon/o/octal.htm) numbers (the digits **0** through **7**).

Let's say you are the owner of a file named **myfile**, and you want to set its permissions so that:

1. the **u**ser can **r**ead, **w**rite, ande **x**ecute it;
2. members of your **g**roup can **r**ead ande **x**ecute it; and
3. **o**thers may only **r**ead it.

This command will do the trick:

chmod u=rwx,g=rx,o=r myfile

This example uses symbolic permissions notation. The letters **u**, **g**, and **o** stand for "**user**", "**group**", and "**other**".

The equals sign ("**=**") means "set the permissions exactly like this," and the letters "**r**", "**w**", and "**x**" stand for "read", "write", and "execute", respectively. The commas separate the different classes of permissions, and there are no spaces in between them.

Here is the equivalent command using octal permissions notation:

chmod 754 myfile

Here the digits **7**, **5**, and **4** each individually represent the permissions for the user, group, and others, in that order. Each digit is a combination of the numbers **4**, **2**, **1**, and **0**:

* **4** stands for "read",
* **2** stands for "write",
* **1** stands for "execute", and
* **0** stands for "no permission."

So **7** is the combination of permissions **4**+**2**+**1** (read, write, and execute), **5** is **4**+**0**+**1**(read, no write, and execute), and **4** is **4**+**0**+**0** (read, no write, and no

Eg. Change permissions in directory recursively

chmod -R 755 directory-name/

Chown

What’s the difference between chown and chmod?

|  |  |
| --- | --- |
|  | In simple term chown is used to change the ownership of a file while chmod is for changing the file mode bits.* chown defines who owns the file.
* chmod defines who can do what.
 |



Create a file

touch rainbow

Let's have a look at the file's permissions

$ ls -l rainbow

-rw-r--r-- 1 bshaban SG0009 0 Oct 24 09:44 rainbow

The first part of the information is the file type (- at the beginning means it's a regular file) and the permission bits



In most cases chown should be enough.

find

To use the find command, at the [Unix](https://kb.iu.edu/d/agat) prompt, enter:

 find . -name "pattern" -print

Replace "pattern" with a filename or matching expression, such as "\*.txt". (Leave the double quotes in.)

 [bshaban@snowy-sg1 ~]$ find -name "rainbow" -print

./rainbow

./tute4/rainbow

You have several options for matching criteria:

|  |  |
| --- | --- |
| -atime n | File was accessed n days ago |
| -mtime n | File was modified n days ago |
| -size n | File is n blocks big (a block is 512 bytes) |
| -type c | Specifies file type: f=plain text, d=directory |
| -fstype typ | Specifies file system type: 4.2 or nfs |
| -name nam | The filename is nam |
| -user usr | The file's owner is usr |
| -group grp | The file's group owner is grp |
| -perm p | The file's access mode is p (where p is an integer) |

You can use + (plus) and - (minus) modifiers with the atime, mtime, and size criteria to increase their usefulness, for example:

|  |  |
| --- | --- |
| -mtime +7 | Matches files modified more than seven days ago |
| -atime -2 | Matches files accessed less than two days ago |
| -size +100 | Matches files larger than 100 blocks (50KB) |

By default, multiple options are joined by "and". You may specify "or" with the -o flag and the use of grouped parentheses. To match all files modified more than 7 days ago and accessed more than 30 days ago, use:

 \( -mtime +7 -atime +30 \)

To match all files modified more than 7 days ago or accessed more than 30 days ago, use:

 \( -mtime +7 -o -atime +30 \)

You may specify "not" with an exclamation point. To match all files ending in .txt except the file notme.txt, use:

 \! -name notme.txt -name \\*.txt

You can specify the following actions for the list of files that the find command locates:

|  |  |
| --- | --- |
| -print | Display pathnames of matching files. |
| -exec cmd | Execute command cmd on a file. |
| -ok cmd | Prompt before executing the command cmd on a file. |
| -mount | (System V) Restrict to file system of starting directory. |
| -xdev | (BSD) Restrict to file system of starting directory. |
| -prune | (BSD) Don't descend into subdirectories. |

Executed commands must end with \; (a backslash and semi-colon) and may use {} (curly braces) as a placeholder for each file that the find command locates. For example, for a long listing of each file found, use:

 -exec ls -l {} \;

Matching criteria and actions may appear in any order and are evaluated from left to right.

## Full examples

* To find and report all text files starting at the current directory, enter:

 find . -name \\*.txt -print

* To report all files starting in the directories /mydir1 and /mydir2 larger than 2,000 blocks (about 1,000KB) and that have not been accessed in over 30 days, enter:

 find /mydir1 /mydir2 -size +2000 -atime +30 -print

* To remove (with prompting) all files starting in the /mydir directory that have not been accessed in over 100 days, enter:

 find /mydir -atime +100 -ok rm {} \;

* To show a long listing starting in /mydir of files not modified in over 20 days or not accessed in over 40 days, enter:

 find /mydir \(-mtime +20 -o -atime +40\) -exec ls -l {} \;

* To list and remove all regular files named core starting in the directory /prog that are larger than 500KB, enter:

 find /prog -type f -size +1000 -print -name core -exec rm {} \;

Rsync

**Rsync** (**Remote Sync**) is a most commonly used command for **copying** and **synchronizing** files and directories **remotely** as well as **locally** in **Linux**/**Unix** systems. With the help of **rsync** command you can copy and synchronize your data remotely and locally across directories, across disks and networks, perform data backups and mirroring between two Linux machines.

##### **Some advantages and features of Rsync command**

1. It efficiently copies and sync files to or from a remote system.
2. Supports copying links, devices, owners, groups and permissions.
3. It’s faster than **scp** (**Secure Copy**) because **rsync** uses remote-update protocol which allows to transfer just the differences between two sets of files. First time, it copies the whole content of a file or a directory from source to destination but from next time, it copies only the changed blocks and bytes to the destination.
4. Rsync consumes less **bandwidth** as it uses compression and decompression method while sending and receiving data both ends.

##### **Basic syntax of rsync command**

# rsync options source destination

##### **Some common options used with rsync commands**

1. **-v** : verbose
2. **-r** : copies data recursively (but don’t preserve timestamps and permission while transferring data
3. **-a** : archive mode, archive mode allows copying files recursively and it also preserves symbolic links, file permissions, user & group ownerships and timestamps
4. **-z** : compress file data
5. **-h** : human-readable, output numbers in a human-readable format

##### **Copy/Sync a File on a Local Computer**

This following command will sync a single file on a local machine from one location to another location.

[[bshaban@snowy-sg1 tute4]$ rsync -zvh generic.tar backup/

created directory backup

generic.tar

sent 57.09M bytes received 31 bytes 4.96M bytes/sec

total size is 370.89M speedup is 6.50

[bshaban@snowy-sg1 tute4]$In above example, you can see that if the destination is not already exists rsync will create a directory automatically for destination.

##### **Copy/Sync a Directory on Local Computer**

The following command will transfer or sync all the files of from one directory to a different directory in the same machine.

[bshaban@snowy-sg1 tute4]$ rsync -avzh \* new\_backup

sending incremental file list

created directory new\_backup

generic.tar

generic.txt

rainbow

text.txt

backup/

backup/generic.tar

transfer

sent 171.26M bytes received 115 bytes 5.44M bytes/sec

total size is 1.11G speedup is 6.50

### 2. Copy/Sync Files and Directory to or From a Server

##### **Copy a Directory from Local Server to a Remote Server**

This command will sync a directory from a local machine to a remote machine. **(Is possible, I won’t go through)**

[root@tecmint]$ rsync -avz rpmpkgs/ root@192.168.0.101:/home/

root@192.168.0.101's password:

sending incremental file list

./

httpd-2.2.3-82.el5.centos.i386.rpm

mod\_ssl-2.2.3-82.el5.centos.i386.rpm

nagios-3.5.0.tar.gz

nagios-plugins-1.4.16.tar.gz

sent 4993369 bytes received 91 bytes 399476.80 bytes/sec

total size is 4991313 speedup is 1.00

##### **Copy/Sync a Remote Directory to a Local Machine**

This command will help you sync a remote directory to a local directory. Here in this example, a directory **/home/tarunika/rpmpkgs** which is on a remote server is being copied in your local computer in **/tmp/myrpms**.

[root@tecmint]# rsync -avzh root@192.168.0.100:/home/tarunika/rpmpkgs /tmp/myrpms

root@192.168.0.100's password:

receiving incremental file list

created directory /tmp/myrpms

rpmpkgs/

rpmpkgs/httpd-2.2.3-82.el5.centos.i386.rpm

rpmpkgs/mod\_ssl-2.2.3-82.el5.centos.i386.rpm

rpmpkgs/nagios-3.5.0.tar.gz

rpmpkgs/nagios-plugins-1.4.16.tar.gz

sent 91 bytes received 4.99M bytes 322.16K bytes/sec

total size is 4.99M speedup is 1.00

### 3. Rsync Over SSH

With rsync, we can use **SSH** (**Secure Shell**) for data transfer, using **SSH** protocol while transferring our data you can be ensured that your data is being transferred in a secured connection with encryption so that nobody can read your data while it is being transferred over the wire on the internet.

Also when we use rsync we need to provide the **user**/**root** password to accomplish that particular task, so using **SSH** option will send your logins in an encrypted manner so that your **password** will be safe.

##### **Copy a File from a Remote Server to a Local Server with SSH**

To specify a protocol with **rsync** you need to give “**-e**” option with protocol name you want to use. Here in this example, We will be using “**ssh**” with “**-e**” option and perform data transfer.

[root@tecmint]# rsync -avzhe ssh root@192.168.0.100:/root/install.log /tmp/

root@192.168.0.100's password:

receiving incremental file list

install.log

sent 30 bytes received 8.12K bytes 1.48K bytes/sec

total size is 30.74K speedup is 3.77

##### **Copy a File from a Local Server to a Remote Server with SSH**

[root@tecmint]# rsync -avzhe ssh backup.tar root@192.168.0.100:/backups/

root@192.168.0.100's password:

sending incremental file list

backup.tar

sent 14.71M bytes received 31 bytes 1.28M bytes/sec

total size is 16.18M speedup is 1.10

**Suggested Read:** [**Use Rsync to Sync New or Changed/Modified Files in Linux**](https://www.tecmint.com/sync-new-changed-modified-files-rsync-linux/)

### 4. Show Progress While Transferring Data with rsync

To show the progress while transferring the data from one machine to a different machine, we can use ‘**–progress’** option for it. It displays the files and the time remaining to complete the transfer.

[root@tecmint]# rsync -avzhe ssh --progress /home/rpmpkgs root@192.168.0.100:/root/rpmpkgs

root@192.168.0.100's password:

sending incremental file list

created directory /root/rpmpkgs

rpmpkgs/

rpmpkgs/httpd-2.2.3-82.el5.centos.i386.rpm

1.02M 100% 2.72MB/s 0:00:00 (xfer#1, to-check=3/5)

rpmpkgs/mod\_ssl-2.2.3-82.el5.centos.i386.rpm

99.04K 100% 241.19kB/s 0:00:00 (xfer#2, to-check=2/5)

rpmpkgs/nagios-3.5.0.tar.gz

1.79M 100% 1.56MB/s 0:00:01 (xfer#3, to-check=1/5)

rpmpkgs/nagios-plugins-1.4.16.tar.gz

2.09M 100% 1.47MB/s 0:00:01 (xfer#4, to-check=0/5)

sent 4.99M bytes received 92 bytes 475.56K bytes/sec

total size is 4.99M speedup is 1.00

### 5. Use of –include and –exclude Options

These two options allows us to **include** and **exclude** files by specifying parameters with these option helps us to specify those files or directories which you want to include in your sync and exclude files and folders with you don’t want to be transferred.

Here in this example, rsync command will include those files and directory only which starts with ‘**R**’ and exclude all other files and directory.

[root@tecmint]# rsync -avze ssh --include 'R\*' --exclude '\*' root@192.168.0.101:/var/lib/rpm/ /root/rpm

root@192.168.0.101's password:

receiving incremental file list

created directory /root/rpm

./

Requirename

Requireversion

sent 67 bytes received 167289 bytes 7438.04 bytes/sec

total size is 434176 speedup is 2.59

### 6. Use of –delete Option

If a file or directory not exist at the source, but already exists at the destination, you might want to delete that existing file/directory at the target while syncing .

We can use ‘**–delete**‘ option to delete files that are not there in source directory.

Source and target are in sync. Now creating new file **test.txt** at the target.

[root@tecmint]# touch test.txt

[root@tecmint]# rsync -avz --delete root@192.168.0.100:/var/lib/rpm/ .

Password:

receiving file list ... done

deleting test.txt

./

sent 26 bytes received 390 bytes 48.94 bytes/sec

total size is 45305958 speedup is 108908.55

Target has the new file called **test.txt**, when synchronize with the source with ‘**–delete**‘ option, it removed the file **test.txt**.

### 7. Set the Max Size of Files to be Transferred

You can specify the **Max** file size to be transferred or sync. You can do it with “**–max-size**” option. Here in this example, Max file size is **200k**, so this command will transfer only those files which are equal or smaller than **200k**.

[root@tecmint]# rsync -avzhe ssh --max-size='200k' /var/lib/rpm/ root@192.168.0.100:/root/tmprpm

root@192.168.0.100's password:

sending incremental file list

created directory /root/tmprpm

./

Conflictname

Group

Installtid

Name

Provideversion

Pubkeys

Requireversion

Sha1header

Sigmd5

Triggername

\_\_db.001

sent 189.79K bytes received 224 bytes 13.10K bytes/sec

total size is 38.08M speedup is 200.43

### 8. Automatically Delete source Files after successful Transfer

Now, suppose you have a main web server and a data backup server, you created a daily backup and synced it with your backup server, now you don’t want to keep that local copy of backup in your web server.

So, will you wait for transfer to complete and then delete those local backup file manually? Of Course NO. This automatic deletion can be done using ‘**–remove-source-files**‘ option.

[root@tecmint]# rsync --remove-source-files -zvh backup.tar /tmp/backups/

backup.tar

sent 14.71M bytes received 31 bytes 4.20M bytes/sec

total size is 16.18M speedup is 1.10

[root@tecmint]# ll backup.tar

ls: backup.tar: No such file or directory

### 9. Set Bandwidth Limit and Transfer File

You can set the bandwidth limit while transferring data from one machine to another machine with the the help of ‘**–bwlimit**‘ option. This options helps us to limit **I/O** bandwidth.

[root@tecmint]# rsync --bwlimit=100 -avzhe ssh /var/lib/rpm/ root@192.168.0.100:/root/tmprpm/

root@192.168.0.100's password:

sending incremental file list

sent 324 bytes received 12 bytes 61.09 bytes/sec

total size is 38.08M speedup is 113347.05

Also, by default rsync syncs changed blocks and bytes only, if you want explicitly want to sync whole file then you use ‘**-W**‘ option with it.

[root@tecmint]# rsync -zvhW backup.tar /tmp/backups/backup.tar

backup.tar

sent 14.71M bytes received 31 bytes 3.27M bytes/sec

total size is 16.18M speedup is 1.10

PWD/PUSHD/POPD

In **Unix**-like and some other operating systems, the **pwd** command (**print working directory**) writes the full pathname of the current working directory to the standard output. The command is a shell builtin in most **Unix** shells such as Bourne shell, ash, bash, ksh, and zsh.

[bshaban@snowy-sg1 tute4]$ echo $PWD

/vlsci/SG0009/bshaban/tute4

PUSHD

[bshaban@snowy-sg1 tute4]$ pushd $PWD

~/tute4 ~/tute4

POPD

[bshaban@snowy-sg1 SG0009]$ popd

~/tute4

grep

Basic example

Grep “literal string” example

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep "We have been unable to ascertain the date when young Lamarck" \*

20556.txt.utf-8:[8] We have been unable to ascertain the date when young Lamarck entered

20556.txt.Less specific example

bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep "this" \* | less

????

20556.txt.utf

Case insensitive search

bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -I “string” FILE

Regular Expressions

bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep REGEX filename

E.g. [bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -i "^Time" \*

### Checking for full words, not for sub-strings using grep -w

If you want to search for a word, and to avoid it to match the substrings use -w option. Just doing out a normal search will show out all the lines.

 [bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -i "is" \* | less

Display lines after match

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -A 5 "International" 20556.txt.utf-8

By Karl Semper. The International Scientific Series. New York, 1881.

[240] \_Organic Evolution as the Result of the Inheritance of Acquired

Characters, according to the Laws of Organic Growth.\_ Translated by

J. T. Cunningham, 1890.

--

International donations are gratefully accepted, but we cannot make

any statements concerning tax treatment of donations received from

outside the United States. U.S. laws alone swamp our small staff.

Please check the Project Gutenberg Web pages for current donation

methods and addresses. Donations are accepted in a number of other

To do the same to show a number of lines before the match use the B option

Invert Match

To perform an invert match i.e. show lines which don’t contain a match use the -v option.

e.g. will show all lines that don’t have the word “is” contained within it.

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -v "is" \* | less

Count the number of lines that match a pattern. (-c)

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -c "International" \*

20556.txt.utf-8:2

33660.txt.utf-8:1

45888-0.txt:2

50598.txt.utf-8:1

676.txt.utf-8:0

Show number of line when performing a grep match

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ grep -n "International" 20556.txt.utf-8

13794:By Karl Semper. The International Scientific Series. New York, 1881.

16074:International donations are gratefully accepted, but we cannot make

tar

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ tar -cvf backup.tar 20556.txt.utf-8

20556.txt.utf-8

1. c – Creates a new .tar archive file.
2. v – Verbosely show the .tar file progress.
3. f – File name type of the archive file.

Untar a file

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ tar -xvf backup.tar

20556.txt.utf-8

List contents

[bshaban@snowy-sg1 bunch\_of\_text\_files]$ tar -tvf backup.tar

-rw-r--r-- bshaban/SG0009 849583 2017-10-24 10:43 20556.txt.utf-8

Tar a directory

[bshaban@snowy-sg1 unzip]$ tar -xvf directory.tar

./bunch\_of\_text\_files/

./bunch\_of\_text\_files/20556.txt.utf-8

./bunch\_of\_text\_files/backup.tar

./bunch\_of\_text\_files/50598.txt.utf-8

./bunch\_of\_text\_files/676.txt.utf-8

./bunch\_of\_text\_files/33660.txt.utf-8

./bunch\_of\_text\_files/45888-0.txt

./bunch\_of\_text\_files/z

**vimdiff:** Opens two files in vim and compares them

 [bshaban@snowy-sg1 vimdiff]$ vimdiff vd1.txt vd2.txt

2 files to edit