

# Linux Command Line

Gowtham

Director of Research Computing, IT

EERC B39 · [g@mtu.edu](mailto:g@mtu.edu) · <http://hpc.mtu.edu>

- \* Intel Xeon X5675 3.07 GHz
- \* 24 CPU cores
- \* 96 GB RAM
- \* Red Hat Enterprise Linux 6.5
- \* Accessible from anywhere via SSH using a Terminal

#### Command

```
ssh -Y john@colossus.it.mtu.edu  
ssh -Y john@guardian.it.mtu.edu
```

Replace **john** with your Michigan ISO username.

These two machines can be used for light to medium weight computations by anyone that belongs to Michigan Tech.

Linux is a Unix-like and mostly POSIX-compliant computer operating system assembled under the model of, and a prime example for concept and practice of, free and open source software development and distribution.

The underlying source code may be used, modified, and distributed – commercially or non-commercially – by anyone under licenses such as the GNU General Public License.

Linux OS is usually packaged in a format known as a *distribution*.

# The home directory

## What is it?

A file system directory in a multi-user operating system that contains files/folders – including configuration files for various applications – for a specified user in the system. The exact location and naming convention used depends on the operating system as well as institutional policies/procedures.

## Why use it?

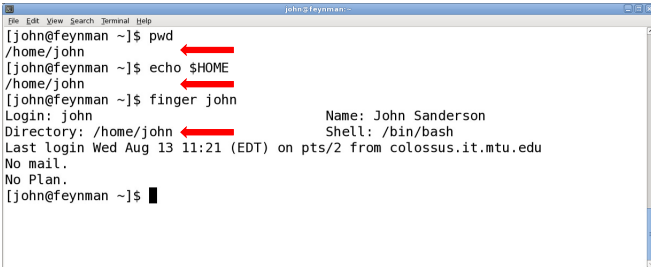
Contents of a user's home directory can be protected by file system permissions. Separating user data from system-wide data makes a backup scheme easier. Cracks (viruses, worms, etc.) running under the ownership of a user will usually be not able to affect system-wide data or that of other users.

# Where is the home directory?

\* Open a Terminal and type one of the following commands

\* `echo $HOME`

\* `finger john`

A terminal window titled 'john@feynman: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[john@feynman ~]$ pwd
/home/john
[john@feynman ~]$ echo $HOME
/home/john
[john@feynman ~]$ finger john
Login: john                                Name: John Sanderson
Directory: /home/john                      Shell: /bin/bash
Last login Wed Aug 13 11:21 (EDT) on pts/2 from colossus.it.mtu.edu
No mail.
No Plan.
[john@feynman ~]$
```

Red arrows point to the output of the 'pwd' command, the '\$HOME' variable output, and the 'Directory:' field in the 'finger' output.

Replace **john** with your Michigan ISO username OR appropriate username on personal machines.

`finger` command may not be available. Use the alternate command or contact administrators.

Typing `cd $HOME` or `cd ~/` (or just `cd`) from anywhere in the file system will bring the user back to the home directory.

# The shell

## What is it?

Although most users think of the shell as an interactive command interpreter, it is really a programming language in which each statement runs as a command. Because it must satisfy both the interactive and programming aspects of command execution, it is a strange language, shaped as much by history as by design.

– Brian Kernighan and Robert Pike

## Why use it?

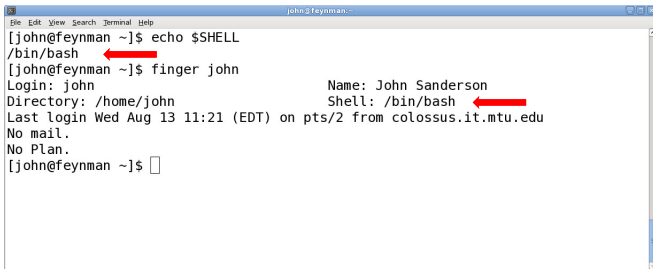
While using the GUI seems easier, the often repeated shell commands – which seamlessly interface with a plethora of other utilities – can be saved as a script or a function. This not only saves time and effort, and prevents errors, but also to naturally extends the system's capability.

# Which shell?

\* Open a Terminal and type one of the following commands

\* `echo $SHELL`

\* `finger john`

A terminal window titled 'john@feynman:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:  
[john@feynman ~]\$ echo \$SHELL  
/bin/bash  
[john@feynman ~]\$ finger john  
Login: john  
Directory: /home/john  
Last login Wed Aug 13 11:21 (EDT) on pts/2 from colossus.it.mtu.edu  
No mail.  
No Plan.  
[john@feynman ~]\$  
The output of the 'finger john' command is displayed on the right side of the terminal window:  
Name: John Sanderson  
Shell: /bin/bash

Replace **john** with your Michigan ISO username OR appropriate username on personal machines.

`finger` command may not be available. Use the alternate command or contact administrators.

`/bin/tcsh` is the default shell in most campus computers. Request IT to have it changed to `/bin/bash`.

# Login shell vs Interactive shell

## Login shell

A BASH shell that is initiated when one of the following commands is executed:

```
sudo su -  
bash --login  
ssh some-user@some-host
```

When BASH is invoked as a login shell, the following files are executed in the following order: /etc/profile, \$HOME/.bash\_profile, \$HOME/.bash\_login, \$HOME/.profile

Refer to `man bash` for more information.

Although `$HOME/.bashrc` is not explicitly listed, it is run by `$HOME/.bash_profile` in most linux distributions.



# Login shell vs Interactive shell

## Interactive shell

A BASH shell that is initiated when one of the following commands is executed:

```
sudo su
```

```
bash
```

```
ssh some-user@some-host some-command
```


When BASH is invoked as an interactive shell, only the `$HOME/.bashrc` is executed.

Refer to `man bash` for more information.

In most linux distributions, `$HOME/.bashrc` runs `/etc/bashrc`.

# Customizing the shell

- \* Open a Terminal
- \* Edit `$HOME/.bashrc` using `vi` (or other) editor as shown below
- \* Save and close the file



The screenshot shows a terminal window titled "john@feynman:~". The window contains the contents of the `.bashrc` file, which is being edited using the `vi` editor. The file content includes comments and code for sourcing global definitions and user-specific aliases. The status bar at the bottom indicates the file is 13 lines long and 174 characters wide, and that 2,0-1 lines are selected.

```
john@feynman:~  
File Edit View Search Terminal Help  
# .bashrc  
# Source global definitions  
if [ -f /etc/bashrc ]  
then  
    . /etc/bashrc  
fi  
# User specific aliases and functions  
if [ -f ~/.bash_$USER ]  
then  
    . ~/.bash_$USER  
fi  
~  
~  
".bashrc" 13L, 174C written 2,0-1 All
```

# Customizing the shell


- \* Open a Terminal
- \* Create/Open `$HOME/.bash_$USER` using `vi` (or other) editor
- \* Add necessary customizations (aliases, variables, functions, etc.)
- \* Save and close the file
- \* Run the command `. $HOME/.bashrc`

## Reserved shell variables

EDITOR, HOME, HOSTNAME, IFS, LD\_LIBRARY\_PATH, LOGNAME, MACHTYPE, MANPATH, OLDPWD, OSTYPE, PATH, PPID, PS1, PS2, PS3, PS4, PWD, SHELL, TMOUT, TZ, UID, USER

Run `env` to get a complete list of shell variables already in use.  
Carefully redefine a variable or extend its definition only if there is a need to do so.

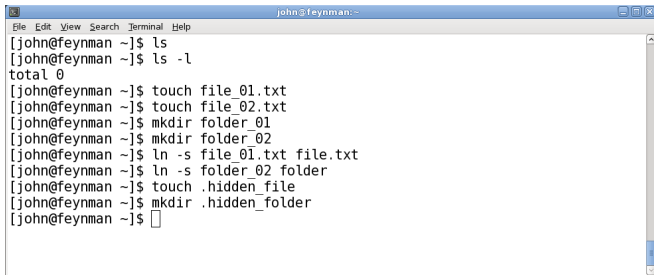
# Customizing the shell

A terminal window titled 'john@feynman:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal displays the contents of the '.bashrc' file. The file contains several 'export' statements for environment variables like HISTCONTROL, HISTSIZE, HISTFILESIZE, HISTTIMEFORMAT, TMOUT, PATH, LD\_LIBRARY\_PATH, and MANPATH. It also contains 'alias' definitions for 'cl', 'll', 'lr', 'la', 'colossus', and 'guardian'. At the bottom, it shows '~' and a status line: '".bash\_john" 23L, 676C written' with '3,0-1' and 'All' on the right.

```
john@feynman:~  
File Edit View Search Terminal Help  
# .bashrc  
# User specific aliases, variables, functions, environment and startup programs  
  
export HISTCONTROL=ignoreboth  
export HISTSIZE=100000  
export HISTFILESIZE=100000  
export HISTTIMEFORMAT="%b %d %H:%M:%S "  
export TMOUT=360000  
  
export PATH="/bin:/usr/bin:/sbin:/usr/sbin:/usr/kerberos/bin"  
export PATH="$PATH:/usr/local/bin:$HOME/bin"  
export LD_LIBRARY_PATH="/lib64:/usr/lib64:/usr/kerberos/lib64:$HOME/lib"  
export MANPATH="/usr/share/man:$HOME/man"  
  
export COLOSSUS="colossus.it.mtu.edu"  
export GUARDIAN="guardian.it.mtu.edu"  
  
alias cl="clear"  
alias ll="ls -lh"  
alias lr="ls -ltrh"  
alias la="ls -a"  
alias colossus="ssh -Y $COLOSSUS"  
alias guardian="ssh -Y $GUARDIAN"  
~  
".bash_john" 23L, 676C written 3,0-1 All
```

Open a Terminal and create `$HOME/bin`, `$HOME/lib`, `$HOME/man` directories using `mkdir` command.  
Save and close `$HOME/.bashrc` after making the changes, and run `. $HOME/.bashrc` for the changes to take effect.  
OS looks for commands, libraries and manual pages in `PATH`, `LD_LIBRARY_PATH` and `MANPATH` respectively.

# Files, folders, symbolic links

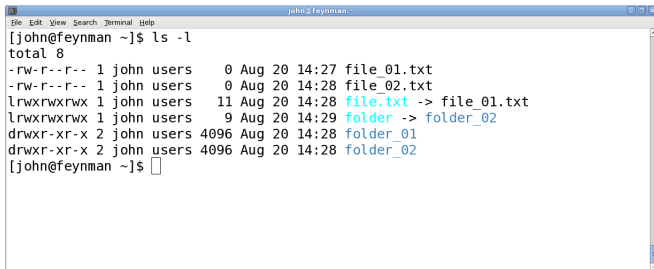
A screenshot of a terminal window titled 'john@feynman:~'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows a series of commands and their outputs: 'ls' returns '[john@feynman ~]\$ ls'; 'ls -l' returns '[john@feynman ~]\$ ls -l' and 'total 0'; 'touch file\_01.txt' returns '[john@feynman ~]\$ touch file\_01.txt'; 'touch file\_02.txt' returns '[john@feynman ~]\$ touch file\_02.txt'; 'mkdir folder\_01' returns '[john@feynman ~]\$ mkdir folder\_01'; 'mkdir folder\_02' returns '[john@feynman ~]\$ mkdir folder\_02'; 'ln -s file\_01.txt file.txt' returns '[john@feynman ~]\$ ln -s file\_01.txt file.txt'; 'ln -s folder\_02 folder' returns '[john@feynman ~]\$ ln -s folder\_02 folder'; 'touch .hidden\_file' returns '[john@feynman ~]\$ touch .hidden\_file'; 'mkdir .hidden\_folder' returns '[john@feynman ~]\$ mkdir .hidden\_folder'; and the prompt '[john@feynman ~]\$' is followed by a cursor.

```
john@feynman:~  
File Edit View Search Terminal Help  
[john@feynman ~]$ ls  
[john@feynman ~]$ ls -l  
total 0  
[john@feynman ~]$ touch file_01.txt  
[john@feynman ~]$ touch file_02.txt  
[john@feynman ~]$ mkdir folder_01  
[john@feynman ~]$ mkdir folder_02  
[john@feynman ~]$ ln -s file_01.txt file.txt  
[john@feynman ~]$ ln -s folder_02 folder  
[john@feynman ~]$ touch .hidden_file  
[john@feynman ~]$ mkdir .hidden_folder  
[john@feynman ~]$
```

Open a Terminal and type the above commands.

Entities that start with a `.` are hidden, and don't appear in `ls` or `ls -l`. Use `man ls` to learn how to list all entities.

# Ownership and permission



```
john@feynman:~$ ls -l
total 8
-rw-r--r-- 1 john users 0 Aug 20 14:27 file_01.txt
-rw-r--r-- 1 john users 0 Aug 20 14:28 file_02.txt
lrwxrwxrwx 1 john users 11 Aug 20 14:28 file.txt -> file_01.txt
lrwxrwxrwx 1 john users 9 Aug 20 14:29 folder -> folder_02
drwxr-xr-x 2 john users 4096 Aug 20 14:28 folder_01
drwxr-xr-x 2 john users 4096 Aug 20 14:28 folder_02
john@feynman ~$
```

- \* Entity type: normal file (-), directory (d), link (-), socket (s)
- \* Ownership levels: user (u), group (g), others (o)
- \* Permission levels: read (4, r), write (2, w), execute (1, x)

Open a Terminal and type `ls -l`.

Permission level values add up at each ownership level.

Ownership and permission can be changed using `chown` and `chmod` commands respectively.

`file_01.txt` and `file_02.txt` have 644, `folder_01` and `folder_02` have 755, and `folder` and `file.txt` have 777.

# Ownership and permission

## Using alphabet approach

```
chmod u=rwx file_01.txt
chmod g+rw,o-rwx file_02.txt
chmod g+x,o-x folder_01
chmod u-x folder_02
```

## Using number approach

```
chmod 744 file_01.txt
chmod 660 file_02.txt
chmod 754 folder_01
chmod 655 folder_02
```

Open a Terminal and type the above commands. Run `ls -l` after each command to observe the changes. Reset the permission to original values after each approach by using `chmod 644 file_01.txt file_02.txt` and `chmod 755 folder_01 folder_02`.

# Commands

## Basic

`cat`, `cd`, `clear`, `cp`, `date`, `echo`, `finger`, `grep`, `head`, `history`, `less`,  
`ls`, `man`, `mkdir`, `more`, `mv`, `pwd`, `rm`, `rmdir`, `tail`, `touch`, `vim`

## Intermediate

`awk`, `basename`, `bc`, `bzip2`, `chmod`, `chown`, `comm`, `crontab`, `cut`, `df`,  
`diff`, `du`, `env`, `expect`, `expr`, `file`, `find`, `free`, `gzip`, `hostname`, `id`,  
`kill`, `killall`, `ln`, `locate`, `paste`, `ping`, `ps`, `rsync`, `scp`, `sdiff`,  
`sed`, `seq`, `sleep`, `sort`, `ssh`, `tar`, `time`, `top`, `tr`, `ulimit`, `uniq`, `wc`

## Advanced

`chgrp`, `groupadd`, `groupmod`, `groupdel`, `ifconfig`, `mount`, `passwd`,  
`poweroff`, `reboot`, `su`, `uptime`, `umount`, `useradd`, `usermod`, `userdel`

To learn more about a command (e.g., `mkdir`), use `man mkdir`.



# Piping

The act of treating the output of one command as the input for a subsequent command. | character represents the pipe.

Prevents creation and keeping track of temporary files to store the output of each command, and is a segue to elegant shell scripting.

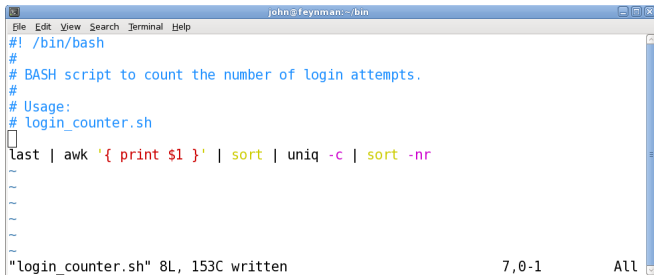
## Examples

```
du | sort -nr  
ls -l | tail -n +2  
ps aux | grep $USER  
ls -l | sed '1d' | wc -l  
last | awk '{ print $1 }' | sort | uniq -c | sort -nr  
seq 1 1 100 | awk '{ sum += $1 } END { print sum }'
```

Open a Terminal and type the above commands. Refer to [man](#) pages to learn what each command (and its option) is doing.

# Shell script

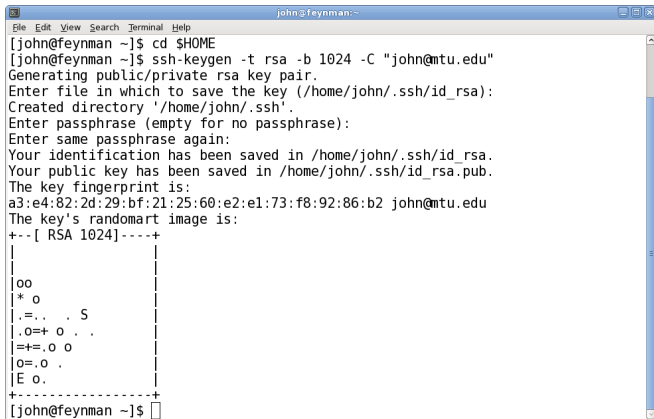
A set of (piped) commands, to accomplish a given task, saved in a file for easier execution – helps automate the work flow, reduce chances of errors, and make time for more productive activities.



```
john@feynman:~/bin
File Edit View Search Terminal Help
#!/bin/bash
#
# BASH script to count the number of login attempts.
#
# Usage:
# login_counter.sh
last | awk '{ print $1 }' | sort | uniq -c | sort -nr
~
~
~
~
~
"login_counter.sh" 8L, 153C written          7,0-1      All
```

Save all shell scripts in \$HOME/bin folder. A shell script requires 755 (or at least 700) permission to run.

# SSH keys



```
john@feynman:~  
File Edit View Search Terminal Help  
[john@feynman ~]$ cd $HOME  
[john@feynman ~]$ ssh-keygen -t rsa -b 1024 -C "john@mtu.edu"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/john/.ssh/id_rsa):  
Created directory '/home/john/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/john/.ssh/id_rsa.  
Your public key has been saved in /home/john/.ssh/id_rsa.pub.  
The key fingerprint is:  
a3:e4:82:2d:29:bf:21:25:60:e2:e1:73:f8:92:86:b2 john@mtu.edu  
The key's randomart image is:  
+--[ RSA 1024 ]-----+  
|  
| oo  
| * o  
| ,=. . . S  
|.o=+ o . .  
|+=+.o o  
|o=.o .  
|E o.  
+-----+  
[john@feynman ~]$
```

Replace **john** with your Michigan ISO username.

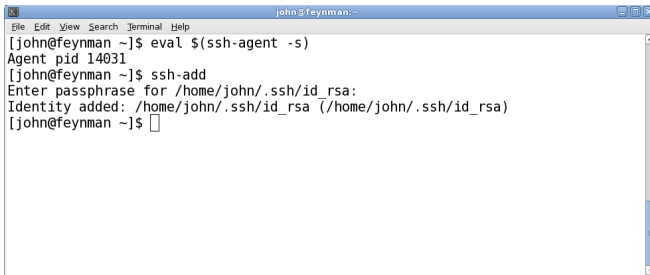
Accept the default location for `$HOME/.ssh/id_rsa`, and set a passphrase (different from Michigan Tech ISO password).

When completed successfully, it will result in two files in `$HOME/.ssh` folder: `id_rsa` and `id_rsa.pub`.

Do not ever share/distribute `id_rsa`.

Copy the contents of `id_rsa.pub` into `$HOME/.ssh/authorized_keys` of a remote server.

# SSH keys

A terminal window titled 'john@feynman:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[john@feynman ~]$ eval $(ssh-agent -s)
Agent pid 14031
[john@feynman ~]$ ssh-add
Enter passphrase for /home/john/.ssh/id_rsa:
Identity added: /home/john/.ssh/id_rsa (/home/john/.ssh/id_rsa)
[john@feynman ~]$
```

Replace **john** with your Michigan ISO username.

`eval $(ssh-agent -s)` followed by `ssh-add` may only need to be run once per session (or day). Check the system settings. `ssh`, `scp`, and `rsync` over `ssh` from local machine to remote server will no longer prompt for the password.

## Additional references

---

- \* [The Linux Command Line](#)
- \* [The Command Line Crash Course](#)
- \* [BASH Guide for Beginners](#)
- \* [BASH Scripting/Programming: Introduction](#) | [Advanced](#)
- \* [Vi editor: Interactive tutorial](#) | [Reference](#)
- \* [Tip of the Week: Michigan Tech HPC](#) | [GitHub repository](#)
- \* [Twitter: @UNIXToolTip](#) | [@Linux\\_Tips](#) | [@CLIMagic](#) | [@RegExTip](#)

A really good and effective way to learn Linux command line quickly is imposing upon yourself to use it for accomplishing as many, if not all, tasks every single day until it starts becoming second nature.

---

# Need help?

<https://servicedesk.mtu.edu>

Request Catalog → Research Computing → Computing and/or Visualization

Subcategory: Work Flow Design and Development (Scripting and Programming)