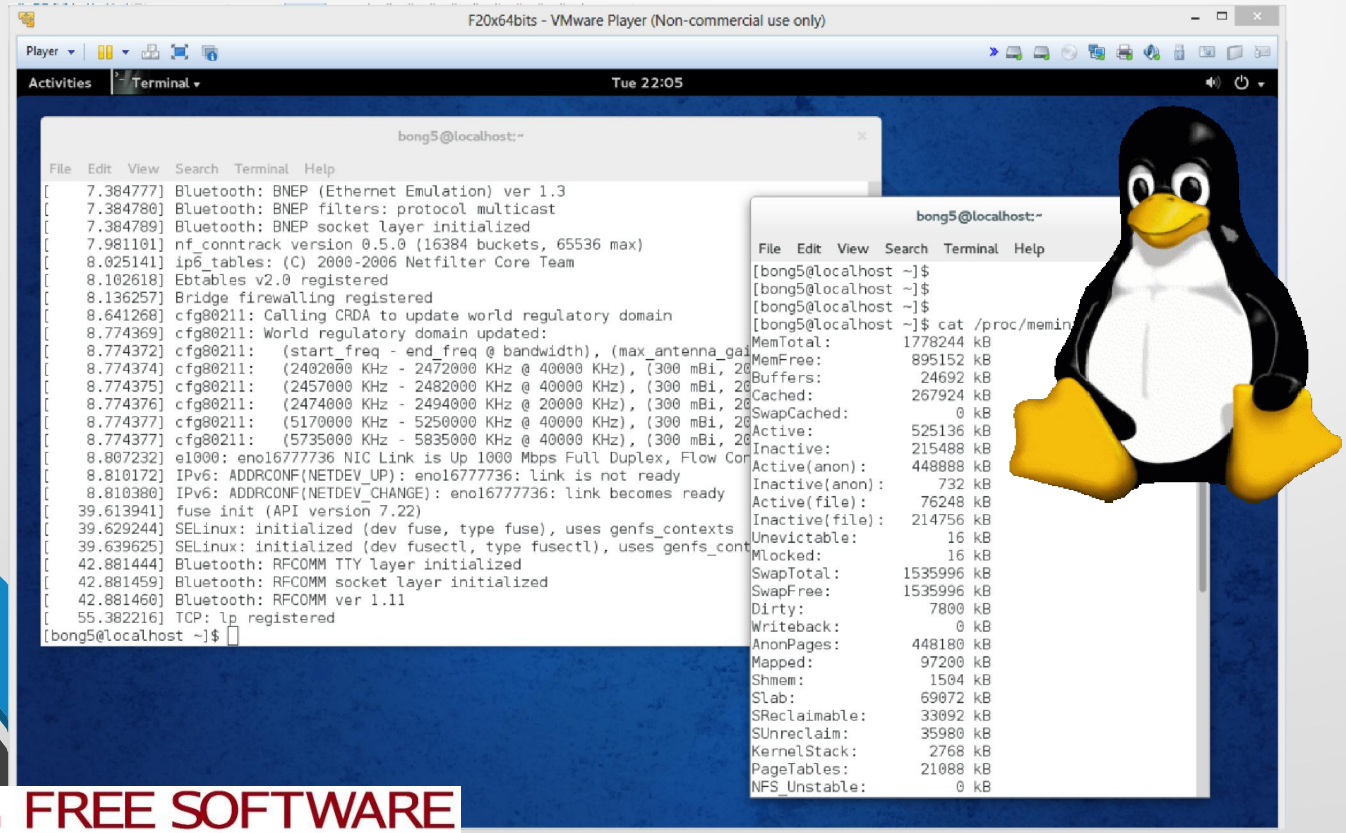
Linux Basic - 101
Let's start our OSS journey

By Ong Boon Leong



FREE SOFTWARE FOUNDATION

Prepared by Mr Ong Boon Leong 2014

Personal Biography

Ong Boon Leong

Industrial Experience:

- Embedded Systems (network processor, low-power system)
- Yocto Project Contributor
- Linux Kernel & Device Driver
- Assembly language, C/C++, Java, Database
- Windows Embedded Compact – OS & Device Driver
- Current working for Intel Malaysia – 12 years.

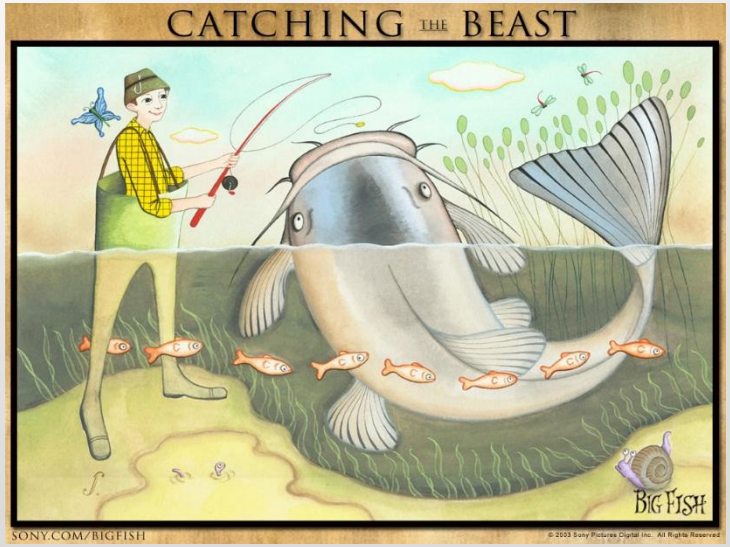
Education:

- MSc (Distinction) Signal Processing & Telecommunication, Imperial College London
- BEng (Hons) EEE, University Tenaga Nasional

Disclaimer: I work for Intel but I don't speak for them.

Expectation on You

- I expect basic class to be **less spoon-feed** than it should be …
- No Silly Question > Doing Silly Thing
- I teach you the **basic about fishing** … you should go on learning how to **catch a BIG FISH …**



Agenda

- Installing Linux on Windows [30-min]
- Familiarizing with Linux Environment [20-min]
  - shell environment
  - proxy settings
  - Software package management introduction
- Break [10-min]
- Basic Shell Commands [20-min]
- Vi Editor Introduction [10-min]
- Hand-on: Write a shell script [20-min]

Installing Linux on Windows [30min]

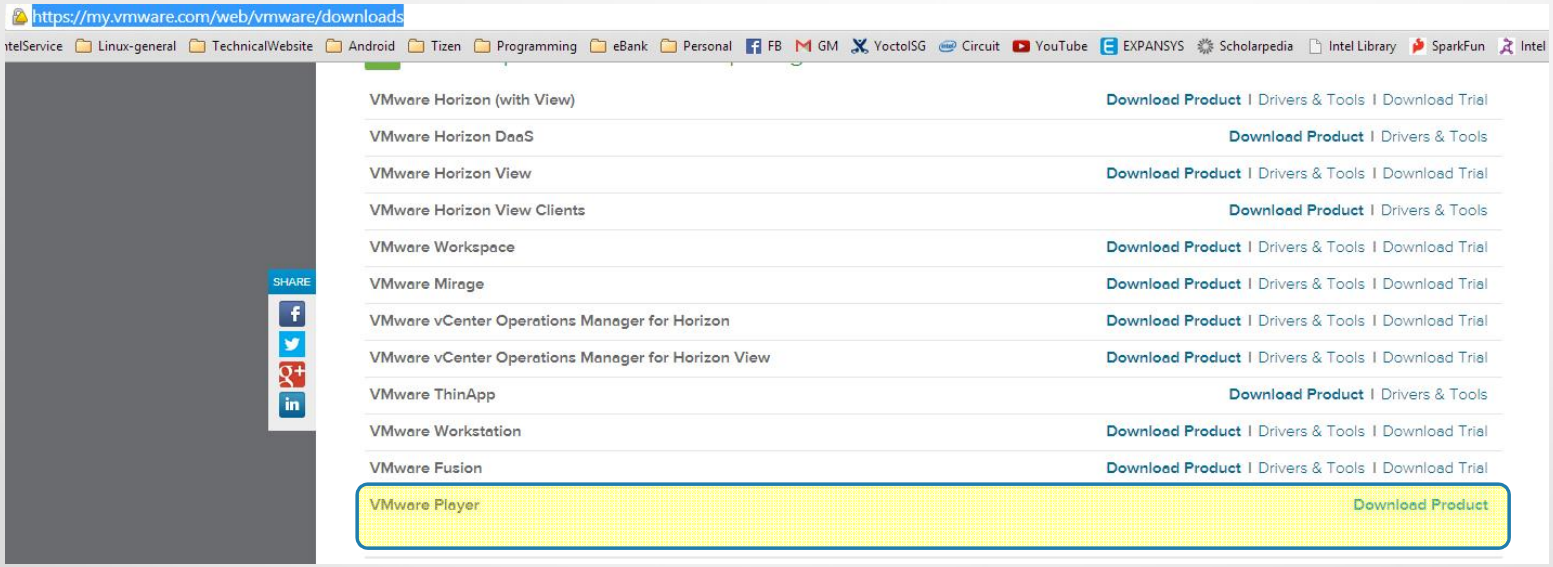
Installing Linux on Windows – 0/7

1. Download virtualization software (e.g. Virtual Box, **VM Player**, etc …)
2. Download Linux OS ISO (e.g. Fedora, **Ubuntu**, etc)
3. Make sure **BIOS** enables **VT-x & VT-d** (if available)
4. Create virtual machine (VM) and start installing ISO
5. Making sure **networking** is enabled
6. Let ISO installed on virtual disk and reboot
7. Installing **Vmware** Tools
8. Enable host (Windows) folder sharing with Linux

Installing Linux on Windows – 1/7

1) Download virtualization software (e.g. Virtual Box, **VM Player**, etc …)

- VM Player: https://my.vmware.com/web/vmware/downloads



You may choose Virtual Box if that is your preference … I prefer VMWare personally

Installing Linux on Windows – 2/7

2) Download Linux OS ISO (e.g. Fedora, Ubuntu, etc)

- Ubuntu: http://releases.ubuntu.com/
- Fedora: http://archive.fedoraproject.org/pub/fedora/linux/releases/

If you have laptop or desktop that will run Linux natively and be used as Yocto Project build-machine, you should consider the supported Linux distro defined in http://git.yoctoproject.org/clean/cgit.cgi/poky/tree/meta-yocto/conf/distro/poky.conf?h=<branch-code-name>

```
SANITY_TESTED_DISTROS ?= " \
            Poky-1.4 \n \
            Poky-1.5 \n \
            Poky-1.6 \n \
            Ubuntu-12.04 \n \
            Ubuntu-13.10 \n \
            Ubuntu-14.04 \n \
            Fedora-19 \n \
            Fedora-20 \n \
            CentOS-6.4 \n \
            CentOS-6.5 \n \
            Debian-7.0 \n \
            Debian-7.1 \n \
            Debian-7.2 \n \
            Debian-7.3 \n \
            Debian-7.4 \n \
            SUSE-LINUX-12.2 \n \
            openSUSE-project-12.3 \n \
            openSUSE-project-13.1 \n \
            "
```

Sanity tested distro for Yocto v1.6.1 (daisy)

See http://git.yoctoproject.org/clean/cgit.cgi/poky/tree/meta-yocto/conf/distro/poky.conf?h=daisy

Normally, I will pick non-bleeding edge version due to stability.

Installing Linux on Windows – 3/7

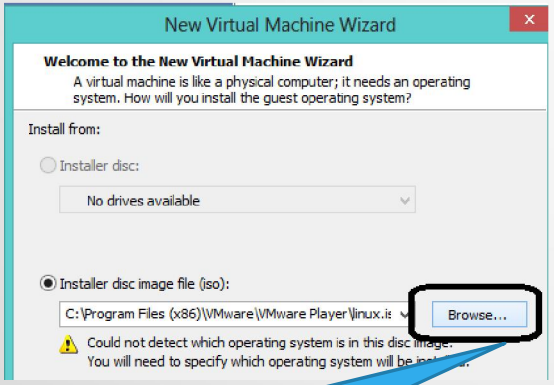
3) Make sure **BIOS** enables **VT-x & VT-d** (if available)

- To ensure your virtualization software uses Virtualization Technology offered by your x86 laptop…
- Note: Different machine comes with different BIOS, you either ESC, DEL or special key to pause it.
- Note: Your machine laptop may not have option to enable VT-x or VT-d, either enabled by default or not supported (too bad).
- This step can be done at the end after you have installed your Linux OS

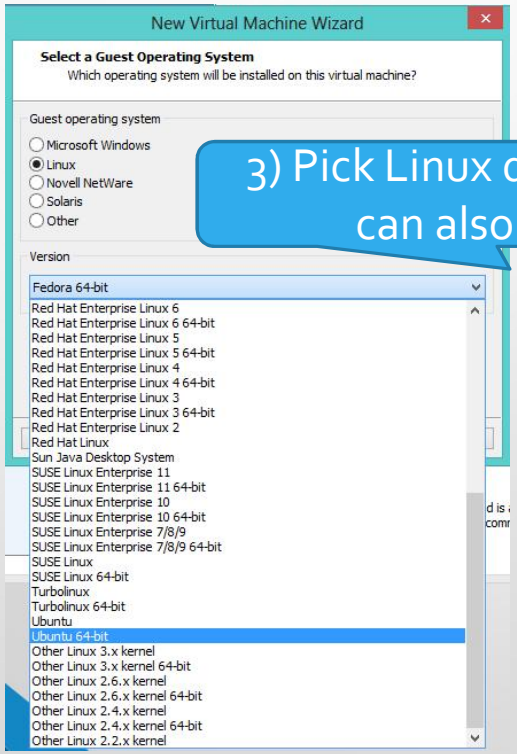
Installing Linux on Windows – 4a/7

4) Create virtual machine (VM) and start installing ISO

1) Create new VM



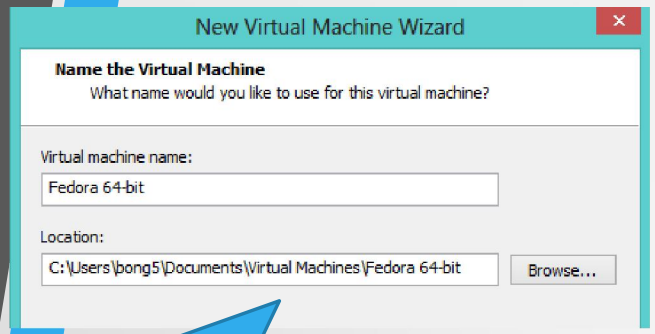
2) Select Linux ISO



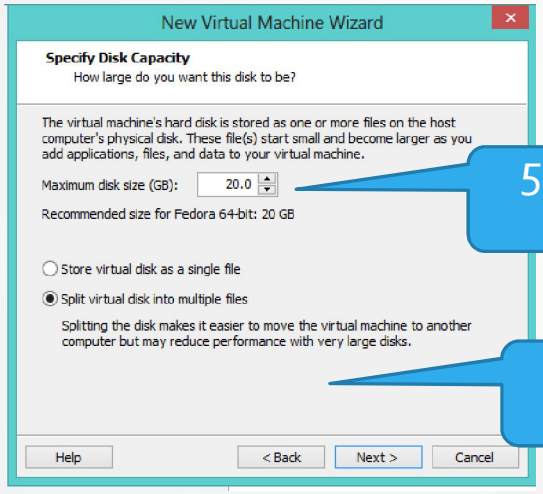
3) Pick Linux distro or VMPlayer can also auto-detect

Installing Linux on Windows – 4b/7

4) Create virtual machine (VM) and start installing ISO

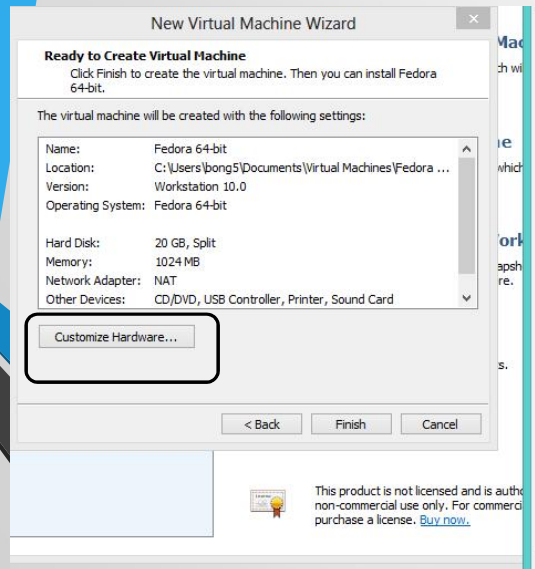


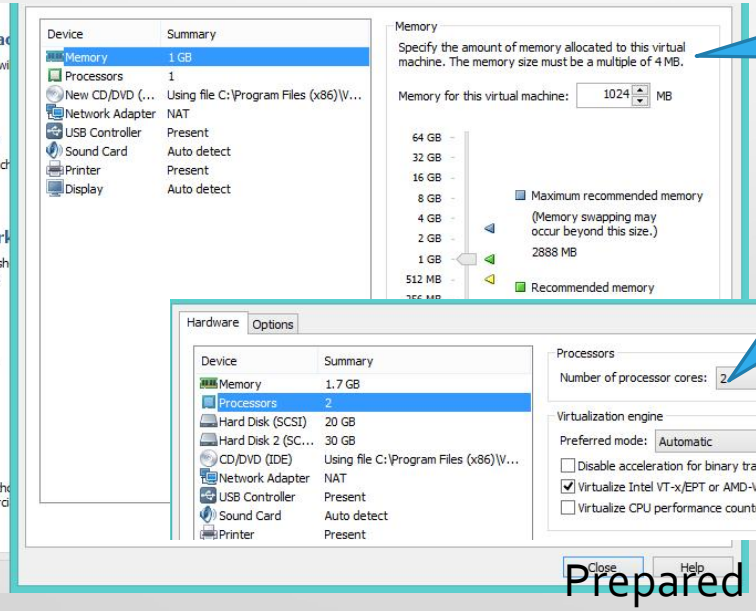
4) Name your VM and note the location



5) May want to have bigger virtual disk if permits

6) Split disk for better scalability across hosts



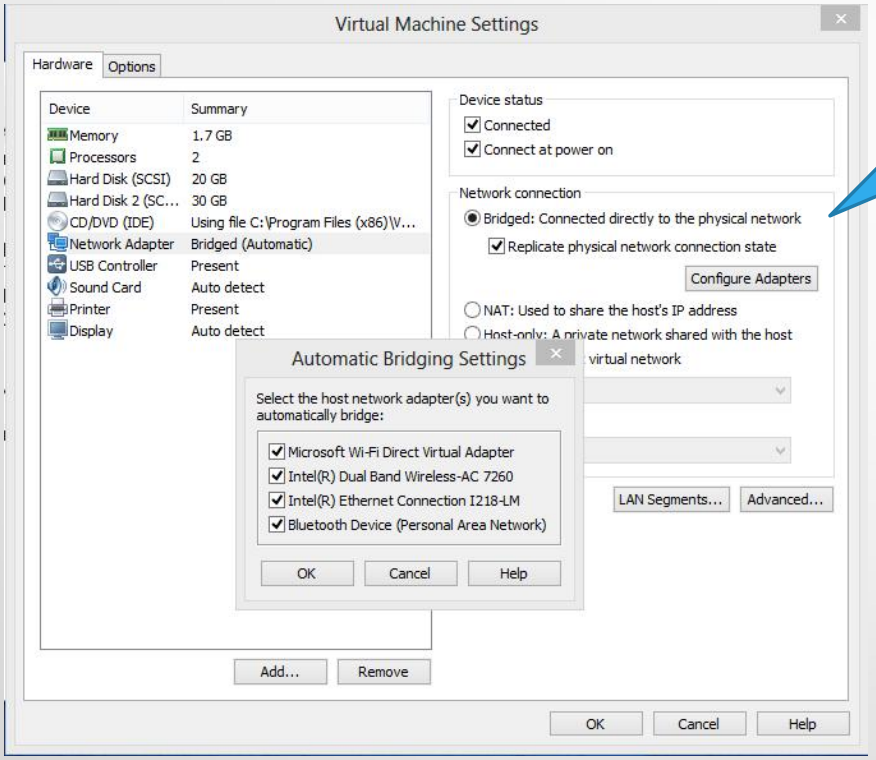


7) Allocate memory gentle to your host OS

8) Allocate processor & enable VT-x

Installing Linux on Windows – 5/7

5) Making sure networking is enabled. Note: you can turn it on later if you don't want your Linus OS to auto-download package during installation.

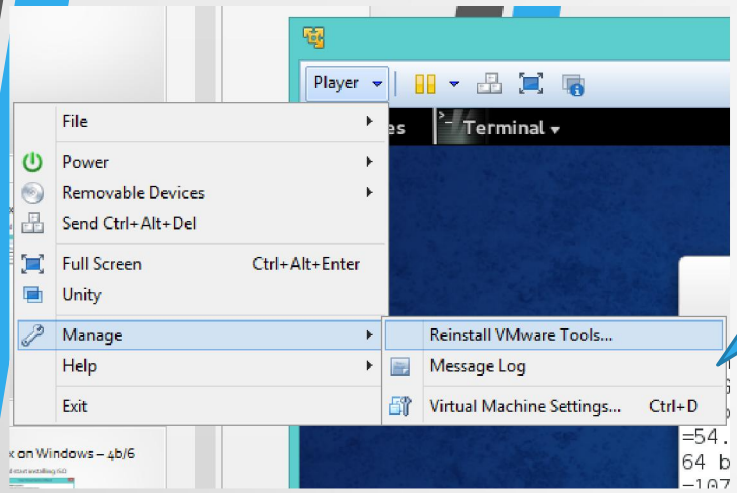


9) Bridged + Replicate network connection state.

6) Let ISO boot-up and installed on virtual disk allocated earlier. Then, reboot

Installing Linux on Windows – 6a/7

7) Installing **Vmware** Tools (for better performance) and access host file-system



10) Mount Vmware tools

```
[bong5@localhost VMware Tools]$ ls
manifest.txt      VMwareTools-9.6.2-1688356.tar.gz  vmware-tools-upgrader-64
run_upgrader.sh   vmware-tools-upgrader-32
[bong5@localhost VMware Tools]$ pwd
/run/media/bong5/VMware Tools
[bong5@localhost VMware Tools]$
```

11) $ cp VMwareTools-xxx.tar.gz ~/Downloads

Installing Linux on Windows – 6b/7

7) Installing **Vmware** Tools (for better performance) and access host file-system

$ cd ~/Downloads
$ tar –zxf VMwareTools-<version>.tar.gz

note down Linux kernel version
$ uname –a
$ rpm –qa | grep kernel-

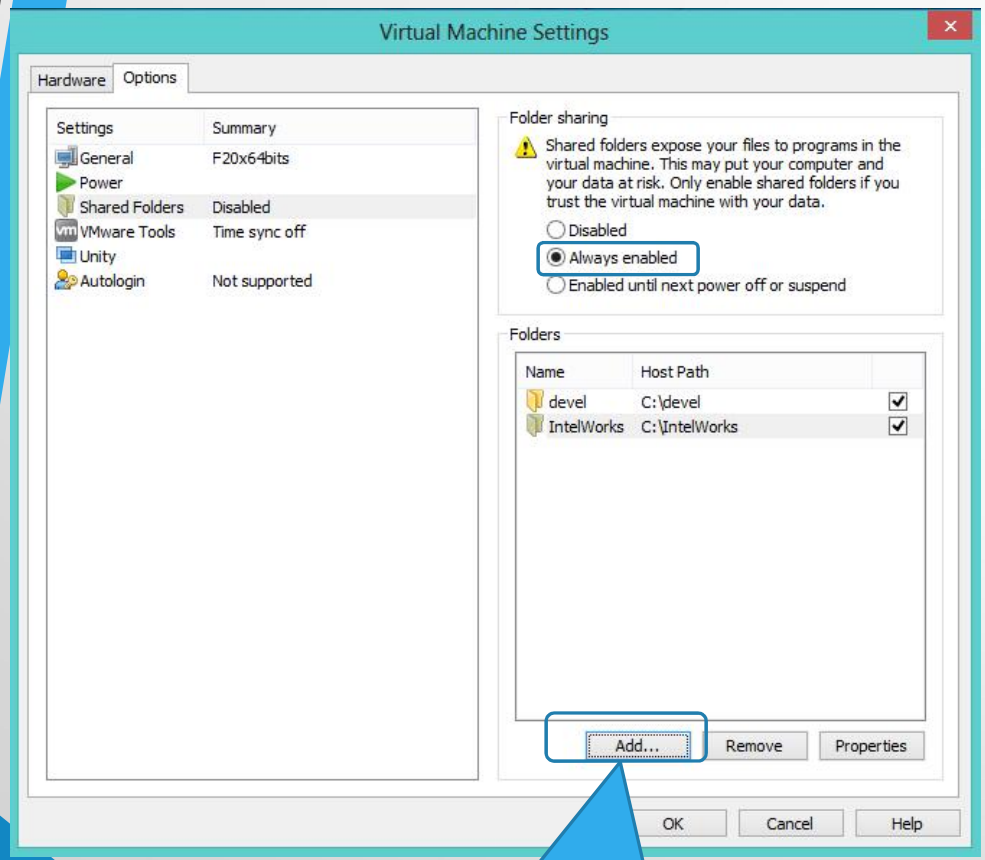
install some useful tool-chain before installing Vmware tools
*$ sudo yum install **gcc make perl***
$ sudo yum install kernel-devel-<3.11.10-301>.fc20.x86_64
Note: yum install kernel-devel gives you the latest version which may not be the same as the run-time version. Enter the run-time version in <version> tag.

# start installing VMware tool & follow-through a series of question by picking default option
$ sudo ./vmware-install.pl

# if hit into open-vm-tools issue, simply remove the package and re-execute *vmware-install.pl*
$ sudo yum remove open-vm-tools

Installing Linux on Windows – 7/7

8) Enable host (Windows) folder sharing with Linux



12) Add host directory

```
[bong5@localhost hgfs]$ pwd
/mnt/hgfs
[bong5@localhost hgfs]$ ls
devel  IntelWorks
[bong5@localhost hgfs]$
```

13) Added host directory appears under /mnt/hgfs

Familiarizing with Linux Environment [20min]

- shell environment settings –
- sudo su != sudoku -
-network proxy settings -
- Software package management introduction -

Shell Environment Settings – 1/3

- Shell environment settings decide couple of things e.g.
  - $ env #display all system variable
  - How command (executable) is searched → PATH
  - What command language interpreter → SHELL
  - Who am i → USER
  - Shell prompt → PS1
  - Many more …
- Generally, environment is setup through
- /home/<user>/.bashrc → user-account wide
- /etc/environment →system-wide

```
bong5@ubuntu:~$ cat /etc/environment
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games"
```

Shell Environment Settings – 2/3

- To introduce extra environment variable:
  - $ export THIS_MY_VAR="Hey Guys…"
- To show the value of newly exported variable
  - $ echo $THIS_MY_VAR
- To confirm it is part of environment variable
  - $ env | grep THIS_MY_VAR

```
bong5@ubuntu:~$ export THISMINE="hellow world"
bong5@ubuntu:~$ echo $THISMINE
hellow world
bong5@ubuntu:~$ env | grep THISMINE
THISMINE=hellow world
```

Shell Environment Settings – 3/3

- How about to remove a system variable?

```
bong5@ubuntu:~$ unset THISMINE
bong5@ubuntu:~$ echo $THISMINE

bong5@ubuntu:~$
bong5@ubuntu:~$
bong5@ubuntu:~$ env | grep THISMINE
```

- How about to add extra value to a system variable?

```
bong5@ubuntu:~$ export THISMINE="hellow world"
bong5@ubuntu:~$ echo $THISMINE
hellow world
bong5@ubuntu:~$ export THISMINE="$THISMINE from PSC"
bong5@ubuntu:~$ echo $THISMINE
hellow world from PSC
```

sudo su != sudoku - 1/2

- **Each account type** is assigned certain **access privilege** so that the dark(stupid)-side of us does not do something damaging … :
  - **root** account = administrative (superuser) account, the almighty creator (destroyer?!) of a system. Handle with care wisely !!!
  - user account = limited in system-level file access (read/write/delete)
- **sudo** gives normal user account a way to execute command as superuser.
  - Security policy decided under **/etc/sudoers** file (only accessed by superuser)
- **sudo su -** switch to superuser before we edit sudoers file
  - By default, for Ubuntu, you are not asked the password for superuser. If you want to set it, use passwd

sudo su != sudoku - 2/2

 # switch user to superuser
user $ sudo su
root $ gedit /etc/sudoers &
root $ su - <username>

```
bong5@ubuntu:~$ sudo su
root@ubuntu:/home/bong5# gedit /etc/sudoers &
[1] 9976
root@ubuntu:/home/bong5#
** (gedit:9976): WARNING **: Couldn't

root@ubuntu:/home/bong5#
root@ubuntu:/home/bong5#
root@ubuntu:/home/bong5#
root@ubuntu:/home/bong5# su - bong5
bong5@ubuntu:~$ pwd
/home/bong5
```

*sudoers (/etc) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

*sudoers

```
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/
sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
bong5   ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
```

I like to be in (superuser) control of my system.
But, System admin may not like this setting for other user.

Network Proxy Settings

- When you are using Linux within company firewall, you will(may) need to deal with network proxy.
- Network proxy is "middle-man" between you and Internet.
- Defined proxy under /etc/environment or ~/.bashrc

```
# Usually, it is small-case settings
export http_proxy=<ip-address>:<port>
export https_proxy=<ip-address>:<port>
export ftp_proxy=<ip-address>:<port>
export socks_server=<ip-address>:<port>
export rsync_proxy=<ip-address>:<port>
export no_proxy=localhost,localaddress,127.0.0.0/8,<locally assigned ip-address range>/8

# Cater ALL-CAP case for rare scenario
export HTTP_PROXY=$http_proxy
export HTTPS_PROXY=$https_proxy
export FTP_PROXY=$ftp_proxy
export SOCKS_SERVER=$socks_server
export ALL_PROXY=$all_proxy
export RSYNC_PROXY=$rsync_proxy
export NO_PROXY=$no_proxy
```

Software Package Management – 1/2

- Pre-packaged Linux distros (e.g. Fedora, Ubuntu) carry **core** & **essential** software packages (not all) due to image size.
- Software Package Management lets user to **extend** the capability of an installed OS as needed.
- Software Packages are hosted at remote server e.g:
  - http://packages.ubuntu.com/
  - http://archive.fedoraproject.org/pub/fedora/linux/releases/<version>/Everything/x86_64/os/Packages/
- We use software package manager to install packages:
  - Fedora : yum
  - Ubuntu: apt-get

Software Package Management – 2/2

	Fedora	Ubuntu
Get help	$ yum --help	$ apt-get --help $ apt-cache –help
Update package list	$ yum update	$ apt-get update
Search package	$ yum search <pkg>	$ apt-cache search -n <pkg>
Install package	$ yum install <pkg>	$ apt-get install <pkg>
Uninstall package	$ yum erase <pkg>	$ apt-get remove <pkg>
Download source	$ yum install yum-utils $ yumdownloader --source <pkg>	$ apt-get source <pkg>
Show package info	$ yum info <pkg>	$ apt-cache show <pkg>
List packages installed		$ apt-cache pkgnames

Break [10min]

Basic Shell Commands [30min]

Shell - Introduction

- shell/terminal commands/utilities are inside:
  - /bin – essential commands for single user mode, e.g. cat, ls , cp
  - /sbin – essential system commands, e.g. init, mount
  - /usr/bin – non-essential commands, e.g. git, gdb,
  - /usr/sbin – non-essential system commands, e.g. daemon
  - Other special path defined by $PATH
- Executed **directly from terminal** or **written on bash script** that is interpreted by command language interpreter (e.g. bash, csh, sh, zsh, tcsh, etc)

$whereis <cmd> displays cmd path

Access mode "+ax"

```
bong5@ubuntu:/usr/bin$ whereis cat
cat: /bin/cat /usr/share/man/man1/cat.1.gz
bong5@ubuntu:/usr/bin$ ls -al /bin/cat
-rwxr-xr-x 1 root root 47904 Mar 24  2014 /bin/cat
bong5@ubuntu:/usr/bin$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

PATH contains "/bin"

Shell – Navigating File System (1/3)

```
bong5@ubuntu:~$ pwd
/home/bong5
bong5@ubuntu:~$ cd devel/
bong5@ubuntu:~/devel$ ls -al
total 12
drwxrwxr-x  3 bong5 bong5 4096 Sep 22 07:06 .
drwxr-xr-x 17 bong5 bong5 4096 Sep 22 06:30 ..
drwxrwxr-x  2 bong5 bong5 4096 Sep 22 06:48 test
bong5@ubuntu:~/devel$
bong5@ubuntu:~/devel$
```

$ pwd = print working directory
$ ls = list current directory

```
bong5@ubuntu:~$ ls
Desktop  Documents  examples.desktop  Pictures  Templates
devel    Downloads  Music             Public    Videos
bong5@ubuntu:~$ ls devel/
test
bong5@ubuntu:~$ mkdir -p devel/junk1/junkapp1/usage1
bong5@ubuntu:~$ ls devel/junk1/
junkapp1
bong5@ubuntu:~$ ls devel/junk1/junkapp1/
usage1
```

$ mkdir = make directory
$ mkdir -p = make multi-level directory

```
bong5@ubuntu:~/devel$ rmdir junk1/junkapp1/usage1/
bong5@ubuntu:~/devel$ ls junk1/junkapp1/
bong5@ubuntu:~/devel$ ls
junk1  test
bong5@ubuntu:~/devel$ rm -rf junk1/
bong5@ubuntu:~/devel$ ls -al
total 12
drwxrwxr-x  3 bong5 bong5 4096 Sep 22 07:06 .
drwxr-xr-x 17 bong5 bong5 4096 Sep 22 06:30 ..
drwxrwxr-x  2 bong5 bong5 4096 Sep 22 06:48 test
```

$ rmdir = remove empty directory
$ rm -rf = remove recursively from named directory

Shell – Navigating File System (2/3)

```
bong5@ubuntu:~/devel/test$ ls
testapp  test.c  test.h
bong5@ubuntu:~/devel/test$ file test.c
test.c: C source, ASCII text
bong5@ubuntu:~/devel/test$ file test.h
test.h: ASCII text
bong5@ubuntu:~/devel/test$ file testapp
testapp: ELF 64-bit LSB  executable, x86-64, version 1 (SYSV), dynamically linke
2113740bda7f209aaa2a18edb8dd0a6280, not stripped
```

tree – listing file and structure

```
bong5@ubuntu:~/devel$ tree
.
└── test
    ├── testapp
    ├── test.c
    └── test.h
```

Not able to run tree? ☺
Software Package Manager ?

```
bong5@ubuntu:devel$ mv test3 test2  # renaming test3 to test2
bong5@ubuntu:devel$ ls
bash-example  junk  sort-test  test  test2  test4
bong5@ubuntu:devel$ ls test2
bong5@ubuntu:devel$ ls test
testapp  test.c  test.h
bong5@ubuntu:devel$ ls test4
file1.txt
bong5@ubuntu:devel$ mv test2 test4 junk # moving folders to junk folder
bong5@ubuntu:devel$ ls junk/
random.log  script.sh  test2  test4  test.c
bong5@ubuntu:devel$ ls
bash-example  junk  sort-test  test
```

$ mv <src> <dest> – is used to move or rename file/dir

Shell – Navigating File System (3/3)

```
bong5@ubuntu:~/devel$ mkdir test2
bong5@ubuntu:~/devel$ ls
test  test2
bong5@ubuntu:~/devel$ cd test2
bong5@ubuntu:~/devel/test2$ touch file1.txt file2.txt file3.txt
bong5@ubuntu:~/devel/test2$ ls
file1.txt  file2.txt  file3.txt
bong5@ubuntu:~/devel/test2$ cat file1.txt
bong5@ubuntu:~/devel/test2$ cd ../
bong5@ubuntu:~/devel$ mkdir test3
bong5@ubuntu:~/devel$ cp test2/file1.txt test2/file3.txt test3/
bong5@ubuntu:~/devel$ ls test3/
file1.txt  file3.txt
bong5@ubuntu:~/devel$ cp -rf test3/ test4
bong5@ubuntu:~/devel$ ls test4
file1.txt  file3.txt
bong5@ubuntu:~/devel$ rm test4/file3.txt test2/file*.txt
bong5@ubuntu:~/devel$ ls test4
file1.txt
bong5@ubuntu:~/devel$ ls test2
bong5@ubuntu:~/devel$ rm -rf test3
bong5@ubuntu:~/devel$ ls
test  test2  test4
```

$ cp <src-file> <dest-file>
$ cp <src-file> <dest-dir>
$ cp <src-file1> <src-file2> … <src-fileN> <dest-dir>
$ cp -rf <src-dir> <dest-dir>

$ rm <file-1>
$ rm <file-1> … <file-N>
$ rm –rf <dir-1>
$ rm –rf <file-1> .. <dir-1> <dir-2> <file-N>

Shell – Display File Content

Command	Scrollable	Exit key
$ more <filename>	No	N/A
$ less <filename>	Yes	q
$ cat <filename>	No	N/A
$ tail <filename> $ tail –n <#line> <filename>	Yes	N/A

What if I want to print file in reverse order?

$ cat <filename> ←→ $ tac <filename>

Shell – *grep* some pattern …

$ grep <pattern> <file> = search pattern in file
$ grep –rn <pattern> . = search matched pattern recursively from current path

```
bong5@ubuntu:~/devel/test$ grep -rn int test.*
test.c:4:void main(int argc, char *argv[])
test.c:6:        int i;
test.c:7:        printf("Hello world from %s \n", argv[0]);
test.c:9:                printf("input-%d = %s \n", i, argv[i]);
test.c:12:       printf("Adder %d & %d = %d", 3, 4, adder(3,4));
test.c:13:       printf("Mul   %d & %d = %d", 3, 4, multiply(3,4));
test.c:15:       printf("Exit function \n");
test.c:18:inline int adder(int i, int j)
test.c:23:inline int multiply(int i, int j)
test.h:1:int adder(int i, int j);
test.h:2:int multiply(int i, int j);
```

I like "–rn" because it searches all possibility and gives me the lines.
A very good way for understanding software & debugging

If you really want to sharpen your "grep skill", learn regular expression.
Use $ grep -e <regex> filename

Shell – *find* some file …

find filename that matches pattern from <start-path>

$ find <start-path> -name <pattern>

```
bong5@ubuntu:~/repo/linux-git/linux$ find . -name i2c-*x.c
./drivers/i2c/i2c-mux.c
./drivers/i2c/muxes/i2c-mux-pca954x.c
./drivers/i2c/busses/i2c-pnx.c
./drivers/i2c/busses/i2c-mv64xxx.c
./drivers/i2c/busses/i2c-sis96x.c
./drivers/i2c/busses/i2c-rk3x.c
./drivers/i2c/busses/i2c-iop3xx.c
./drivers/i2c/busses/i2c-imx.c
```

```
bong5@ubuntu:~/repo/linux-git/linux$ find Doc* -name i2c-*x.*
Documentation/devicetree/bindings/i2c/i2c-mux.txt
Documentation/devicetree/bindings/i2c/i2c-mux-pca954x.txt
Documentation/devicetree/bindings/i2c/i2c-mv64xxx.txt
Documentation/devicetree/bindings/i2c/i2c-pnx.txt
Documentation/devicetree/bindings/i2c/i2c-imx.txt
Documentation/devicetree/bindings/i2c/i2c-rk3x.txt
```

Shell – combining commands |<$>

		Example
|	Pipes	$ ls –al | grep "i2c*"
>, >>	Write to a file (standard output) >> is append to a file	$ dmesg > kernel.log $ echo "EOF of log" >> kernel.log
<	Standard input	$ sort < file-list.txt
$()	Output of command	$ grep fin $(find . –name fish.*)

```
bong5@ubuntu:~/repo/linux-git/linux/drivers/i2c/busses$ ls -al | grep "ii"
-rw-rw-r-- 1 bong5 bong5 19628 Sep  1 23:03 i2c-ibm_iic.c
-rw-rw-r-- 1 bong5 bong5  2703 May 29 15:46 i2c-ibm_iic.h
-rw-rw-r-- 1 bong5 bong5 19582 Sep  1 23:03 i2c-piix4.c
-rw-rw-r-- 1 bong5 bong5 10895 Sep  1 23:03 i2c-riic.c
-rw-rw-r-- 1 bong5 bong5 22546 Sep  9 23:34 i2c-xiic.c
```

```
bong5@ubuntu:~/devel/sort-test$ for i in [0 1 2]; do echo $RANDOM ; done > rando
m.log
bong5@ubuntu:~/devel/sort-test$ cat random.log
23646
22801
10806
bong5@ubuntu:~/devel/sort-test$ sort < random.log
10806
22801
23646
```

Shell – file mode (1/2)

- File mode –decides a file/folder can be "read | write | executable" FOR "user, group, other"
- Use *chmod* to change file mode.
  - $ chmod u|g|o|a+w|r|x filename
  - $ chmod [0-7][0-7][0-7] filename

RWX = 111-b = 7
RwX = 101-b = 5
rWX = 011-b = 3

U G O

```
bong5@ubuntu:~/devel/test$ ls -l testapp
-rw-r--r-- 1 bong5 bong5 8625 Sep 22 06:48 testapp
bong5@ubuntu:~/devel/test$ chmod a+x testapp
bong5@ubuntu:~/devel/test$ ls testapp -l
-rwxr-xr-x 1 bong5 bong5 8625 Sep 22 06:48 testapp
bong5@ubuntu:~/devel/test$ chmod o-x testapp
bong5@ubuntu:~/devel/test$ ls testapp -l
-rwxr-xr-- 1 bong5 bong5 8625 Sep 22 06:48 testapp
bong5@ubuntu:~/devel/test$ chmod g+w testapp
bong5@ubuntu:~/devel/test$ ls testapp -l
-rwxrwxr-- 1 bong5 bong5 8625 Sep 22 06:48 testapp
```

Shell – file mode (2/2)

File type	File mode	PATH	Executable
Command	Exe = Y	Listed	Yes
Command	Exe = Y	Not Listed	Yes but locally through ./<command>
Command	Exe = N	Listed	No
Non-command	Exe = Y	Listed	Yes then crash

Shell – File Archiving Utility

	To extract	To create archiving file
tar.gz (-z)	$ tar -zxvf <filename>.tar.gz	$ tar -zcvf <tarball_name>.tar.gz <directory>
tar.bz2 (-j)	$ tar -jxvf <filename>.tar.bz2	$ tar -jcvf <filename>.tar.bz2 <directory>
tar.xz (-J) XZ archive	$ tar -Jxvf <filename>.tar.xz	$ tar -Jcvf <tarball>.tar.xz <directory>
tar	$ tar -xvf <filename>.tar	$ tar -cvf <filename>.tar <directory>

gunzip | gzip	$ gunzip <filename>.gz
	
	# recursively de-compress files inside <directory>
	$gunzip –r <directory>
		$ gzip filename
		
		# recursively compress files inside <directory>
		$ gzip -r <directory>

zip | unzip	$ unzip <filename>.zip	$ zip -r filename.zip <directory|filename>

Shell – Networking Utility (1/3)

```
bong5@ubuntu:~/devel$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:2a:ce:a2
          inet addr:192.168.1.16  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe2a:cea2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7721 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2813 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5682216 (5.6 MB)  TX bytes:222700 (222.7 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:256 errors:0 dropped:0 overruns:0 frame:0
          TX packets:256 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:33400 (33.4 KB)  TX bytes:33400 (33.4 KB)

bong5@ubuntu:~/devel$ ping 192.168.1.16
PING 192.168.1.16 (192.168.1.16) 56(84) bytes of data.
64 bytes from 192.168.1.16: icmp_seq=1 ttl=64 time=0.069 ms
64 bytes from 192.168.1.16: icmp_seq=2 ttl=64 time=0.052 ms
```

$ ifconfig = show IP Address

$ping <ip-address|url> = to check remote is connected to you

# Normally, IP address is assigned by DHCP server when you are connected.
$ sudo ifconfig etho up|down # enable|disable Ethernet
# If you don't have a DHCP server running, then assign it yourself for intranet testing
$ sudo ifconfig etho <statically-assigned IP>

Shell – Networking Utility (2/3)

```
bong5@ubuntu:~/devel$ wget http://pad3.whstatic.com/images/thumb/3/36/Unzip-File
s-in-Linux-Step-6.jpg/629px-Unzip-Files-in-Linux-Step-6.jpg
--2014-09-22 07:51:26--  http://pad3.whst
in-Linux-Step-6.jpg/629px-Unzip-File
Resolving pad3.whstatic.com (pad3.whstat
Connecting to pad3.whstatic.com (pad3.wh
HTTP request sent, awaiting response...
Length: 99157 (97K) [image/jpeg]
Saving to: '629px-Unzip-Files-in-Linux-Step-6.jpg'

100%[===================================>] 99,157       427KB/s   in 0.2s

2014-09-22 07:51:27 (427 KB/s) - '629px-Unzip-Files-in-Linux-Step-6.jpg' saved [
99157/99157]

bong5@ubuntu:~/devel$
bong5@ubuntu:~/devel$
bong5@ubuntu:~/devel$ ls
629px-Unzip-Files-in-Linux-Step-6.jpg  test
```

$wget <URL of a downloaded file>

```
NAME
       Wget - The non-interactive network downloader.

SYNOPSIS
       wget [option]... [URL]...

DESCRIPTION
       GNU Wget is a free utility for non-interactive download of files from the
 Web.  It supports HTTP, HTTPS, and FTP protocols, as well
       as retrieval through HTTP proxies.
```

$ wget only supports "HTTP, HTTPS, FTP"

Shell – Networking Utility (3/3)

# Secure cp between local machine and remove machine over SSH
$ sudo scp <src-path> <dest-path>
# 1 of the paths must be either local or remote …
# remote path format = <username>**@**<ip-adress>**:**<path-to-file>

```
NAME
     scp – secure copy (remote file copy program)

SYNOPSIS
     scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
         [-l limit] [-o ssh_option] [-P port] [-S program]
         [[user@]host1:]file1 ... [[user@]host2:]file2

DESCRIPTION
     scp copies files between hosts on a network.  It uses ssh(1) for data
     transfer, and uses the same authentication and provides the same security
     as ssh(1).  Unlike rcp(1), scp will ask for passwords or passphrases if
     they are needed for authentication.
```

$ scp – a good way to securely copy files between two machine

For your own exploration: sftp, ethtool, nc, netstat, ssh

Shell – Disk Utility (1/2)

```
bong5@ubuntu:test$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        19G  7.2G   11G  41% /
none            4.0K     0  4.0K   0% /sys/fs/cgroup
udev            483M  4.0K  483M   1% /dev
tmpfs            99M  1.2M   98M   2% /run
none            5.0M     0  5.0M   0% /run/lock
none            494M  152K  494M   1% /run/shm
none            100M   40K  100M   1% /run/user
.host:/         167G  141G   27G  85% /mnt/hgfs
/dev/sdb3       3.0G  4.0K  3.0G   1% /mnt/sdb-vfat
/dev/sdb1       2.0G  3.1M  1.9G   1% /mnt/sdb-ext3
```

$ df -h = disk usage of all mount point

```
bong5@ubuntu:devel$ du -h -c
24K     ./test
8.0K    ./sort-test
4.0K    ./junk/test2
4.0K    ./junk/test4
24K     ./junk
8.0K    ./bash-example
72K     .
72K     total
```

$ du -h -c = disk usage in current directory

$ mount = list all mounted file-system

```
bong5@ubuntu:devel$ mount
/dev/sda1 on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/cgroup type tmpfs (rw)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
udev on /dev type devtmpfs (rw,mode=0755)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/lock type tmpfs (rw,noexec,nosuid,nodev,size=5242880)
none on /run/shm type tmpfs (rw,nosuid,nodev)
none on /run/user type tmpfs (rw,noexec,nosuid,nodev,size=104857600,mode=0755)
none on /sys/fs/pstore type pstore (rw)
systemd on /sys/fs/cgroup/systemd type cgroup (rw,noexec,nosuid,nodev,none,name=systemd)
.host:/ on /mnt/hgfs type vmhgfs (rw,ttl=1)
vmware-vmblock on /run/vmblock-fuse type fuse.vmware-vmblock (rw,nosuid,nodev,default_permissions,allow_other)
gvfsd-fuse on /run/user/1000/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,user=bong5)
```

Shell – Disk Utility (2/2)

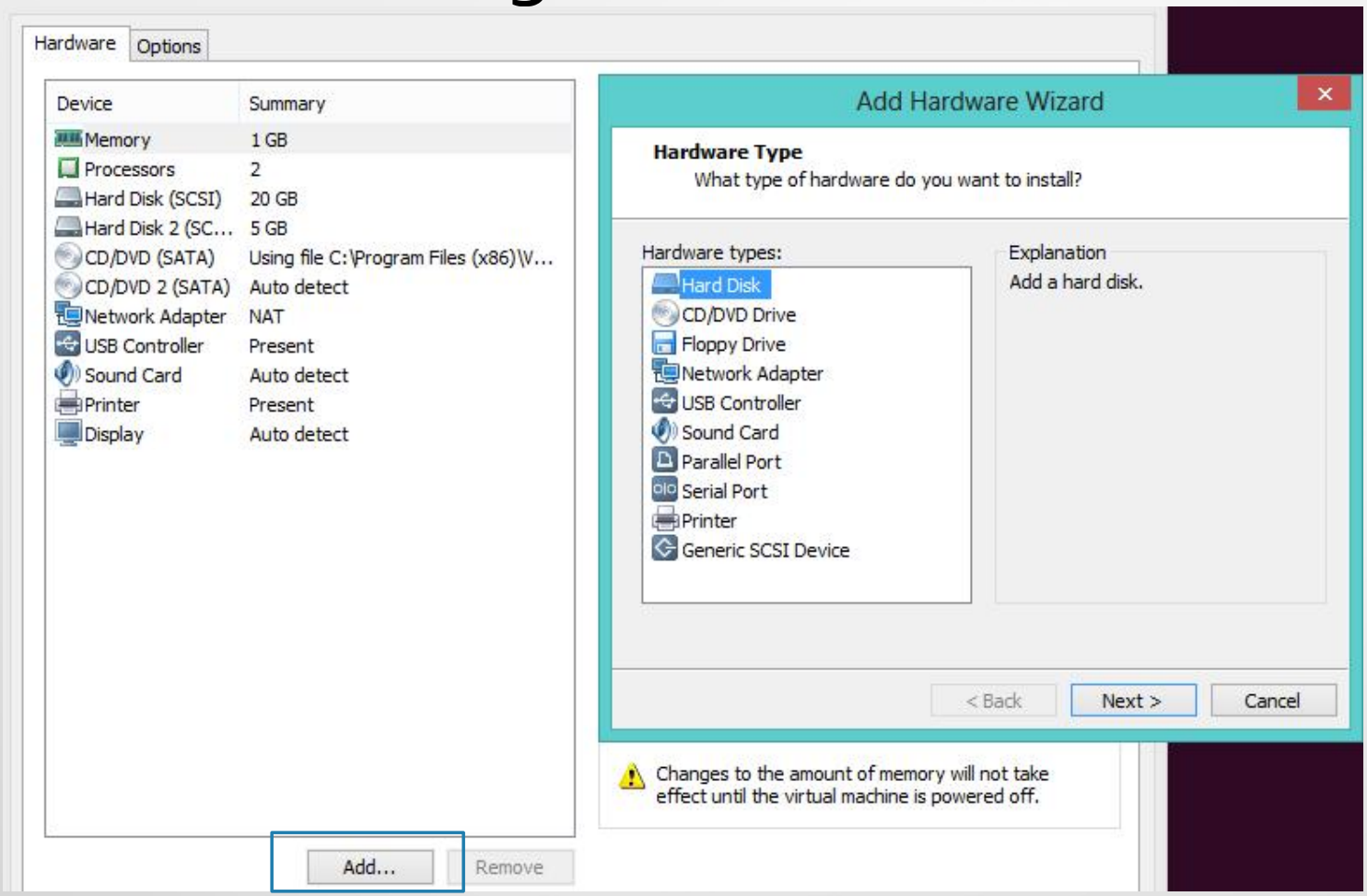
```
bong5@ubuntu:devel$ sudo fdisk /dev/sda
[sudo] password for bong5:

Command (m for help): p

Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000c7f9a

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048    39845887    19921920   83  Linux
/dev/sda2        39847934    41940991     1046529    5  Extended
/dev/sda5        39847936    41940991     1046528   82  Linux swap / Solaris

Command (m for help):
```

$ sudo fdisk <disk-device> = use for prepare partition of a newly added disk

```
sudo mkfs.ext3 /dev/sdb1
sudo mkfs.vfat /dev/sdb2
mount
cd /mnt
ls
mkdir sdb-ext3 sdb-vfat
sudo mkdir sdb-ext3 sdb-vfat
ls
sudo mount -t ext3 /dev/sdb1 sdb-ext3/
ls sdb-ext3/
sudo mount -t vfat /dev/sdb2 sdb-vfat/
ls sdb-vfat/
```

$ sudo mkfs.ext3 <disk-partition> = format partition with EXT3 FS
$ sudo mkfs.vfat <disk-partition> = format partition with MS-DOS FAT FS
$ sudo mount –t ext3|vfat <disk-partition> <path-to-mount>

Adding a new disk to VM



- Add vmdk to VM then reboot your VM to OS detect new disk
- Check disk device at **/dev/sd***
- Use **fdisk** to create partition for disk → /dev/sdb1, /dev/sdb2, etc …
- Use **mkfs.<fs>** to format partition, then **mount** the new partition

vi Editor Introduction [20-min]

Vi(m) Introduction - 1/2

For Ubuntu, by default, the Improved version of Vi (Vim) may not be installed, so bring in some software package for it, then we need to setup vi & vim to point to **vim.nox** under **~/.bashrc** as follow:

```
$ sudo apt-get install vim-common vim-nox vim-scripts vim-youcompleteme
$ gedit ~/.bashrc   # add below:
   alias vi='vim'
   alias vim='vim.nox'
```

- vi(m) is an TUX (text user interface) editor. Another option is ***emacs*** if you prefer more advanced features.
- If you are more GUI person, use ***gedit.*** But, you are losing out over time..... seriously.
- Encourage you to download "**vi cheat-sheet**" if you want to sharpen your skill as you use it ... goooooooogle...

- Operating Mode:
  - **[ESC] command mode**: to search, replace, changing setting, copy (yank) & paste, delete lines ...
  - **[insert] insert mode**: for editing (adding & delete) character
    - insert <-> replace

Vi(m) Introduction - 2/2

Command mode [ESC]	Description
:w(!) :q(!) :set (no)number :syntax off|enable	: write text buffer to file (! = by force for system files) : quit editor (! = without saving buffer to file) (not) show line number syntax highlighting
/<text> then n | *Shift+n*	Search <text>, n = next Shift+n = previous
dd *D3d*	delete current line delete 3 lines from current line
yy [move cursor] then *p* *"z10yy* [move cursor] then *Shift+p*	copy current line and paste **after** current line Copy 10 lines and paste **before** current line
:s#oldtext#newtext#g *:**%**s#oldtext#newtext#g* *:%s#oldtext##g OR* *:%s/oldtext//g*	Replace current line "oldtext" with "newtext" Replace all "oldtext" with "newtext" Remove all "oldtext" Note: # or / = valid separator
gg *Shift+g* *:123*	Move cursor to line-1 : 1st character Move cursor to last line : 1st character Move cursor to line-123 : 1st character

Break [10min]

Hand-on: Write a basic shell script [30 min]

- *Create ~/dev/bash/lab-1 folder and write the following script:*
  - *Generates 100 files with each 10 random number line inside.*
  - *Next, search through all 100 files and display lines with pattern of your choice (input to script) – i.e. a match found.*
  - *Then, get the total number of lines of such match.*

- *Tips:*
  - *All bash script starts with #!/bin/bash*
  - *Make sure file mode is executable*
  - *$1*
  - *echo $RANDOM*
  - *wc*
  - *grep*