Welcome to LinuxONE

This lab is designed to provide critical Linux environment information available to those relatively new to Linux.

The lab assumes registration for a LinuxONE Community Cloud Linux and ssh login is complete.

From ssh shell prompt, this lab suggests and briefly explains a few linux commands that are fundamental to using Linux.

The ssh Linux shell prompt in the lab instruction appears as follows:

```
[linux1]$
```

Actual ssh Linux shell prompt will look slightly different.

Linux **cat** command displays the content of a unix file to the terminal.

```
[linux1]$ cat /etc/*-release
```

/etc/*-release contains information about the installed Linux distribution. This information is very significant and will be explained later in these lab instructions.

```
[linux1]$ cat /proc/version
```

/proc/version contains additional information about the Linux kernel and GCC compiler associated with the Linux distribution

```
[linux1]$ cat /proc/sysinfo
```

/proc/sysinfo contains information about the hardware and virtualization hypervisor hosting this Linux.

Observe all the processors are available in this LinuxONE system.

Like the Linux distribution release, this information is very significant and will be explained later in these lab instructions.

However, the information streamed past. So, several ways exist to hold data from the file on the screen until ready to view more. Space bar will display next page and enter will display only the next line of the data.

Vance – I'd prefer you taught them less instead of more ;)

```
[linux1]$ more /proc/sysinfo
```

Alternate method makes use of pipe which is the vertical bar. A pipe (|) is passes output from the left of the pipe symbol to the command on the right of the pipe symbol.

```
[linux1]$ cat /proc/sysinfo  |  more
```

The following writes the output to a unix file

```
[linux1]$ cat /proc/sysinfo >  sysinfo.txt
```

The > symbol is known as a redirection. The output is being redirected from the terminal display to file sysinfo.txt

```
[linux1]$ ls
```

The **ls** command will display file names in the current file directory

```
[linux1]$ pwd
```

The pwd command will display the current present working directory where ls command listed files names.

Assuming sysinfo.txt was created in the present working directory, then the following command using >> will a**ppend** data to the sysinfo.txt

Vance – I'm pretty sure you can't have a space in between the >'s below:

```
[linux1]$ cat /etc/*-release > > sysinfo.txt
```

sysinfo.txt now contains output from /proc/sysinfo followed by output from /etc/*-release

While it is possible to display the entire sysinfo.txt, the following commands will display a few lines at the beginning of the file (head), then display a few lines at the end of the file (tail).

```
[linux1]$ head sysinfo.txt

[linux1]$ tail sysinfo.txt
```

The following will rename the file. A rename is accomplished with **mv** (move) command.

```
[linux1]$ mv sysinfo.txt sysinfo.listing
```

The following will delete or remove (**rm**) the file

```
[linux1]$ rm sysinfo.listing
```

The linux command set is very rich. A few suggested ways to learn about all the available commands is to enter one or more letters at the shell prompt, then tab key twice.

Vance – This might be confusing – if one types “ls” followed by a space, before pressing tab twice, the contents of the current directory will be listed, instead of all commands that start with “ls”

```
[linux1]$ ls <tab> <tab>
```

The output lists all the Linux commands that begin with **ls**

The output has many more Linux commands beyond **ls**

This quick tip of entering one or more letters followed by <tab> <tab> is very handy to achieving advanced level knowledge of Linux.

```
[linux1]$ man lscpu
```

The above man command provides manual pages about a specific command. In this case, manual pages for the lscpu command are displayed. Again, enter key will advance 1 line at a time , space bar will advance 1 page at a time, and q would quit returning control to the ssh shell prompt

```
[linux1]$ lscpu
```

The man pages explained the lscpu displays information about the CPU architecture.

Is that BogoMIPS value for real? Try the same command on a different hardware architecture as a comparison.

The architecture of LinuxONE hardware has technology only available in this server.   Bottom line - this architecture is made to host thousands of Linux images running concurrently where the network delay between Linux images is near zero.

Use the **man** command to learn the purpose for each of the suggested Linux fundamental commands that follow.

```
[linux1]$ lsof | more
```

```
[linux1]$ df
```

```
[linux1]$ ps
```

```
[linux1]$ ps -ef | more
```

```
[linux1]$ ls -al | more
```

```
[linux1]$ cd /
```

```
[linux1]$ pwd
```

```
[linux1]$ ls -al | more
```

```
[linux1]$ cd ~
```

```
[linux1]$ ls -al | more
```

```
[linux1]$ id
```

```
[linux1]$ env | more

[linux1]$ echo $PATH

[linux1]$ echo 'a test stream of data' > testfile

[linux1]$ cat testfile

[linux1]$ echo 'a second test stream of data' >> testfile

[linux1]$ cat testfile

[linux1]$ rm testfile

[linux1]$ date

[linux1]$ cal 9 1752
```

September 1752 is missing 11 days. An internet search will explain why. It is an interesting piece of history.

`[linux1]$ man cal` … also discusses 1752

```
[linux1]$ date +%s
```

Now do the same command

```
[linux1]$ date +%s
```

Observe number incremented

The above command displays a 10 digit unix time, aka Epoch time, aka POSIX time. Enter the 10 digit number immediately following the @ below (2 dashes precede 'date @')

```
[linux1]$ date --date @
```

Now try the following command to get the date this clock began ticking

```
[linux1]$ date --date @0
```

This date represents the beginning of unix epoch time

This date output is based on the system time zone which is relative to GMT midnight of 1970.

In computer language, when an event is scheduled for **@1600000000**, it is possible to use the date command to convert that to week day, month, day, time, and year.

```
[linux1]$ time
```

Output of time command may not be what was expected. So, what good is the time command. Time in front of any other command, shell script, or program execution will output system time consumed to complete the command, shell script, or program.

```
[linux1]$ time cat /proc/sysinfo
```

`[linux1]$ man time`   … to understand purpose of time command

```
[linux1]$ echo $SHELL
```

While different shells exist, **bash** shell is typically used by Linux distributions.

Want to learn about the richness of **bash** shell?

http://www.gnu.org/software/bash/manual/bashref.html

What is the status of modules in the Linux Kernel

```
[linux1]$ lsmod
```

Investigating a few utilities to monitor real time usage

```
[linux1]$ top
```

Cntl-c to break out of it.

Stats virtual memory

```
[linux1]$ vmstat
```

Stats about network

```
[linux1]$ netstat
```

List the open files

```
[linux1]$ lsof
```

Next commands require root authority

```
[linux1]$ sudo su -
```

The kernel-resident network interface is eth0

```
[linux1]$ ifconfig
```

Exploring the details of how TCP/IP works

```
[linux1]$ ss -l
```

```
[linux1]$ ss -o state established
```

```
[linux1]$ ss -s
```

```
[linux1]$ ifstat
```

```
[linux1]$ tcpdump -A -i eth0
```

A **cntl-c** or **cntl-z** maybe necessary to break out this

Best practice is to only be root when needed because it has the authority to accidentally mess up the environment. Exit will terminate root authority and return to user authority environment.

```
[linux1]$ exit
```

OK - out of root authority

What packages are available in this Linux distribution?

Query all packages

```
[linux1]$ rpm -qa | more
```

RPM is a powerful Package Manager, which can be used to build, install, query, verify, update, and erase individual software packages.

An entire lab is needed to explore RPM.

Firewall is typically turned on by default. Root authority is needed to work with firewall.

```
[linux1]$ gcc -version
```

Output shows the **gcc** compiler is installed along with the level.

An entire lab is needed to explore **gcc** compiler

```
[linux1]$ java -version
```

If the output of the above command shows java not found, then java needs to be installed.

What if one or more software packages are needed on the Linux distribution for a project to build the next big thing?

Popular packages include java, node.js, Apache, spark, vsftpd, mongoDB, PostgreSQL, Apache Web Server, MySQL, Perl, PHP, Python – and many more

Remember earlier comment, “This information is very significant and will be explained later in these lab instructions”?

Knowledge of the Linux hardware and software distribution is needed to install software packages.

Key things to know:

Packages compiled to be installed on LinuxONE or Linux on z will have **s390x** in the package name. Packages compiled to be installed on a 64 bit capable intel PC will have **x86_64** in the package name.

Linux is a standard that has an accepted Linux kernel used by all. Linux distributors, Red Hat, SUSE, Ubuntu, etc. all use a standard Linux kernel. However, each Linux distributor includes “value add” scripts and system utilities. The Linux distributor “value add” system utilities are different.

An example is the Linux distributor utilities to manage packages

Red Hat –	**man yum**	<<< or internet search for yum
SUSE –	**man zypper**	<<< or internet search for zypper
Ubuntu –	**man apt-get**	<<< or internet search for apt-get

An entire lab is needed to explore each Linux distribution utility to install, update, remove software, manage repositories, perform various queries, and more.

Learning to use these utilities is pretty easy.

What is the hardware platform?

```
[linux1]$ lscpu
```

```
[linux1]$ more /proc/sysinfo
```

What is the Linux distribution?

```
[linux1]$ cat /etc/*-release
```

The other way of making software available is to compile from source code. And again, this is yet another lab.

Another significant Linux administration task requiring full awareness is the firewall. Most Linux distributions are shipped with very restrictive firewall rules. This can get in the way of development progress in some cases.

```
[linux1]$ sudo su -
```

```
[linux1]$ iptables -L -n -v
```

```
[linux1]$ iptables -S
```

```
[linux1]$ iptables -h
```

```
[linux1]$ exit
```

If uncertain how to work with the firewall, (yet another lab needed), then firewall can be disabled to get something done, such as ftp data, then enable firewall when done.

The Linux distribution procedure to stop and start firewall varies among the Linux distribution and Linux releases of individual distributions.

Examples:

Red Hat

```
[linux1]$ service iptables status
[linux1]$ service iptables stop
[linux1]$ service iptables status
[linux1]$ service iptables start
```

Want to see all the linux kernel parameters ?

```
[linux1]$ sudo su -
[linux1]$ sysctl -a
[linux1]$ exit   ...get out of root
```

A command of the Linux basics will really make things fun.

The internet is loaded with linux information.

“The Linux Command Line”, a free download.
http://linuxcommand.org/tlcl.php

Linux Foundation org a Free Linux Training Online course and some Free Linux Training Videos.
https://training.linuxfoundation.org/free-linux-training?SSAID=221207

Simply speaking - Linux is fantastic and at its best on LinuxONE

Have Fun!