2006-10-03

Intro to Linux

Derek Carter

<goozbach@friocorte.com>

Linux History

In the early sixties, AT&T Bell Labs was developing a computer operating system called UNIX. This operating system was licensed to many higher institutions of education as a teaching OS. One of these universities was the University of Helsinki, in Finland. There a young student, Linus Torvalds was working on this licensed version of UNIX, branded Minux. Linus re-implemented all the different system calls of Minux and decided to release his work into the public eye. Linus named his new version Linux.

Linux was licensed under the GNU Public License, which allows for free modification of the source-code of the Linux kernel. The GNU Public License, or GPL was developed by a project called GNU, started by one Richard Stallman. The goal of the GNU project was to develop a UNIX derivative, however the GNU project started by building all the tools, whereas Torvalds started by building the kernel.

Linux Architecture

UNIX, and therefore Linux, was built upon a small set of core principles. Those include:

- Small single purpose commands
- Strictly simple output from said commands
- Loosely accepted varied input for said commands
- EVERYTHING is a file (a consistent interface)
- All configuration stored in ASCII

These core principles, as well as being written in C led to UNIX being a very easy to maintain, easy to update, and easy to port operating system. An added benefit was the greater ease for universities to use the UNIX source-code as a teaching tool.

The Linux architecture is comprised of three distinct layers. Those are:

- Hardware
- Kernel-space
- User-space

The basic level is the hardware. A Linux Kernel is compiled into the appropriate machine code for whichever processor architecture the machine is running on. The kernel then creates a consistent

kernel-level interface to all the different pieces of hardware as well as taking care of managing processes, security, and other “operating system” tasks. All other software runs in “User-space” which includes all shells, programs, GUIs etc. The main suite of tools developed by the GNU project are known as the GNU-Utilities. When you speak about Linux in the common sense you are actually talking about the GNU-Utilities running on top of the Linux kernel.

Distributions or One Brand Doesn't Fit All

Originally, installing Linux was a very belabored process. You had to have an already-installed operating system, usually Minux, then you compiled all the GNU tools, then compiled your Linux kernel. Needless to say Linux was mostly for hobbyists. Distributions solved that problem by providing a simple installer, a package (software) management tool and other perks. Some distributions of note include:

SLS

The first distro. Not very notable other than for that simple fact.

Slackware

The first distro to have a package management tool – simple tarballs with precompiled software. Still in use today.

RedHat

First commercial distro. Now branched into the community version, Fedora Core, and the commercially supported version, RedHat Enterprise Linux. Most popular distro in commercial applications domestically.

SUSE

Another commercial distro. Similarly split into the community version and the commercial version. Most popular distro worldwide.

Debian

A distro “by the people, for the people and of the people”. Very stable, not commercially supported. Very adherent to the Free Software Foundation's standards.

Ubuntu

A distro based on Debian which IS commercially supported. Probably the most popular for hobbyists today, and is rapidly gaining ground.

The choice of which distro you choose depends largely on what you want to do with Linux. Each distribution has it's strengths and weaknesses. This is a choice to be taken seriously.

How Does Linux Work? or How to Not be Afraid of the CLI

Every program you execute under Linux is known as a process. Each process has at least three communication channels; standard input, standard output, and standard error. These three communication channels can be redirected with the < and > characters. The output of one process can also be sent to the input of another process by using the “pipe” or | character.

Most commands have options which change how the command executes. These options are (usually) either prefixed by a dash - or double dash --. Any other words which are placed upon the command line are known as arguments. Arguments are used to define which objects the command should process.

Multiple arguments can be selected at once by a fairly powerful language know as wildcards or “globbing”. You have probably seen wildcards before, such as “*.exe”. The preceding wild card would match any file that ended in the characters of “.exe”.

Variables can be used in the command line as well. Setting a variable is extremely simple, specify the variable name then the equals sign, then the value like so:

```
BIRTHDAY='1980-07-31'
```

Commands can also be nested by using the back tick `echo hello` or dollar-peren $(echo hello) syntax.

Configure This

At some point you are going to have to change an ASCII configuration file. This is best done using a text editor. It is HIGHLY suggested you learn at least the basics of the vi editor, as it will almost always be present on a Unix-like system.

Help Me

One of the hardest things for a first time Linux user is how to find help. There tends to be a bit of elitism in the ranks of Linux power users. This isn't entirely unexpected. You can help alleviate the scorn, mocking, and rough answers that you would normally receive by being prepared. You'll probably hear the phrase Read the F*&&^ing Manual. This is expected. If you ask what a certain option for a commonly use program is people might get upset. This is where the man command comes into play. If you want to find out about the options and arguments for a given command may be just run the man command against the program in question. In other words “ man cp “ would give you the options and arguments for the cp command. Most commands also support the --help option, which prints out the commonly used options.

The directory /usr/share/doc is the standard location for software packages to place their documentation when they are installed on your system. Look there for useful guides, howtos, and manuals.

Another great source of help is the Internet. JFGI stands for “Just F*&^ing Google It”. Google is your friend. If you cannot find help from the man pages, Google, /usr/share/doc/ then you have other options. Contact the person/persons who wrote the software. Their email is usually somewhere in the –help output or man page. Mailing lists and IRC channels are other great resources. And never underestimate the value of your local Linux User's Group.