

# Useful Tools and Tips

Hans Kruse and Carl Bruggeman

Jan 5, 2007

## Command Line Access

All three of our operating systems are normally used through their Graphical User Interface (GUI). However, many diagnostic and low-level configuration tools require the use of a command line. Linux can be manipulated entirely via the command line, and that is often preferred for servers which are often administered remotely.

Windows inherits a (now updated) command line capability from its predecessor, DOS. Microsoft provides GUI administration tools for all configuration tasks (with a few strange exceptions you will see later in the course), as well as a GUI-based remote access capability. Nevertheless, the Windows command line contains a set of very powerful tools.

Mac OS X is built on top of BSD Unix, so it has a command line capability similar to Linux; however, Apple has introduced GUI tools not only for the typical user, but also for system administration. Mac OS X servers (and workstations like ours) can be administered entirely from the command line, but not with the “standard” Unix tools; you have to use the Apple-supplied command line tools instead. In the lab you will use GUI tools to configure Mac OS X and Windows, and use its command line for diagnostics. We will configure Linux via the command line.

## Windows

Open command windows by selecting **Start->Run** and enter `cmd` in the dialog box. You can (and should) open as many windows as you need.

## Mac OS X

The Mac operating system has two command line environments, one part of the “native” Mac OS windowing system (which Apple calls “Aqua”), and a traditional Unix X-windows environment. For most applications, the Aqua environment is what you want, the command line tool is called “Terminal”. On the Dock at the bottom of the screen that is the icon that looks like a black computer screen. When you click on the Terminal icon, the Terminal application will open multiple windows for you. You can open additional windows using the **File** menu.

In some cases you will be asked to open an application “*from an X-windows prompt*”. You bring up the X-windows environment by clicking on the Dock icon that has a big “X” in it. The program is actually called “X11”. Once X11 is running, start new command windows from the **Applications** menu.

## Ubuntu Linux

Linux is based on the X-windows environment. In our case we use the “Gnome” desktop environment on top of the X-windows system. Start a terminal with **Applications->Terminal**, you can then either open new windows from within terminal with **File->Open Terminal**, or open a new “Tab” within the same window with **File->Open Tab**.

## Privileged Commands

On all three operating systems, some commands are restricted for use by system administrators (called `root` on Unix). Not only does that prevent unauthorized users from creating problems – either on purpose or by mistake – but the fact that a command requires additional authorization can also remind an administrator to consider a command carefully before performing a potentially dangerous function. Traditionally you had to log into the system as an administrator to access these commands, and that is still the norm on Windows. Your login, `itladmin`, on Windows has been configured as an administrator, so you do not need to take any further action to execute commands.

On Unix/Linux systems, we have a better option. Being logged in as an administrator all the time has obvious security risks. So by default, `itladmin` on the Linux and Mac OS systems is not a root user. The login has simply been authorized to temporarily *become* a root user, usually for one command at a time. This can happen in two ways:

- If you are using a GUI tool, you will be prompted by a dialog box to enter your password (the same one you used to log in) to enable root access for the GUI tool.
- If you want to use a privileged command on the command line, you must preface the command with `sudo`, e.g. instead of typing `ifconfig` to set up an interface on Linux, you will type `sudo ifconfig`. The first time you use `sudo` you will be prompted for your password. Normally `sudo` will not ask for your password again if you use it a second time in a short period (typically a few minutes). In the lab we have extended this time out period to a much longer time, so you should rarely get prompted for your password again after the first use of `sudo`.

## Using ssh

The Secure Shell program, `ssh`, is a modern version of the old `telnet` program. You use it to open a new command window on a remote computer. On a Mac OS or Linux command line, type `ssh <user>@<address>` where `<user>` usually is replaced by `itladmin` (that is the username you will use on all lab machines) and `<address>` is the IP address of the machine you want to connect to. You may get a prompt asking you if you really want to connect to this host (only the first time), then you will get a prompt for your password.

On Windows you use `Start->Programs->SSH Secure Shell->Secure Shell Client` to get a GUI version of SSH. Use the `Quick Connect` button to establish an ssh connection to a remote address.

## Manipulating Text Files

In most of our labs, you will need to create and/or edit simple text files. In many cases you use text files to collect data for your lab report, e.g. capturing out from commands and diagnostic utilities. Here are the suggested tools for this purpose.

### Windows

The program to use is `Start->Programs->Accessories->Notepad`. Save your files often, and please save files *on the desktop* to make cleanup easier for the lab staff.

### Mac OS X

The simplest tool is the `TextEdit` program that comes with Mac OS. It is designed as a miniature word processor and can use RTF (Rich Text Format) files in addition to simple text. **Make sure** that you always work in plain text mode. If you see style and font drop-down menus above your text, you

are in RTF mode. Use **Format->Make Plain Text** to get back to plain text mode. You can also open a **Terminal** window and use the command **nano** to start a small text editor. The on-screen command reminders should be self-explanatory. Please save files in your home directory to facilitate cleanup. Finally, for those of you that are “old hands” in Unix, **vi** is also available.

## Ubuntu Linux

The GUI text editor of choice is **Applications->Accessories->Text Editor**. On the command line, **nano** and **vi** are available. Again, please save files in the home directory.

## Getting Command Help

You should of course always bring your note book with printouts of the readings and slides from class when you come to the lab. Be prepared for the fact that Internet/Web access will not always be available in the lab as you construct your network. Fortunately, most systems have some form of built-in help system.

## Windows

Most of the commands we use produce a help screen when you use the argument **/?**, as in **netstat /?**.

## Mac OS X and Ubuntu Linux

The Unix-based systems have a built-in documentation system called **man** (for manual). The command **man ifconfig** will display the manual for the **ifconfig** command. The text is displayed one page at a time. To navigate:

- Press the space-bar to go forward one page.
- “U” will go backward one page.
- “g” will go back to the top.
- “G” will go to the end.
- The **Enter** key will scroll down by one line.
- “h” will bring up a summary of movement commands.

You can use the command **man -k** to search for all man pages relevant to a command.

## Collecting Data

You will need to keep very careful notes during the lab, because the commands you type and their output are the data you use to write your lab report! Open a text editor on each machine as soon as you start your lab, and use the text file to collect notes. Make sure to type annotations and remarks into the file so you know what you are looking at when you review your notes. Many labs have several distinct sections; it makes sense to start a new text file for each section. To capture output from a command, you can append **> file.txt** to most commands, as in **netstat -rn > file.txt** (replace **file.txt** with something more meaningful). Or, you can copy text from a command window directly into your text file with your note.

## Copying text from command windows to text files

You can copy and paste anything you see in a command window. The procedure varies a bit between operating systems.

### Windows

Right-click in the command window and select **Mark** from the context menu. Press Enter to copy the highlighted text. In Notepad you can use the Edit menu or right-click for the context menu to select **Paste**. “**^V** ” will also paste.

### Mac OS X

Drag with the mouse to make a selection in a Terminal window. Then use **Edit->Copy**, or **Command-C**, or **Control-Click** and select **Copy** from the contextual menu. In the text editor, select **Paste** from the **Edit** or contextual menu, or press **Command-V**.

### Ubuntu Linux

Drag with the mouse to select text. Right-click and select **Copy** to copy the text. Right-click and **Paste** the text into the editor.

## Gather data in one place

We suggest that you gather all your data files in one place, then copy them to a USB storage drive. The recommended way to do this is to pull all files to the Windows system. Use the **Start->Programs->SSH Secure Shell->Secure Shell Client** program to open a connection to the Mac OS system. Use the tool bar button to open the GUI file transfer program, and download your files. Repeat the same process for your Linux machine.

## Useful Command-Line Commands

**ls** List files in the current directory on Unix. Use **ls -l** for more detail.

**dir** List files in the current directory on Windows.

**cd <directory>** Switch to the specified directory.