NPS Linux (with ROS) Tips and Tricks

There are many, many great tutorials, references and cheatsheets on the internet that you will undoubtedly need to consult as you learn to use Linux. Below, are some of the most common useful tips and tricks that may help you be productive and efficient you learn Linux for the purposes of using ROS.

Desktop Navigation

- Window Navigation: You will likely end working in a number of concurrent windows
    - **Alt+Tab** will switch between windows
- Launching Terminals: Bringing up a new a ROS system often requires a few terminal windows
    - **Ctrl+Alt+t** will open a new terminal window
    - **Shift+Cntl+t** will open a new tab in the terminal window (having multiple terminals in a single window can be easier to navigate)
- Command Line
    - Learn to use Tab Completion - this will save you many keystrokes
    - Learn to use the Up and Down arrows to browse your command line history for the same reason.
    - To exit a program running (in the foreground) on the terminal, use **Ctrl+c**
- Copy and Paste
    - In many GUI programs the copy and paste shortcuts are the same as Windows: Ctrl-c to copy and Ctrl-v to paste.
    - For the terminal window, the shortcuts are slightly different: Shift-Ctrl-c to copy and Shift-Ctrl-v to paste. You can also copy and paste through the menu: Edit->Copy or Edit->Paste

Using SSH with Git Server

There are two options for interacting with the Git remote server (clone, push, pull, etc.): HTTPS and SSH

Setting up an SSH connection between your computer and the GitLab server takes a few minutes to setup, but allows you to clone and push without entering your GitLab password.

Here are instructions on setting this connection up with GitLab. The instructions are verbose, so here is a condensed summary for our use case...

1. On your computer, generate local SSH keys

    ```
    ssh-keygen -o -t rsa -b 4096 -C "email@example.com"
    ```

    This should generate both private and public keys as files in the `~/.ssh/` directory.
2. Echo the public key to the terminal

    ```
    more ~/.ssh/id_rsa.pub
    ```

3. Copy the public key from the terminal - text starts with `ssh-rsa` and ends with the email you provided.
4. On the GitLab server
    a. Under your user account, go to `Settings->SSH Keys`
    b. Create a new key and past the text you copied from the terminal

Done - once you GitLab has your public key you should be able to clone, push and pull using the SSH protocol.

Asking for Help / Debugging

For assignments and tutorials, the best way to document problems is to use the comments section of the particular wiki page you are using. The authors, and anyone else "watching" the page are immediately notified.

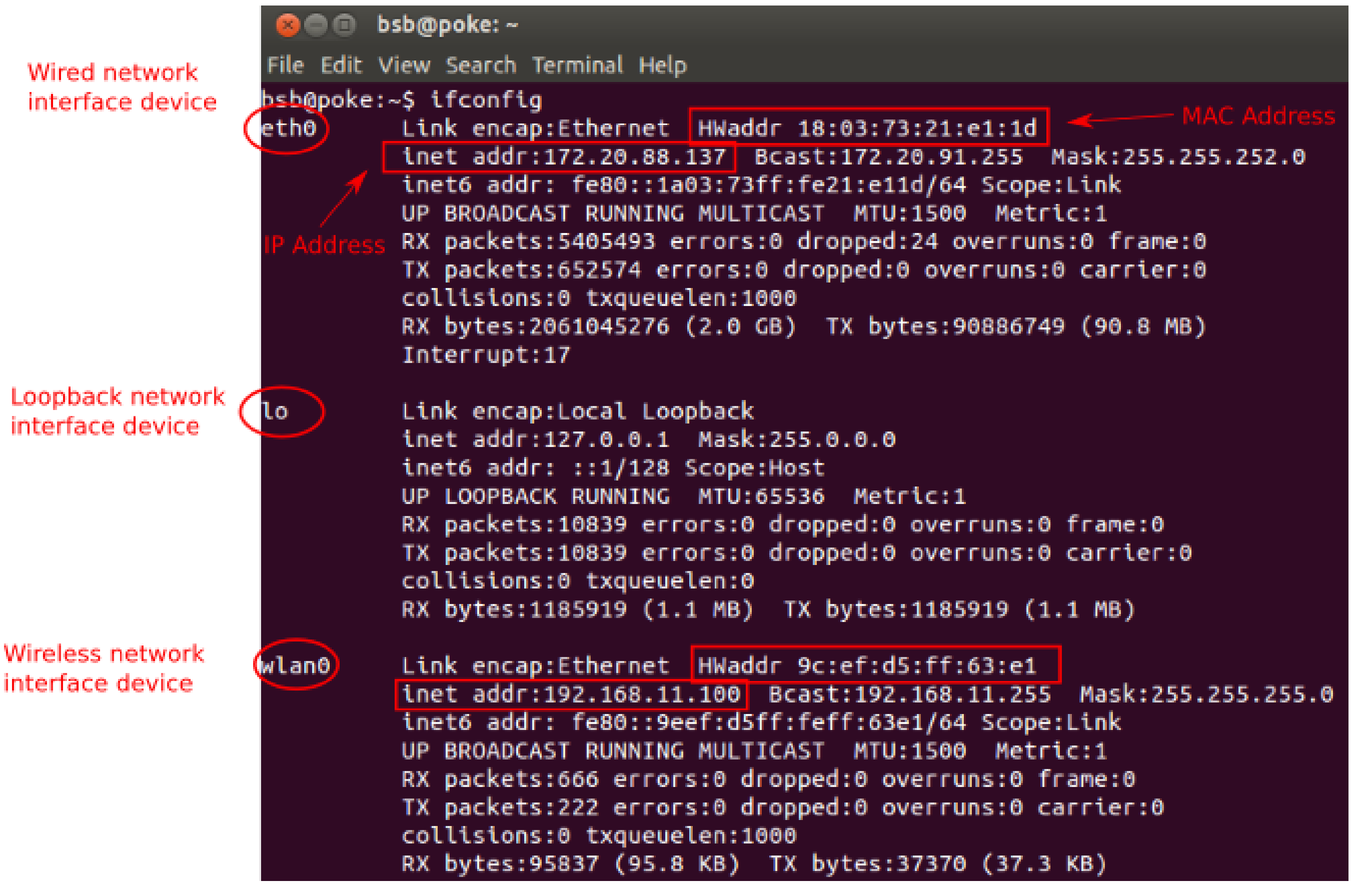
If your problem includes computer issues (errors from comments, code snippets, etc.) it is a good idea to include the commands you issued (you can use the `history` command in a terminal to see what commands you have issued) and any error reports.   If you are attempting to paste terminal commands, code, etc. into the comment, use the Code Block Macro feature to avoid formatting problems with the text you are placing in the comment.

- In general, reproducible errors are much easier to address than intermittent ones. If you can provide the commands that lead to the error and then the error output, that can be really helpful in determining what is wrong.
- Copy and paste in the Gnome terminal

Finding your IP and MAC Addresses

An easy way to examine the status of the network on your computer is with the `ifconfig` command. The example output below highlights some of the more useful parts of the output.

If there is no IP address listed in the output, then that adapter does not have an active connection.


bsb@poke: ~
File Edit View Search Terminal Help
Wired network interface device
MAC Address
IP Address
Loopback network interface device
Wireless network interface device

```
bsb@poke:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 18:03:73:21:e1:1d
          inet addr:172.20.88.137  Bcast:172.20.91.255  Mask:255.255.252.0
          inet6 addr: fe80::1a03:73ff:fe21:e11d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5405493 errors:0 dropped:24 overruns:0 frame:0
          TX packets:652574 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2061045276 (2.0 GB)  TX bytes:90886749 (90.8 MB)
          Interrupt:17

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:10839 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10839 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1185919 (1.1 MB)  TX bytes:1185919 (1.1 MB)

wlan0     Link encap:Ethernet  HWaddr 9c:ef:d5:ff:63:e1
          inet addr:192.168.11.100  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::9eef:d5ff:feff:63e1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:666 errors:0 dropped:0 overruns:0 frame:0
          TX packets:222 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:95837 (95.8 KB)  TX bytes:37370 (37.3 KB)
```

Setup Hosts File

Example - local lab wifi network with reserved IP addressed for robots. Tired of remembering and typing IP addresses. Add lines to the `/etc/hosts` file so that we specify the correspondence between IP addresses and hostnames (as listed in Robotics Computation Lab).

For example, I could add two specific turtlebots to my hosts file so that it looks like...

```
127.0.0.1    localhost
127.0.1.1    nunu
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# FRL Turtlbots
192.168.11.100    burger0
192.168.11.101    smash
```

Notice to two new lines at the bottom.

Now instead of...

```
ssh frl@192.168.11.100
```

I could equivalently use...

```
ssh frl@burger0
```

Setting Time/Date on Raspberry Pi - without internet

Setup Local NTP Server

Eg., on a laptop connected to local wifi. Following directions http://captainbodgit.blogspot.com/2013/10/raspberrypi-keeping-time-without.html

Backup, then edit /etc/ntp.conf

```
#server time.nps.edu

server 127.127.1.0 prefer
fudge 127.127.1.0 stratum 10

# Give machines on our network access to query us
restrict 192.168.11.0 mask 255.255.255.0 nomodify notrap
broadcast 192.168.11.0
```

Sync form Raspberry Pi

From Raspberry Pi

```
 sudo ntpdate 192.168.11.109
```

Where the IP adddress is the NTP server on the local network.