So you bought yourself a little Nexx (WT3020) router. Congrats. It comes with OEM firmware, so let's get some OpenWRT on it so we can hack away.

It is important that you remember the OpenWRT Hardware Page for your router: WT1520 and WT3020. These pages will provide useful information compiled by the OpenWRT community that has worked with these guys. (Another good reference is wikidev).

First, you are going to want to download the appropriate image. Now, as of 5/5/15, OpenWRT's stable releases do not support any of the Nexx routers. I'm not sure when the next release (code named "Chaos Calmer" will be out, but there is a "roadmap". For now, we will work with images from the unstable trunk. The important thing to note here is that there may be bugs out of your control, and that since new images and packages are compiled frequently, there may be times you have to run a sysupgrade in order to upgrade your kernel version to meet the requirements of a package.

SO, let's **download the image from trunk**. For **WT3020**, you can find your image **here**. More specifically, you want the **image openwrt-ramips-mt7620-wt3020-8M-squashfs-factory.bin**. (Note this image is for wt3020s with 8MB of flash memory, which, according to the hardware page, is any white guy with a USB port). For sysupgrades, use openwrt-ramips-mt7620-wt3020-8M-squashfs-sysupgrade.bin. The image for WT1520 can be found here, but note that the hardware page indicates wt1520s have 4M of flash memory. (I don't know why they list an 8M image version honestly).

Out of the box, these guys have Wifi (SSID: Nexx?) and LAN enabled with DHCP running. It's IP address is **192.168.8.1**. Type the IP address into your browser after connecting, find the page where you upload an image to flash (usually meant for upgrading firmware), upload the openwrt factory image you downloaded previously, and press apply (or whatever the text box says).

You may have to wait 0.5-1 minutes, but eventually the router will reboot with the base openwrt image configured for the Nexx router.

Here are the default configs:

- Wifi disabled (but if enabled, default SSID is "OpenWRT")
- LAN IP address: 192.168.1.1
- DHCP server on LAN enabled
- WAN port is DHCP client
- Telnet enabled.
- Inability to ssh (I believe because root password not set).

(Based off http://wiki.openwrt.org/doc/howto/firstlogin)
With these configurations in mind, clearly you have to hook yourself up to the **LAN port**, **telnet** into 192.168.1.1, then run "passwd" and set the password for root. Then "**exit**" and

reconnect with **ssh root@192.168.1.1**. (If you want to then use keys instead of a password, check out this)

Congrats! Now you are able to simply ssh into your Nexx router.
An important list of references for configuring your OpenWRT router can be found here

One important tool is the package manager "**opkg**". It works like apt-get in that you should run update then install when installing a package. The packages you will need for running our program are (install with "opkg install"):

- libopenssl
- iptables-mod-nfqueue
- kmod-ipt-nfqueue
- kmod-nfnetlink-queue
- kmod-nfnetlink
- libnfnetlink
- libnetfilter-queue

You can run "opkg info" on each of these for more info (dependencies, description, etc.). Packages can also be installed with .ipk files in case your Nexx router is not connected to the internet. You can get the ipks here (and maybe if what you need is not there, try here but I'm not positive about it.)

Important config files can be found in **/etc/config/**, such as **network**, **wireless**, and **firewall**. OpenWRT uses UCI for configuration. The link has some useful information, like a list of the config files and what they are used for. I never really really looked into it, as I ended up simply manually editing config files, but there is a command line tool that is supposed to make configuration easier.

You may want to enable wifi just in case something goes haywire with the configuration for the LAN ethernet port. You can do this by opening /etc/config/wireless and changing the value next to disable from 1 to 0.

For our setup, we have two routers, each with different LANs, both have WANs on same network. Let's say nexx_1 has LAN on 192.168.1.1 (and controls 192.168.1.0/24) and WAN ip address 192.168.3.1. nexx_2 has LAN on 192.168.2.1 and WAN ip 192.168.3.2.

Our setup is such: nexx_1 assigns 192.168.1.100 to host_1, nexx_2 assigns 192.168.2.100 to host_2. Both hosts' default route is out of it's LAN port and the nexx router's LAN ip address is its gateway (a pretty normal set up.) nexx_1's default route is out of it's WAN port (i.e. iface eth0.2) with gateway 192.168.3.2 (the WAN ip addr of nexx_2). nexx_2's default route is the same, except with gateway 192.168.3.1 (i.e. nexx_1). (NOTE: truth be told, I cannot recall why the gateway is necessary, but I am pretty certain our setup didn't work without it.)

Example: When host_1 sends a packet to host_2 at 192.168.2.100, its routing table says to use its default route, and send it out its LAN to nexx_1 at 192.168.1.1. Upon receiving the packet, nexx_1 sees that the destination ip address is not on its wan or lan, so it sends it out its default route on the WAN port to its gateway nexx_2 at 192.168.3.2. nexx_2 then receives it, recognizes the dest ip addr as being on its LAN and forwards the packet along to host_2 at 192.168.2.100.

(Your default network config should look like this)
So to set this up, let's say we are dealing with the nexx_1. Open /etc/config/network, and find the "lan" interface. This is an uci interface for the eth0.1 iface (i.e. the "LAN-labeled ethernet port"). We see it uses the static protocol (has static ipaddr), has ipaddr 192.168.1.1, subnet mask 255.255.255.0, and some other options that are not too important to us for our setup. This is the setup we want for nexx_1's LAN.
Next, we must set up our static WAN interface. See the "wan" interface already there? This is set up to be a dhcp client on iface 'eth0.2' (the "wan-labeled" ethernet port). Whether you want to keep this or not is really up to you. If you want it to be dynamically assigned an ip address by a DHCP server, then might as well. We don't really need it, so you can delete it, or just comment it out with #'s. We want to add a interface that fits our setup, so add this to the file:

```
config interface 'fallback'
        option ifname 'eth0.2'
        option proto 'static'
        option ipaddr '192.168.3.1'
        option force_link '1'
        option netmask '255.255.255.0'
        option gateway '192.168.3.2'
```

You can call it "Fallback" or whatever you want. The important part is that you designate the ifname as the wan-labelled ethernet port "eth0.2", provide a static ip, and set the gateway. By setting the gateway here, this actually sets up the default route configuration that we mentioned before.

Next, we need to load these configurations. We load the network service with init scripts. Here, we can call "/etc/init.d/network [reload or restart]". I believe reload is enough, and it won't temporarily disconnect you from the router like restart, but if reload doesn't seem to set the correct default route in the routing table (check with route -n), then do "/etc/init.d/network restart".

Running ifconfig now shows us br-lan ("br" stands for "bridge"), eth0.1 and eth0.2. There are others, like lo for loopback, eth0 for the actual network card (".1 and .2" actually represent

vlan tagging by the NIC's onboard switch, so eth0.1/2 are actually virtual network interfaces), and wlan (the wireless interface).

Running route should show routes to 192.168.1.0/24 for iface eth0.1, 192.168.3.0/24 for iface eth0.2, and default for iface eth0.2 through gateway 192.168.3.2.

Next, set up nexx_2. Do the same things, except change the lan ip to 192.168.2.1 and the "fallaback" ip addr to 192.168.3.2 with gateway 192.168.3.1.

Now the network topology is set up.