Running an amprnet gateway using NAT - How-To:

Some people are concerned from a security point of view about opening up their DMZ to host amprnet. Now this is not necessary, and in some instances is even easier to setup and maintain. I've tested this at a couple locations and it appears to work fine. There are some drawbacks to this as of this writing and I hope we'll be able to better automate this process. Please see http://www.n1uro.com/linuxconf/ampr-with-nat.html before continuing if you are not familiar with configuring amprnet or some of it's terms.

Most consumer grade routers (known to ISPs as CPE - customer premise equipment) are preconfigured for NAT (network address translation) which enables a single publically routed IP to be shared amongst a group of RFC-1918 (non-globally routed IPs such as 192.168/16, 10/8 and others) for outside internet usage. Typically speaking when a CPE opens a NAT session, there's a 5 minute keep-alive socket that remains until no more activity is sensed and it's then closed by the router. As you may have previously read, a means to pass ipencap (ip protocol 4) through requires an open socket for the bi-directional communications for the amprnet. Below I'll easily show you using your linux based system how to maintain an open socket for amprnet routing.

One downside is that you will not be able to host your own amprnet IP or block. You'll need to coordinate a link with a host that is running some form of amprnet routing and they will have to take ownership of your IP or block and reroute your amprnet space to you. This host will have to set up an ignore statement in their auto-routing routines so that they don't duplicate your IP or block in their route tables. With ampr-ripd, this would simply happen by adding your block comma delimited in their -a switch, and (for now) perm enter your IP or block to you manually in their startup script. You'll have to (in the interim) inform your host whenever your commercial IP changes so they can keep the route information to you current.

On your end and from the device you will use as your amprnet gateway device, you would have to do two things only:
1) cron a 4-minute ping to your amprnet host
2) set your policy route for your amprnet routing (I use table 1) with
   a single default route to your amprnet host, local routing for any
   other devices you may have on your lan would also need to be included
   here.

```
*/4  * * * * nobody    /bin/ping -c 1 <host's comm IP > /dev/null 2>&1
```

and restart cron. A 4-minute ping will keep that 5-minute NAT window alive from you to your host. Insure you have "-c 1" so you're not constantly hitting your host with pings. You also want to insure you use your host's commercial IP in this ping keep-alive so that you're going NAT->host.

*Note: as of this writing, I've been in contact with Terry Dawson (labrat) and we're going through the old DGIP package. This was a system originally designed by K2MF for MFNOS and ported to Linux by Terry. DGIP will allow your host to in a sense "broker" your amprnet IP or block to you upon a password protected request similar to how Hurricane Electric does it for IPv6. Once this is updated and designed for amprnet policy routing, you'll be able to eliminate the cronjob to ping your host as you'll be able to request your block from your host every 4 minutes which will by default keep your NAT socket alive.

73 de N1URO