APRS rx-only iGate

Written by Jim Kinter, K5KTF

Have APRS but need an iGate to the APRS network? HSMM-Mesh(tm) to the rescue (again!)

This was built on a USB-modified node, due to the extra capacity in memory and storage.

What I did was take my K5KTF-USB with 40GB harddrive and plug in a PL2303-compatible USB to serial adapter cable to the USB hub attached to the node.

Then attached the other end to a Kantronics KPC3 (no +) that I picked up at a swap fest a while back, fairly cheap, using a DB9-to-DB25 adapter (KPC3 has a DB25 for its serial connection).

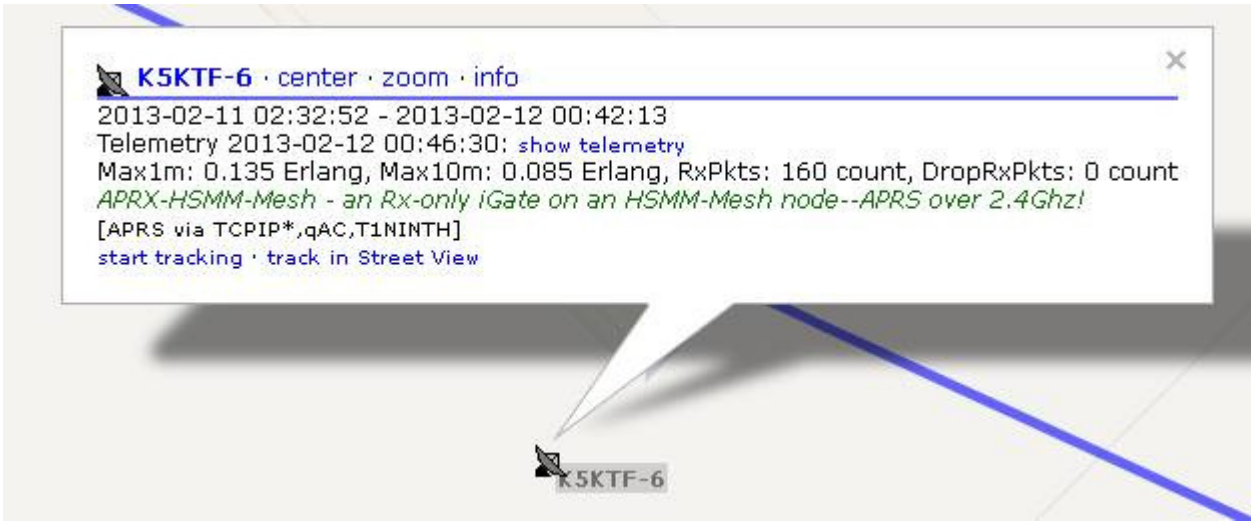
I took an old audio cable that had a 1/8" phone plug on one end and cut the other end off. I then soldered the cut end to a spare DB9-mail connector I had scarfed off an old motherboard (yes, I steal parts off scrap motherboards....). Center conductor to Pin5, shield to Pin 9. That DB9 then plugs onto the DB9-female on the KPC3 for radio-to-TNC input. The phone plug into the EXT SPK jack on the radio. I turned the volume on the radio down until the RCV light on the TNC went out, then bumped it back up a hair till it came on again. Scientific enough :-)

The radio I used is my 1st FT-2800M that it appears the TX PIN diode died (spec'd at 25W, when the full output of the finals is 50W @ 2m? hmmmm). So since it wont TX, I figured it would be good for this use, as its receiver is really good. That is attached to a dual-band 3 section vertical about 10 feet AGL outside. I will probably get it up on the tower or a push-up pole soon. But for now, good enough for testing.

Next I loaded the APRX v0.9 package compiled for use on OpenWRT/Broadcom/mipsel. Version 0.9 is a receive-only package. It will only take data from a TNC and send it to the APRS network via the internet. I have read that v2.0 is duplex (RX & TX), but at this time have not been able to find a package compiled for OpenWRT/Broadcom/mipsel. I have heard rumor that it may have too many dependencies or some such to be efficient enough for use on a node.

Then with a bit of tweaking of the /etc/aprx.conf file (my Callsign, LatLon, the APRS icon of a satellite dish, etc), it was finally showing up on the APRS network, and then soon after started passing traffic (as shown easily when you mouse over local APRS clients/cookies by the line drawn to K5KTF-6)

EXTREMELY simple to hook up ! The hardest part was getting the format of my LatLon string appropriate so that the APRS servers didnt complain.



So to make things easy for anyone wanting to duplicate this, here is my /etc/aprx.conf file, with my stuff edited out and replaced with <CHANGE THIS> tags. There are more items you can play with, which I havent yet, but probably will at some point.

==========

```
#
#  Sample configuration file for the APRX  -- an Rx-only APRS iGate
#  Chopped out extraneous comments to minimize size.... K5KTF
#

mycall          <YOUR-CALL-SSID>
aprsis-server    rotate.aprs.net  14580
#aprsis-heartbeat-timeout   600
```

```
# APRS-IS server may support some filter commands.  Although this
# program does not transmit out to RF, filter rules can be used to
# ensure that there is sufficient dataflow from APRS-IS server to
# this program that it very likely will not timeout within network
# monitoring timeout..
#
aprsis-filter    "m/100 t/p"

# Additional APRS-IS servers can be entered, by repeating following
# three configuration entries as many times as are needed.
#
#aprsis-server           rotate.aprs.net 14580
#aprsis-heartbeat-timeout 120
#aprsis-filter "m/100 t/p"

# AX.25 filters block selected messages matching on selected regular
# expressions.  The expressions are case sensitive, and AX.25 address
# elements are in all uppercase text.  There can be unlimited number
# of patterns, type fields are four: "source", "destination", "via",
# and "data".  These patterns can be used in addition to built-in
# hard-coded reject rules listed in documentation.
#
#ax25-reject-filter source      "^NOCALL"
#ax25-reject-filter destination "^NOCALL"
#ax25-reject-filter via         "^NOGATE"
#ax25-reject-filter data        "^\\?"

# ax25-rxport limits reception on listed AX.25 ports (their callsigns),
# if system happens to use AX.25 ports also for other purposes than APRS.
# If this option is not used, all reception ports are accepted.
# Number of port definitions here is unlimited.
#
#ax25-rxport NoCALL
#ax25-rxport N0CALL

pidfile /var/run/aprx.pid
rflog /var/log/aprx/rf.log
aprxlog /var/log/aprx/aprx.log

# erlangfile defines a mmap():able binary file, which stores
# running sums of interfaces upon which the channel erlang
# estimator runs, and collects data.
# Depending on the system, it may be running on a filesystem
# that actually retains data over reboots, or it may not.
# With this backing store, the system does not loose cumulating
# erlang data over the current period, if the restart is quick,
# and does not stradle any exact minute.
# (Do restarts at 15 seconds over an even minute..)
# This file is around 0.5 MB per each interface talking APRS.
# Things go BADLY WRONG if this file can not be created or
# it is corrupted!
#
# Built-in default value is: /var/run/aprx.state
#
#erlangfile /var/run/aprx.state

# erlang-loglevel is config file version of the "-l" option
# pushing erlang data to syslog(3).
# Valid values are (possibly) following: NONE, LOG_DAEMON,
# LOG_FTP, LOG_LPR, LOG_MAIL, LOG_NEWS, LOG_USER, LOG_UUCP,
# LOG_LOCAL0, LOG_LOCAL1, LOG_LOCAL2, LOG_LOCAL3, LOG_LOCAL4,
```

```
# LOG_LOCAL5, LOG_LOCAL6, LOG_LOCAL7.  If the parameter value is
# not acceptable, list of accepted values are printed at startup.
#
#erlang-loglevel NONE

# erlanglog defines a rotatable file into which erlang data
# is written in text form.
#
#erlanglog /var/log/aprx/erlang.log

# erlang-log1min option logs to syslog/file also 1 minute
# interval data from the program. (In addition to 10m and 60m.)
#
#erlang-log1min

##### Adjust the /dev/  to match your serial port on the mesh node !
radio serial   /dev/usb/tts/0  9600 8n1    KISS  callsign <YOUR-CALL-SSID>

# Additional/alternate options for the "radio" line.
#
# "initstring" is of two parts, the keyword, and then a string.
#    initstring "\xC0\xC0\xFF\xC0\r\nMO 0\rKISS $01\r"
#
# "callsign NAME" sets callsign used in statistics displays,
#    and when the message is sent to APRS-IS.
#    If none are given, then it will use physical port name.
#
# "timeout 900" sets a timeout monitor (in seconds) to make
#    reopen/reconnect if the (tcp) connection to radio has
#    failed somehow and nothing is heard. Local serial ports
#    do not (in general) need this.  At APRS silent sites
#    this may cause repeated reconnects, but it should not
#    harm either.  At busy sites this will handle reconnect
#    gracefully in case of network failures, and timeout
#    value can be shortened.
#
#  "KISS"                - plain basic KISS mode
#  "XORSUM" alias "BPQCRC" - KISS with BPQ "CRC" byte
#  "SMACK" alias "CRC16"   - KISS with better CRC
#  "TNC2"                - TNC2 monitor format


# The  radio tcp  option defines a connection to remote socket
# beyond which is a binary transparent connection to a serial
# port.  The parameter fields:  literal IP address (IPv4 or IPv6),
# then literal port number, and finally protocol mode.
# KISS-protocol parameters are same as with normal serial port.
#
#radio tcp  12.34.56.78  4001  KISS  timeout 900 callsign N0CALL-12
#radio tcp  12.34.56.78  4002  TNC2  timeout 300 callsign N0CALL-11
#


# The  netbeacon  option.
# Parameter string (in quotes) is sent to network (without quotes)
# at varying intervals --  1200-1800 seconds in between restransmits.
# This interval is intentionally randomized.
#
# Multiple netbeacons are evenly distributed for each time period,
# or at least 3 seconds apart.  First netbeacon is sent to network
# 30 seconds after connection to APRS-IS.
```

```
#
# There can be multiple netbeacon options.
# The parameter sets can vary:
#  a) traditional "just a string" (but in quotes)
#  b) 'for nnn-n raw "string"'
#  c) 'for nnn-n symbol "R&" lat "ddmm.mmN" lon "dddmm.mmE" [comment "any text"]
# The c) form flags on some of possible syntax errors in parameters.
# It will also create only "!" type messages.
#
# Symbol  R&   is for "Rx-only iGate"
#
#netbeacon  "!3051.90NR09787.89W&aprx over HSMM-Mesh - an Rx-only iGate running on HSMM-Mesh"
## The format for your Lat and Lon are DDMM.MMN and DDDMM.MMW for North America--so if you were 30 deg 50.55min would be 3050.55N and 09934.55W for -099*34.55' (west)
## This will also give you the parabolic dish icon - the / after the N for North and the ` (lower case ~ on your kbd) after W for West.

netbeacon for <YOUR-CALL-SSID> raw "!<YOUR-LATITUDE>N/<YOUR-LONGITUDE>W`APRX-HSMM-Mesh - an Rx-only iGate on an HSMM-Mesh node--APRS over 2.4Ghz!"
#netbeacon for <YOUR-CALL-SSID> symbol "R&" lat "3031.13" lon "09752.71" comment "APRX-HSMM-Meshnode-iGate"
==================================================
```

Oh, yeah.... you might need this:

RIGHT CLICK ME AND SAVE-AS to your PC <<the APRX IPK file --kudos to Rusty AE5AE for supplying. I will try and get this on the firmware package section of the server, so it will show up in the package list if you refresh to install directly.

Last Updated on Tuesday, 12 February 2013 01:23