Successful APRS IGate Operation

WB2OSZ, January 1, 2017

Part 1 - Introduction

Dire Wolf can serve as a gateway between the APRS radio network and APRS-IS servers on the Internet.

This Internet Gateway for APRS is known as an IGate. An IGate is a critical component of the APRS messaging infrastructure.

IGate stations allow communication between disjoint radio networks by allowing some content to flow between them over the Internet.

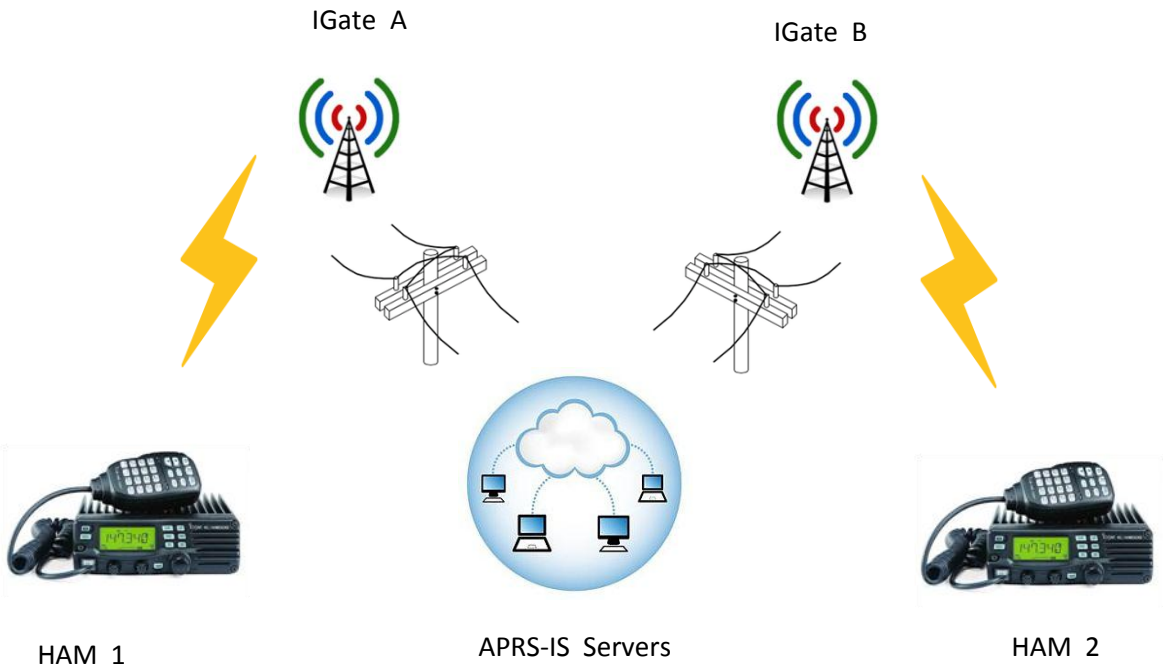
IGate stations also allow someone without a radio, or outside of your local area, to view APRS activity at http://aprs.fi or http://findu.com or with other applications. Information can also be relayed from the Internet Servers (IS), through your station, on to the radio channel.

You will see mention of receive-only IGate stations which means they don't transmit on the radio channel. It is recommended that all IGates are bidirectional so desired information can flow in both directions.

This guide to setting up an APRS Internet Gateway (IGate) with Dire Wolf contains information from scattered places in the Dire Wolf User Guide, additional background, and tips.

Typically an APRS Internet Gateway (IGate) station should be configured to transmit only "messages" to recently heard nearby stations. If you transmit much more than that you will probably be unpopular for clogging up the radio channel.

Let's review how messaging is supposed to work.



Ham1 wants to send a "message" to Ham2. "Message" refers to a very specific type of APRS packet which might look like this:

```
HAM1>APRS::HAM2      :Happy Birthday{001
```

First we have the source address, where it came from. The AX.25 destination field doesn't matter in this case and usually identifies the system type. The first byte of the information part is called the Data Type Indicator. In this case it is ":" to indicate message type. The Addressee comes next and must be exactly 9 characters. Spaces are added if it is shorter. Next is the message text and a sequence number.

IGate A receives this over the radio and passes it along to the APRS-IS Servers. They maintain a list of where stations have been heard during the past few hours. In this example, it turns out that IGate B is one of the stations that heard Ham 2 over the radio recently. An APRS-IS server passes the message along to IGate B and possibly others. IGate B transmits the message over the radio. The Ham 2 station replies with an **ack**nowledgement that message with sequence number 001 was received. It might look like this:

```
HAM2>APRS::HAM1      :ack001
```

This is picked up by IGate B, sent to the servers, and passed along to IGate A because it recently reported something from Ham 1. IGate A transmits it and it is received by Ham 1. This provides confirmation that the message reached the addressee station. If the "ack" is not received within a certain amount of time, the Ham 1 station automatically resends the message a few more times before giving up. The originating user would normally be given an indication that the ack was received or multiple attempts at delivery failed.

Note that the two stations can be anywhere in the world. They only need to be in range of IGate stations, possibly with the aid of digipeaters along the way. If the two stations don't transmit position reports, their exact locations won't be known. It shouldn't matter in this case. The important information is the IGate stations that can hear them.

There is one other subtle detail. When an IGate station transmits a message over the radio, it needs to remember the sender station address. The next time a position from the station is seen, it is also transmitted even if the usually filtering rules would not allow it.

Now let's take a look at the requirements for the IGate stations. This simplified description is based on http://www.aprs-is.net/IGateDetails.aspx and close enough for our discussion here. We have a more in-depth analysis near the end of this document for those who care about the details.

> *The following is the basic criteria for what an IGate gates to/from RF.*
>
> *A. Gate all packets heard on RF to the Internet Server (IS), with a couple of exceptions.*
> *B. Gate message packets and associated position of sender to RF if the destination station has been heard within range within a predefined time period and distance.*
> *C. Optionally gate additional packets to RF based on criteria set by the sysop (such as callsign, object name, etc.).*

Part 2 – Quick Start Guide

Assuming that you already have working Dire Wolf installation, here are the essential steps for configuring it to be an APRS IGate. Some of the steps assume you are using version 1.4 or later.

Receive Configuration

Setting up a receive-only IGate station with Dire Wolf is easy. You need to add two new lines to the configuration file to log in to an APRS-IS server.

First you need to specify the name of a Tier 2 server. The current preferred method is to use one of these regional rotate addresses:

- noam.aprs2.net - for North America
- soam.aprs2.net - for South America
- euro.aprs2.net - for Europe and Africa
- asia.aprs2.net - for Asia
- aunz.aprs2.net - for Oceania

Each name has multiple addresses corresponding to the various servers available in your region. Why not just specify the name of one specific server? This approach offers several advantages:

- Simplicity – You don't need to change your configuration as new servers become available or disappear.
- Resilience – If your current server becomes unavailable, another one will be found automatically.
- Load balancing – Picking one at random helps to spread out the load.

Visit http://aprs2.net/ for the most recent information.

Add two commands to the configuration file like this, using your region, callsign, and passcode.

```
IGSERVER  noam.aprs2.net
IGLOGIN  WB2OSZ-5  123456
```

If you don't have a passcode, and can't figure how to obtain it, contact the author.

You can optionally add a "client side" filter to limit what is passed from RF to the Server. Generally you wouldn't want to do this but there might be special circumstances. Details are in the client side filtering section later.

Transmit Configuration

You are highly encouraged to make your IGate two-way to aid in APRS messaging. All you need is one more configuration option.

Specify two additional pieces of information: the radio channel for transmitting and the via path for the packet header. Examples:

```
IGTXVIA  0  WIDE1-1,WIDE2-1
IGTXVIA  1  WZ9ZZZ
IGTXVIA  0
```

In the first case, packets will be transmitted on the first radio channel with a path of WIDE1-1,WIDE2-1, for a total of 2 possible digipeater hops. In the second case, packets are transmitted on the second radio channel and directed to a known nearby digipeater with wide coverage. This provides a maximum of one digipeater hop. In the third case, there will be no digipeating. The destination station must be able to hear you without any digipeaters along the way.

In locations with densely populated IGate stations, don't exceed 1 hop.

The maximum number of digipeater hops is used in two other places:

- The maximum digipeater hops influences the local station count (LOC_CNT) in the IGATE status beacon. In the first case, LOC_CNT would include stations heard with a maximum of two used digipeater hops. In the second case, LOC_CNT would include only those heard directly or via one digipeater. In the third case, LOC_CNT will be the same as DIR_CNT, only stations heard directly.

- This might enter into the decision of whether to transmit a "message" to a local station.

You can optionally add a client side filter to limit what is passed from the Server to RF. Details are in the client side filtering section later.

Typical Configurations for Messaging

The servers have a tendency of sending us too much unexpected stuff. In earlier versions, lack of an explicit client side filter often resulted in too much undesired radio traffic. Starting with version 1.4, there is now a reasonable default filter when going from the IS to RF. It is equivalent to:

```
FILTER IG 0 i/30
```

Client-Side filters are explored in more depth in a later second.

Suppose you wanted to forward messages to stations within 50 km, regardless of digipeater hops required:

```
FILTER IG 0 i/30/8/42.6/-71.3/50
```

Personally, I think the physical distance restriction for messages is a bad idea. Others may disagree. We might not know the location of the nearby stations that send packet types other than Position Report. We could relax the restriction a bit by allowing anyone heard directly, in the past hour, even if we don't know the location:

```
FILTER IG 0 i/30/8/42.6/-71.3/50 | i/60/0
```

Suppose we are in the USA, not too far from the Canadian border. We might want to avoid sending messages across the border due to third party traffic legal concerns. In this case, we could add another filter requiring the addressee to begin with W, K, A, or N.

```
FILTER IG 0 ( i/30/8/42.6/-71.3/50 | i/60/0 ) & g/W*/K*/A*/N*
```

If you want to allow additional types of packets, just append the 'or' operator and something creative. Here we will also transmit any telemetry data from WB2OSZ.

```
FILTER IG 0 i/30 | ( t/t & b/WB2OSZ )
```

Of course you would also need to ask for additional types of packets, from the server, with a server side filter.

Message Sender Position Report

There is one subtle point to message passing.

When an IGate transmits a "message" it needs to keep track of the original sender. The next time a position report, from the sender, is seen, it needs to be transmitted regardless of any other filtering rules.

"messages" have a built-in retry mechanism, other packet types don't. You might want to allow more than one position report from a successful message sender, to increase chances of it being received. You might want to disable this feature.

This is configured with the message sender position quantity option. 1 is the default. A larger number will allow additional position reports to be transmitted. 0 means disable the feature.

```
IGMSP 1
```

Advertising your station

You can advertize your station by using different types of beacons.

Position Beacon

The Position Beacon description is rather long and not repeated here. Please refer to the Dire Wolf User Guide for details. As a typical example, you might use something like this to transmit your station position over the radio.

```
PBEACON delay=0:30 every=10:00 symbol="igate"
                                overlay=T lat=42^37.14N long=071^20.83W
```

This means wait 30 seconds after Dire Wolf starts up. Then, every 10 minutes after that, send a Position Report with the specified symbol and latitude / longitude.

The overlay character indicates the type of IGate configuration. From http://www.aprs.org/symbols/symbols-new.txt

R Receive only IGate. (not recommended)
T Transmit with path set to 1 digipeater hop at most.
2 Transmit with 2 digipeater hops. Probably not a good idea unless in very sparsely covered area where IGates are few and far between.

Suppose you had a receive only station but wanted to put it on the map for those viewing http://findu.com or http://aprs.fi. In this case you can send the beacon directly to the server rather than over the radio. Just add the "SENDTO=IG" option in the beacon configuration

```
PBEACON sendto=IG delay=0:30 every=60:00 symbol="igate"
                                overlay=R lat=42^37.14N long=071^20.83W
```

IGate Status Beacon

IGate stations will often send occasional status reports with statistics.

The timing (DELAY, EVERY), transmission channel (SENDTO), and digipeater path (VIA) are the same as for the other types of beacons. Any other options, not listed below, will be ignored.

Keyword	Description	Example values	Comment
DELAY	Time, in minutes or minutes:seconds, to delay before sending first time. Default is 1 minute.	1 0:30	One minute. Half minute.
EVERY	Time, in minutes or minutes:seconds, between transmissions. Default is 10 minutes.	10 9:45	Ten minutes. 9 ¾ minutes
SENDTO	Radio channel for transmission or "IG" to send to Internet Gateway. Default is the first, or only, radio channel 0. "R" followed by a number simulates signal received on that channel.	1 IG R0	Second radio channel. Internet Gateway. Simulated channel 0 reception.

DEST	Explicit destination field for AX.25 packet. Normally you will want the default which identifies the software version.	CQ	"SPEECH", "MORSE", and "DTMF" are special cases explained elsewhere.
VIA	Digipeater path. Default none.	WIDE1-1 WIDE1-1,WIDE2-1	Upper case only. No spaces.

A few examples:

```
IBEACON
IBEACON DELAY=30 EVERY=30 VIA=WIDE1-1
IBEACON SENDTO=R0 DELAY=10 EVERY=10
```

The last one will just display statistics on your screen, for your own use, and not actually transmit it.

The information part of the packet will look something like this:

```
<IGATE,MSG_CNT=2,PKT_CNT=0,DIR_CNT=10,LOC_CNT=35,RF_CNT=45,UPL_CNT=122,DNL_CNT=456
```

It contains several identifier / value pairs.

MSG_CNT — Number of APRS "messages" from the Internet Server which have been transmitted over the radio.

PKT_CNT — Number of other (non-message) packets from the Internet Server which have been transmitted.

DIR_CNT — Number of stations heard directly (without going through any digipeaters) during the past 30 minutes.

LOC_CNT — Number of "local" stations which IS-to-RF packets are expected to reach. This is based on the via path specified for IGate transmission. For example, if the path was WIDE2-2, packets could travel up to 2 digipeater hops. LOC_CNT is the number of stations heard with this number of used digipeater addresses or fewer.

RF_CNT — Number of stations heard in the past 30 minutes regardless of the number of digipeater hops along the way.

UPL_CNT — Number of packets which have been uploaded to the Internet Server. Most of them probably came from the radio but it is also possible to generate beacons and send them to the Server rather than transmitting them. (i.e. "SENDTO=IG" option)

DNL_CNT — Number of packets which have been downloaded from the Internet Server. The number actually transmitted (sum of MSG_CNT + PKT_CNT) can be lower due to filtering and transmit rate limiting.

MSG_CNT and LOC_CNT are from the original APRS specification.

PKT_CNT, DIR_CNT, and RF_CNT followed precedent set by APRSISCE32.

UPL_CNT and DNL_CNT are unique to this software.

Part 3 – Advanced Configuration

The simple configuration above should be appropriate for most common situations. More advanced options are available when you want to customize behavior for special situations.

Server Side Filters

The word filter was introduced in this context as the original authors of the server software sought to limit the fire hose of data available from a "full feed". Unfortunately, the context has changed, and a better phrase would be "Server Subscription". With many clients, this subscription entered into a field named "filter" is sent to the APRS-IS server, which in turn sends the desired data to display on a screen.

You can subscribe to additional types of information as described here: http://www.aprs-is.net/javaprsfilter.aspx . The important thing to remember is that each subscription adds to the amount of information sent. Adding "filters" does not limit the amount of information sent to the IGate. Each one adds more. This example means that I want to get weather packets within 50 km of my station.

```
IGFILTER  t/w/WB2OSZ-5/50
```

Note that the parameter for "IGFILTER" is sent to the server and interpreted there. It is not processed locally.

The Internet Servers will often send more than what you are expecting, so you will usually want to apply an additional client side filter as described later.

> **Important!**
>
> Do not confuse this "IGFILTER" (server side) with the "FILTER" (client side) command which is processed by Dire Wolf. Here we are simply passing along the filter specification and not processing or checking it in any way.

Note: In an IGate configuration, this subscription should be limited to just a few objects in order to support the local information Initiative, described here: http://www.aprs.org/localinfo.html

The timing of these transmissions, including beacons should be conservative, such as direct every ten minutes and more hops once an hour.

Typical stuff sent by the Server

Below is a very small sample of some of the packets from a Server when no server side filter was sent to it. You might expect it to just provide "messages" for nearby and position reports from the senders of those messages. Here we find position reports, telemetry data, an item, an object, and status. The

point I'm trying to make is that the Server will probably send a lot more than you are expecting. You will be unpopular if you transmit all of it and clog up the radio channel. Pay attention to the filters and observe what is going on for a while.

N1LMA>APU25N,TCPIP*,qAC,T2NUENGLD:@250058z4123.63N/07148.85W_094/001g159t042r002p003P003h98b10253CRSnet {UIV32N}

N3LLO-4>APRX28,TCPIP*,qAC,T2NUENGLD:T#474,21.4,0.3,114.0,4.0,0.0,00000000

N1WJO>APWW10,TCPIP*,qAC,T2MAINE:)147.120!4412.27N/07033.27WrW1OCA repeater136.5 Tone Norway Me

N1RCW-2>APU25N,TCPIP*,qAC,T2NUENGLD:>250100zDX: K1CKK-2 41.45.79N 69.59.43W 16.8 miles 75  19:52

N1RCW-2>APU25N,TCPIP*,qAC,T2NUENGLD:;W1SGL-R *040110z4141.93NE07018.20W0146.730- PL 67.0 Echolink Enabled

K1RTA-1>APU25N,TCPIP*,qAC,T2NUENGLD:@250114z4154.23N/06959.62W_126/001g003t045r000p000P000h84b10267 WX STATION AND DIGI {UIV32N}

AB1OC-10>APWW10,TCPIP*,qAC,T2IAD2:=4242.70N/07135.41W#(Time 0:00:00)!INSERVICE!!W60!

W1TG-1>APU25N,TCPIP*,qAC,T2NUENGLD:>250154zDX: N3LLO-2 43.22.27N 71.50.55W 59.6 miles 301 20:53

Sometimes we see **qAR**, instead of **qAC**, and the format is different.

N1YG-1>T1SY9P,WIDE1-1,WIDE2-2,qAR,W2DAN-15:'c&<0x7f>l <0x1c>-/>

W1HS-8>TSSP9T,WIDE1-1,WIDE2-1,qAR,N3LLO-2:`d^Vl"W>/'"85}|*&%_'[|!wLK!|3

N1RCW-1>APU25N,MA2-2,qAR,KA1VCQ-1:=4140.41N/07030.21W-Home Station/Fill-in Digi {UIV32N}

The difference is explained in the "Server Generated" section here: http://www.aprs-is.net/q.aspx

Client Side Filtering

After setting an appropriate "**server-side**" filter with "IGFILTER," the server might send more than you want, creating excessive clutter on the radio channel. It is possible to apply another stage of filtering inside of Dire Wolf, the "client-side."

Each filter specification has where the packet is coming from, where it is going to, and what to allow through. A digit represents a radio channel and "IG" means the IGate server. Normally you want everything from the radio to be sent to the Server. However, if you wanted to limit it somehow, you could do it like this.

```
FILTER 0 IG  t/m          Only "messages" from channel 0.
FILTER 1 IG  t/wn         Only weather from channel 1.
FILTER 2 IG               Nothing from channel 2.
```

Much more often you will want to fine tune what gets transmitted. Note the reverse order. From IG to radio channel 0.

```
FILTER IG 0  t/mwn
```

The differences between the two types of filtering are summarized below.

	Server-side filtering	**Client-side filtering**
Configuration file	IGFILTER	FILTER
Where defined	http://www.aprs-is.net/javaprsfilter.aspx	"**Packet Filtering**" section of the Dire Wolf User Guide, included again below.
Where processed	APRS Internet Server (IS)	Inside of Dire Wolf application.
Expressions with & | ! ( )	No	Yes
Precise control over what is passed through	No	Yes
Can be different for each radio channel	No	Yes

Client Side Filter Expressions

The filter expression is loosely based on "***Server-side Filter Commands***" http://www.aprs-is.net/javaprsfilter.aspx with the addition of logical operators to combine the filter results. For example, you could decide to digipeat only telemetry originating from WB2OSZ or object reports not within a certain distance of a given location.

```
FILTER  0  0  ( t/t & b/WB2OSZ ) | ( t/o & ! r/42.6/-71.3/50 )
```

It's not necessary to put quotes around the filter expression even though it contains spaces.

Logical Operators

The individual filter specifications return a true or false value depending whether the current packet satisfies the condition. These results can be combined into larger expressions to permit very flexible configuration. The operators are:

|	Logical OR. Result is true if either argument is true.
&	Logical AND. Result is true if both arguments are true.
!	Logical NOT. This inverts the value of the following part.
( )	Parentheses are used for grouping.

& has higher precedence than the | operator so the two following forms are equivalent:

```
w & x | y & z
( w & x ) | ( y & z )
```

This is the same as the rule for multiplying and adding. When evaluating the arithmetic expression, a * b + c *d, you would first multiply a * b, then multiply c *d, and finally add the two products together.

When in doubt, use parentheses to make the order more explicit.

Filter Specifications

The filter specifications are composed of a lower case letter, the special character to be used as a field separator, and parameters. These two are equivalent:

```
b/W2UB/N2GH
b#W2UB#N2GH
```

Other implementations allow only the "/" separator character. This extra flexibility comes in handy when you want to use the "/" character in a parameter value.

Everything is case sensitive. This means that upper and lower case are not equivalent.

All Filter Specifications must be followed by a space. This is so we can distinguish between special characters that are part of the filter or a logical operator.

Wildcarding

Most of the filters allow the "*" character at the end of a string to mean match anything here. This operates on character strings without any knowledge of the callsign-SSID syntax. If you wanted to match "W2UB" regardless of any SSID, your first reaction might be to use

```
b/W2UB*
```

This would not be correct because it would also match W2UBA, W2UBZ, and many others. The correct form would be:

```
b/W2UB/W2UB-*
```

This will match only that callsign (implied SSID of zero) or that callsign followed by any SSID.

Range Filter

r/*lat***/***lon***/***dist*

This allows **position** and **object** reports with a location within the specified distance of given location.

Latitude and longitude are in decimal degrees. (negative for south or west.)
Distance is in kilometers.

Note that this applies only to packets containing a location. It will return a false result for other types such as messages and telemetry. If you wanted to digipeat stations only within 50 km you might use something like this:

```
FILTER  0  0  r/42.6/-71.3/50
```

This would reject other types of packets such as messages and telemetry. To allow them, use the "or" operator to also allow all types other than position and object:

```
FILTER  0  0  r/42.6/-71.3/50 | ( ! t/po )
```

Budlist Filter

b/*call1***/***call2*…

Allow all packets from the specified calls. These must be exact matches including the SSID. Wildcarding is allowed.

When combined with the "!" (not) operator, it can be used to reject packets from specified calls.

Object Filter

o/*obj1***/***obj2*…

Allow objects and items whose name matches one of them listed. Wildcarding is allowed.

Type Filter

t/*poimqstunw*

List one or more of the following letters for types of packets to be allowed.

p	-	Position Packets
o	-	Object
i	-	Item
m	-	Message
q	-	Query
s	-	Status
t	-	Telemetry
u	-	User-defined
n	-	NWS format
w	-	Weather

Symbol Filter

s/*pri***/***alt***/***over*

"*pri*" is zero or more symbols from the primary symbol set.
"*alt*" is one or more symbols from the alternate symbol set.
"*over*" is overlay characters. Overlays apply only to the alternate symbol set.

Examples:

`s/->`	Allow house and car from primary symbol table.
`s//#`	Allow alternate table digipeater, with or without overlay.
`s//#/\`	Allow alternate table digipeater, only if no overlay.
`s//#/SL1`	Allow alternate table digipeater, with overlay S, L, or 1

Digipeater Filter

d/*digi1***/***digi2*…

Allow packets that have been repeated by any of the listed digipeaters. Wildcarding is allowed.

If you wanted to run a "fill in" digi, which would repeat packets only if heard directly, use this:

```
FILTER 0 0 ! d/*
```

That means, when digipeating from channel 0 to channel 0 allow packets only if they have **not** been digipeated through some other station.

Via digipeater unused Filter

v/*digi1***/***digi2*…

Allow packets that have any listed digipeaters that don't have the "has-been-used" flag set. Wildcarding is allowed.

Group Message Filter

g/*call1***/***call2*…

Allow "message" packets with any of the listed addressees. Wildcarding is allowed.

As the name suggests, this is really intended for "group" bulletins rather than "messages" addressed to a specific station. The APRS protocol spec lists some special prefixes for sending to a group rather than an individual.

- BLN – General Bulletins and Announcements.
- NWS – National Weather Service Bulletins.

Example:

```
g/BLN*
```

I'm not happy with this. I think it should filter on the "group name" rather than the entire addressee field. I have bigger fish to fry right now and will get back to this little detail later. (The exact behavior here is subject to change.)

The "i" filter is the preferred method for messages addressed to a specific station

Unproto Filter

u/*unproto1***/***unproto2*…

Allow packets with any of the specified strings in the AX.25 destination field. APRS uses this field in a variety of ways. Most often it is the system type from the tocalls.txt file. For example, to select packets from the Kantronics KPC-3+, version 9.1, use:

```
u/APN391
```

This does not apply to the MIC-E packet types because they use the destination field for part of the position.

Wildcarding is allowed so you could use "`u/APDW*`" to mean any version of Dire Wolf.

Individual Message Filter

i/*time*
i/*time/hops*
i/*time/hops/lat/lon/km*

Allow "messages" for a station heard over the radio in the last '*time*' minutes within the specified distance. Distance can be digipeater hops and/or geographical distance.

Typical time limits might be 30 or 60 minutes. If we haven't heard from a station for that long, it's probably no longer hearing us.

'*hops*' is the number of digipeater hops necessary to hear the message addressee.

If *hops* is not specified, the maximum transmit digipeater hop count, from the `IGTXVIA` configuration will be used. Suppose that we heard three local stations over the radio:

```
W1ABC>APRS,DIGI1,DIGI2:whatever
W2DEF>APRS,DIGI1*,DIGI2:whatever
W3GHI>APRS,DIGI1,DIGI2*:whatever
```

The first station was heard directly. You can tell because there is no "*" in the path.
The second station was heard after one digipeater hop.
The third station was heard after two digipeater hops.

- If we had the filter "i/30/0" we would transmit only messages for the first station because it was heard directly.
- If we had the filter "i/30/1" we would also transmit messages for the second station.
- We would need "i/30/2" or larger to forward messages the third which is 2 digipeater hops away.

This is not entirely reliable because some digipeaters don't maintain the via path to indicate the actual path taken. I have little rant about this, called "APRS Digipeater – Compared to other implementations," in the User Guide. Currently section 9.5.5 but subject to change as new material is added.

You can also specify a physical distance, in kilometers, from a given latitude and longitude. If you only want to use physical distance, and not limit by number of digipeater hops, use a large number for hops as in:

```
i/30/8/42.6/-71.3/50
```

The "i" filter only makes sense when filtering packets from the Server going to RF.

Rate Limiting

We don't want to flood the radio channel. If something goes wrong, this rate limiting will limit the damage. The transmit IGate will limit the number of packets transmitted during 1 minute and 5 minute intervals. If a limit would be exceeded, the packet is dropped and warning is displayed in red. This will give you the default of 6 packets in a 1 minute interval and 10 packets during a 5 minute interval.

```
IGTXLIMIT   6   10
```

SATgate mode

If we hear a packet directly and the same one digipeated, we only send the first to the APRS IS due to duplicate removal. It may be desirable to favor the digipeated packet over the original. For example, you might be more interested in packets that have been forwarded by satellites rather than heard directly. For this situation, we have an option which delays a packet if we hear it directly and the via path is not empty.

When using this option, the digipeated packets will go to the server immediately. The original, heard directly, is sent after a delay, typically 10 seconds. Duplicate removal will drop the original if there is a corresponding digipeated version.

The configuration option for this feature is:

```
SATGATE
```

You can optionally add a delay time in seconds. The default is 10.

You can find more discussion here: http://www.tapr.org/pipermail/aprssig/2016-January/045283.html

Part 4 – More details on how it works

This is probably more than you want to know but it can come in handy when trying to troubleshoot perplexing situations.

Default APRS-IS server behavior

A client connected to a restricted feed port (typically TCP port 14580) of an APRS-IS server with nothing sent as a 'filter' (explained in a later section), will receive nothing from the server immediately. This is to be expected when configuring a TX IGate (equipped with one or more transceivers).

The server feed only becomes active in response to packets received on RF by the IGate and sent to the server. Each of the RF-remote station IDs are kept in a 'heard list' maintained by the server on behalf of each connected & verified client. This table is internal to the server, and cannot be directly retrieved.

There are loosely three types of packets sent from the server, related to the entries in the heard list:

- Messages destined for the heard stations. These may be transmitted on RF, if they are deemed local by the IGate operator.

- Position packets of the sending stations originating those related messages. These may be transmitted, if deemed appropriate by the IGate operator.

- Packets from the destination (or heard) stations, if they are heard elsewhere. These are never to be transmitted on RF, but are only provided to allow the IGate client to make decisions on whether or not to transmit to them. Some client applications use all of these packets to populate the display screen, however. (Note: when a single instance of Dire Wolf is used as concurrently as an IGate and a TNC for some other client application, only packets from the radio are sent to the client application by KISS or the AGW network interface.)

At the time of this writing, there are two different implementations of APRS-IS server software, and each behaves differently with regards to the packets sent from the server originated by stations in the 'heard list'. Be aware that you could get either one when using the rotate addresses.

If there is a desire to limit the data stream incoming from the APRS-IS server (due to bandwidth cost considerations, for example), then packets can be sent to TCP port 8080 (typical) as HTTP, and the originating station IDs will not be inserted into the clients 'heard list'.

It should be made clear that the APRS-IS server only switches packets based on the heard list, and makes no routing decisions, or conclusions as to whether a particular packets should be transmitted to RF. All clients that have heard a station, should expect to see all messages for those heard stations at about the same time. There can be slight variations between APRS-IS server instances at any given time, however.

IGate Station Behavior

Let's take a closer look at the rules from http://www.aprs-is.net/IGateDetails.aspx The original is in Times New Roman type face on the left.

My commentary is indented and italicized.

The following is the basic criteria for what an IGate gates to/from RF.

Gating Criteria from RF to Server

Gate all packets heard on RF to the Internet EXCEPT if any of the following are true:

1. 3rd-party packets (data type }).
   3rd-party packets should have all before and including the data type stripped and then the packet should be processed again starting with step 1 again.

   In my experience, the only place we see 3rd party packets is from an IGate station. The payload packet will have TCPIP in the path so we end up dropping it.

2. Generic queries (data type "?").

   This could generate an overwhelming number of responses.

3. Packets with TCPIP, TCPXX, NOGATE, or RFONLY in the header (last 2 are optional).

   TCPIP means it already traveled via the Internet so we don't want to send it back again.

   NOGATE and RFONLY mean an IGate should not pass it in either direction.

Duplicate removal for RF to IS: *We will often see the same packet multiple times close together due to digipeating. The consensus seems to be that we should just send the first and drop the later duplicates. There is some dissent on this issue. http://www.tapr.org/pipermail/aprssig/2016-July/045907.html There could be some value to sending them all to provide information about digipeater paths.*

If you feel strongly about this issue, you can modify the source code to remove the duplicate removal. Look for comments in the function send_packet_to_server in source file igate.c.

Gating Criteria from Server to RF

Gate message packets and associated posits to RF if all of the following are true:

1. The receiving station has been heard within range within a predefined time period (range defined as digi hops, distance, or both).

   This is easy to understand. We don't want waste effort sending the message unless we think the "addressee" station might be listening.

2. The sending station has not been heard via RF within a predefined time period (packets gated from the Internet by other stations are excluded from this test).

   This is the part I'm not so sure about.

   I guess the intention is that if the sender can be heard over RF, then the addressee might hear the sender without the help of Igate stations. Suppose the sender was 1 digipeater hop to the west and the addressee was 1 digipeater hop to the east.

   I can communicate with each of them with 1 digipeater hop but for them to reach each other, they might need 3 hops and using that many is generally frowned upon and rare.

   I'm tempted to take out this test entirely. "Messages" are very rare and a tiny drop in the bucket. "Messages" have value, unlike most of the redundant repetitive positions of stationary objects. In this case I think we should lean in the direction of get it through rather than being too aggressive about dropping it.

   Maybe we could compromise here and say the sender must have been heard directly. It sent the message currently being processed so we must have heard it very recently, i.e. in the past minute, rather than the usual 30 or 60 minutes for the addressee.

3. The sending station does not have TCPXX, NOGATE, or RFONLY in the header.

   Where is an official definition of these? I found only various scattered mentions in different discussion groups.

   TCPXX means it came from an unverified client, probably not a ham operator so we don't want to put this on the air.

   NOGATE and RFONLY mean an IGate should not pass it in either direction.

These are hardcoded for anything going through the transmit part of the IGate. They are not specific to "messages."

We usually see TCPIP in the path.

4. The receiving station has not been heard via the Internet within a predefined time period. A station is said to be heard via the Internet if packets from the station contain TCPIP* or TCPXX* in the header or if gated (3rd-party) packets are seen on RF gated by the station and containing TCPIP or TCPXX in the 3rd-party header (in other words, the station is seen on RF as being an IGate).

The intention here is that if a station is known to be an IGate, it shouldn't need any help from another IGate station.

This condition has not been implemented.

If the addressee is an IGate, the servers would know that. Why are they sending the message to me? Again, messages are very rare, and have value so I think the emphasis should be on trying to get the message delivered rather than thinking of situations where it might be able to get there some other way.

Duplicate removal for IS to RF: *Originally my thinking was that we should have duplicate removal in this direction as well. Maybe that is not such a good idea. Hopefully the Servers are smart enough not to send too much useless redundant information too quickly.*

Where it could be an issue is with "message" retries. Suppose there is no acknowledgement to a message and the sender tries again in less than 30 seconds. We would not want to drop that retry.

The transmit rate limiting is also increased for "messages" so they will get through even if the limit has been reached with other types of packets.

Gate all packets to RF based on criteria set by the sysop (such as callsign, object name, etc.).

The IGate station would send server side filters to request additional subscriptions. Note that each only increases the amount sent to the IGate station.

The IGate station would want to apply additional client side filtering to control exactly what it wants to allow.

Part 5 - IGate Troubleshooting Options

To see more of what is going on behind the scenes, use these debugging options on the command line:

`-di`	*Show packets being sent to Server.* *Normally almost everything from the radio is sent to the server so this would be redundant and only add clutter so the default is off.*
`-dii`	Also, *show information about duplicate removal.* *Again, not real interesting and of use only for verifying that duplicate removal is working properly.*
`-dm`	*Display the monitoring stations heard list each time it is updated.*
`-df`	*Display client side packet filter final result.*
`-dff`	*Also show results of individual filter specifications.*
`-dfff`	*Also show results of the logical operators: & | !*

Receive IGate Example

Here is an example of SATgate mode. We hear a station directly. The packet gets put into a waiting area instead of being sent to the server immediately. Verbose output is enabled with the "-dii" command line option.

```
N3LEE-15 audio level = 26(13/8)    [NONE]    _|||||:::
[0.4] N3LEE-15>TRTS5Q,WIDE1-1,WIDE2-1:`cHDl!'>/"6!}/TinyTrak4 <0x7f><0x7f><0x7f><0
x7f><0x7f>
MIC-E, normal car (side view), Unknown manufacturer, Off Duty
N 42 43.5100, W 071 44.4000, 0 MPH, course 111, alt 630 ft
/TinyTrak4 <0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f
Rx IGate: SATgate mode, delay packet heard directly.
```

We hear it via some digipeater. It's not a duplicate so it is sent to the server.
Note that [rx>ig] means transfer from receiver to IGate. Magenta indicates outgoing direction.

```
Digipeater WIDE1 (probably N3LEE-10) audio level = 16(8/5)    [NONE]    __:||||__
[0.4] N3LEE-15>TRTS5Q,N3LEE-10,WIDE1*,WIDE2-1:`cHDl!'>/"6!}/TinyTrak4 <0x7f><0x7f><0
<0x7f><0x7f><0x7f>
MIC-E, normal car (side view), Unknown manufacturer, Off Duty
N 42 43.5100, W 071 44.4000, 0 MPH, course 111, alt 630 ft
/TinyTrak4 <0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><
rx_to_ig_allow? 4105 "N3LEE-15>TRTS5Q:`cHDl!'>/"6!}/TinyTrak4 □□□□□□□□□□□□□□□□□□□□"
rx_to_ig_allow? YES
[rx>ig] N3LEE-15>TRTS5Q,N3LEE-10,WIDE1*,WIDE2-1,qAR,WB2OSZ-15:`cHDl!'>/"6!}/TinyTrak
x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f>
```

We hear it from another digipeater. It is dropped as a duplicate.

```
Digipeater W1MRA audio level = 20(10/8)   [NONE]   ||||||___
[0.2] N3LEE-15>TRTS5Q,N3LEE-10,WIDE1,W1MRA*,WIDE2:`cHDl!'>/"6!}/TinyTrak4 <0x7
x7f><0x7f><0x7f><0x7f>
MIC-E, normal car (side view), Unknown manufacturer, Off Duty
N 42 43.5100, W 071 44.4000, 0 MPH, course 111, alt 630 ft
/TinyTrak4 <0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><0x7f><
rx_to_ig_allow? 4105 "N3LEE-15>TRTS5Q:`cHDl!'>/"6!}/TinyTrak4 □□□□□□□□□□□□□□□□□
rx_to_ig_allow? NO. Seen 1 seconds ago.
Rx IGate: Drop duplicate of same packet seen recently.
```

About 10 seconds after it was heard, the original packet is released from the waiting area. It is a duplicate so it is dropped.

```
rx_to_ig_allow? 4105 "N3LEE-15>TRTS5Q:`cHDl!'>/"6!}/TinyTrak4 □□□□□□□□□□□□□□□□□□"
rx_to_ig_allow? NO. Seen 9 seconds ago.
Rx IGate: Drop duplicate of same packet seen recently.
```

Heard Station List example

When the **"-dm**" command line debug option is used, the monitoring heard station list is printed each time it is updated. The list is sorted with the most recently heard at the top.

callsign - In this case "heard" means appeared as the AX.25 source address. Something inside of a third party wrapper is not counted.

cnt - Count of number of times this station was heard.

chan - Radio channel where heard. Could be non-zero for system with multiple radios.

hops - Number of digipeater hops. i.e. Number of addresses in the via path that are marked as used.

RF - Hours:minutes since last heard on the radio.

IS - Hours:minutes since last heard via an APRS-IS server. Currently collected but not used.

lat, long - Most recent position of station. Some don't have a position listed because we only heard other packet types from that station.

msp - "Message Sender Position." This is to record that the next position report, from the message sender, should be allowed through. For example if we forward a message from sender A to addressee B, msp for A would be set. The next time a position report, from A comes from the server, that would be forwarded as well.

```
callsign  cnt chan hops    RF      IS    lat     long  msp
N1MJL      66    0    2     0:19    0:00   42.64  -71.49  0
N1NCI-3    11    0    1     0:00     -     43.03  -72.37  0
```

```
N1OLA        6   0   4     0:00      -      44.47  -73.21  0
W1JT        61   0   1     0:00      -      42.15  -70.99  0
VE2PQP      36   0   5     0:10   0:00        -       -    0
W1CNH-5     14   0   2     0:00      -      43.76  -71.46  0
W1OCA        7   0   5     0:01      -      44.22  -70.52  0
KB1TSO      74   0   0     0:01   0:25      42.71  -71.22  0
N3LLO-2     80   0   0     0:01   0:04      43.37  -71.84  0
N3LLO-4     61   0   0     0:10   0:01      42.74  -71.49  0
KB1CHU      77   0   2     0:22   0:01      42.08  -70.69  0
WA1GOV-10   22   0   2     0:02   0:02      41.85  -71.01  0
UNCAN       36   0   0     0:02      -      42.98  -71.59  0
WB2OSZ-5    16   0   2     0:02      -      42.62  -71.35  0
KM4IFK-7    22   0   4     0:03      -      43.08  -70.75  0
VE2PQP-1    28   0   7     0:03      -      45.54  -73.34  0
WA1HOG-7    13   0   4     0:03      -      42.76  -72.00  0
N3LLO-3     85   0   0     0:13   0:04      42.77  -71.90  0
N1OHZ       55   0   1     0:22   0:04      42.24  -71.85  0
N4IK-10      1   0   3     0:04      -      44.43  -71.86  0
N1ESA       42   0   3     0:04      -      41.48  -72.13  0
KC1BOS-10   63   0   0     0:05   0:07      43.06  -71.31  0
W1MHL       28   0   0     0:06      -      42.39  -71.25  0
W1TG-1      60   0   2     0:06   0:06      42.94  -70.82  0
N1WJO      100   0   4     0:06   0:06      44.20  -70.55  0
K1RTA-1     55   0   4     0:06   0:11      41.90  -69.99  0
K1DF-7      18   0   2     0:07      -      43.17  -73.12  0
N1RCW-2    149   0   2     0:17   0:07      41.70  -70.30  0
N8VIM       49   0   0     0:07      -      42.68  -71.57  0
W1PID       17   0   4     0:08      -        -       -    0
KD1KE       47   0   5     0:16   0:08      44.48  -69.34  0
VA2DGE-14    4   0   4     0:08      -      45.80  -74.00  0
KB5VP-3     38   0   4     0:08      -      43.48  -72.43  0
N1NCB-14    12   0   4     0:09      -      43.81  -69.94  0
N1MD-3      19   0   2     0:09   0:19      41.87  -71.92  0
AB1OC-10    85   0   2     0:09   0:09      42.71  -71.59  0
N1MD-1      10   0   2     0:09   4:39      41.87  -71.92  0
W2DAN-15    17   0   1     0:09   0:10      41.65  -71.19  0
```

… (list truncated here) …

Transmit IGate example

(1) This was received by the Gateway, and destined for transmission, as indicated by [ig>tx].

```
[ig>tx] KL2KL-5>APOA00,TCPIP*,qAC,N6NAR::KL2KL-7  :great{AF}
```

(2) Debug output for "FILTER IG 0 i/60/8/42.62/-71.35/50"
This is enabled with –df command line option.

```
Was message addressee KL2KL-7 heard in the past 60 minutes, with 8 or fewer digipeater hops, and
within 50.0 km of 42.62 -71.35?
Yes, KL2KL-7 last heard over radio 0 minutes ago, 1 digipeater hops.  Last location 47.9 km away.
Was message source KL2KL-5 heard in the past 1 minutes, with 0 or fewer digipeater hops?
No, we have not heard KL2KL-5 over the radio.
```

(3) We transmit the message wrapped as third party traffic. Other stations do too.

```
[0L] WB2OSZ-14>APDW14,WIDE1-1,WIDE2-1:}KL2KL-5>APOA00,TCPIP,WB2OSZ-14*::KL2KL-7  :great{AF}
[0.4] KB1TSO>APWW10,WIDE2-2:}KL2KL-5>APOA00,TCPIP,KB1TSO*::KL2KL-7  :great{AF}
```

```
[0.3] N1OHZ>APWW10,W1MRA*,WIDE2:}KL2KL-5>APOA00,TCPIP,N1OHZ*::KL2KL-7  :great{AF}
[0.3] N3LLO-3>APRS:}KL2KL-5>APOA00,TCPIP,N3LLO-3*::KL2KL-7  :great{AF}
```

(4) Another example where the message was not forwarded because it did not fit into the distance filter. Another IGate did transmit it because the message met its criteria.

```
[ig>tx] WXBOT>APRS,qAS,KI6WJP::WA1GOV-10:Taunton MA. Hazardous Weather Outlook,
Tonight,Increasing Clouds Lo
Was message addressee WA1GOV-10 heard in the past 60 minutes, with 8 or fewer digipeater hops,
and within 50.0 km of 42.62 -71.35?
No, WA1GOV-10 was 89.7 km away although it was 2 digipeater hops 21 minutes ago.
[0.4] KB1TSO>APWW10,WIDE2-2:}WXBOT>APRS,TCPIP,KB1TSO*::WA1GOV-10:Taunton MA. Hazardous Weather
Outlook, Tonight,Increasing Clouds Lo
```