APRS Digipeaters

John Langner, WB2OSZ
Rough Draft - August 2022

Amateur radio repeaters retransmit signals from other stations to increase their range.

Analog voice repeaters listen on one frequency and simultaneously retransmit the same signal on a different frequency.

AX.25 digital repeaters (digipeaters) use a "store and forward" approach. A packet is received, examined, then possibly modified and retransmitted. Usually it is retransmitted on the same radio channel but it is also possible for a multi-port digipeater to link multiple radio channels. Packets received on one channel can be retransmitted on different channels.

The ***APRS Protocol Reference 1.01*** doesn't describe how a digipeater should work. As a result, we find many implementations that don't behave exactly the same. Newer implementations have found different ways to overcome weaknesses in the traditional TNCs.

1 The Standard "TNC-2" Monitoring Format

First we need to understand what the standard display format is telling us so we can understand the examples and actual packets seen over the air.

source > destination : information
source > destination , digipeater1 : information
source > destination , digipeater1, … , digipeater8 : information

The standard display format begins with 2 or more addresses:

Source address - Originating station.
Most often a ham radio callsign but it could be a tactical callsign. It can optionally be followed by a "-" and a number, maximum 15, called the substation identifier (SSID), to allow up to 16 stations to be operated with the same callsign.

Destination address - In traditional connected mode packet, this would be a specific station. In APRS, this is used in several different ways. We can ignore it for this discussion.

Via path - Up to 8 addresses for path that the packet has taken already and where it might go.

When a digipeater address is followed by "*", that one, and all earlier digipeater addresses, have been used up. They show the path of where the packet has been retransmitted already.

(Behind the scenes, in the AX.25 frame, each digipeater address has a "has been used" H bit to indicate that the address has already been used.)

2 Manual Routing

Suppose I wanted to explicitly route a packet through N2GH digipeater and then W2UB digipeater. The original packet would look like this, with 2 specific digipeaters listed. Notice that there is no "*" so we are hearing the original (source) station:

```
WB2OSZ>APRS,N2GH,W2UB:something
```

N2GH recognizes its name, in the **first unused** digipeater position, and retransmits the packet. The result would look like this:

```
WB2OSZ>APRS,N2GH*,W2UB:something
```

The N2GH digipeater sets the "has been used" flag (the AX.25 "H" bit), on the address to indicate that it has been used up and won't be considered for any future digipeating decisions. When you see "*" after a digipeater name, you know that you are hearing the transmission from there.

The same thing happens again. W2UB sees its name in the **first unused** digipeater position and retransmits the packet. You see the "*" after the callsign so you know that you are hearing that station.

```
WB2OSZ>APRS,N2GH,W2UB*:something
```

All of the digipeater addresses have been used up and this packet can't be retransmitted again.

Some software might display it like this with two "*" characters.

```
WB2OSZ>APRS,N2GH*,W2UB*:something          ← Wrong
```

It might seem more intuitive (both addresses have been used) but **it is wrong**. It does not conform to the rules of the standard monitoring format, where only the **last used digipeater** is marked with "*" and it is implied that earlier addresses have been used up. There should never be a used address after an unused address.

This is what you know if everyone is well behaved:

- The packet originally came from `WB2OSZ`.

- It was retransmitted by `N2GH`. (Therefore `N2GH` can hear `WB2OSZ`.)
- It was retransmitted by `W2UB`. (Therefore `W2UB` can hear `N2GH`.)
- You are hearing the transmission from `W2UB`. (It is followed by "*".)

From the AX.25 protocol specification:

> *"The destination station can determine the route the frame took to reach it by examining the address field and use this path to return frames."*

The second part might be true in theory but not always in practice. You could have a case where station X can hear station Y but Y can't hear X so the same reverse path won't work.

As we will see later, some implementations are not well behaved so we really don't know where the packet travelled. Either they don't identify themselves or leave useless junk in the path which creates an ambiguous situation.

3 What Gets Repeated?

Clearly a digipeater should not retransmit everything it hears. If it did that, anything that was heard on the channel would keep bouncing back and forth between all of the available digipeaters. First the sender needs to construct a suitable via path. The digipeaters need to have a suitable configuration specifying how they should behave.

Using specific station names is usually not very satisfactory. What digipeaters are available nearby? Who can hear me? Who can hear the digipeaters that hear me? What happens if my favorite digipeater is not available? What if I'm traveling and don't know what is in the vicinity?

"Aliases" can allow digipeaters to respond to additional names besides their own callsign. Multiple stations can respond to the same alias. For example, the local Emergency Operations Center (EOC) might respond to the alias "EOC" so you don't have to remember the exact callsign used. A digipeater on the top of Mount Washington might respond to the alias "MTWASH". Sometimes mobile digipeaters will respond to the alias "TEST."

The 20th Century hardware TNCs did not allow much flexibility, allowing at most 4 aliases. For example,

```
UIDIGI ON EOC,TEST
```

If I was to transmit something like this,

```
WB2OSZ>APRS,EOC:something
```

It might be retransmitted as:

```
WB2OSZ>APRS,KB1MKZ*:something        ← correct
```

The alias is always **replaced** by the callsign of the digipeater. It should never be retransmitted like this, with the alias marked as being used:

```
WB2OSZ>APRS,EOC*:something          ← wrong
```

Aliases are always replaced by the callsign of the digipeater. This gets back to the rule mentioned earlier that the used addresses should show you the path that the packet has taken, on its way from the source station, to you.

Two different old TNC manuals mentioned nothing about duplicate suppression for UIDIGI, as they do for UITRACE, so they might clutter up the radio channel with unnecessary duplicates of the same thing. Duplicate suppression will be discussed later.

4 The New n-N Paradigm

Most of the time, we don't want to manually specify a sequence of specific digipeaters. Instead we normally want to specify any digipeater or any with some property such as geographic region.

In the early days of APRS, digipeater aliases of "RELAY" and "WIDE" were used. This has been obsolete, since around 2004, and all uses of them should have been removed long ago. So let's not talk about them any more since it will only cause confusion.

Fixing the 144.39 APRS Network
The New n-N Paradigm
http://www.aprs.org/fix14439.html

The currently accepted method is to specify classes of APRS digipeaters in the generic form *XXXn-N*.

XXX	The prefix, up to 5 characters. Usually this is "WIDE" but others are allowed for geographical regions, special events, or other uses. For example, "MA" might be used for Massachusetts. "TEST" is sometimes enabled for mobile digipeater stations.
n	Usually 1 for a local "fill-in" short range digipeater. 2 is normally for a good location with long range. Theoretically numbers up to 7 can be used, but in practice only 1 and 2 are normally used.
N	The remaining number is maximum number of times the frame should be digipeated, also known as the hop count. Initially it is in the range of 1 thru 7. This is decremented until it reaches 0 and there is no more digipeating.

The 20th Century TNC doesn't allow much flexibility here.

```
UITRACE   WIDE,30
```

This means it will respond to an address composed of

The characters “WIDE”.
A digit in the range of 1 through 7.
An SSID in range of 1 through 7.

This is not very customizable. It will match 49 different combinations such WIDE1-1, WIDE1-7, WIDE2-2, WIDE7-5, etc.

There doesn’t seem to be a way to specify more than a single generic alias or to specify the number before the SSID.

The traditional digipeater configuration commands are inadequate for APRS after 2004. Newer implementations have come up with different approaches for more flexibility in configuring behavior.

5 Packets Gone Wild

If we are not careful, digipeating could get completely out of control. An original packet might get heard by several digipeaters and retransmitted by each of them. A larger growing ring of digipeaters hears multiple others and retransmits what it heard from each. The original station might hear its own packet resent by other stations and retransmit it, forming a loop.

There are a few things we can do to bring the situation under control.

5.1 Decreasing Hop Count

The originating station specifies the maximum number of times that a packet can be retransmitted. For example, we might start with:

WB2OSZ>XXXX,WIDE1-3:whatever

A digipeater configured to repeat for the pattern WIDE1-N would decrement the hop count, to 2, and insert its own call sign:

WB2OSZ>XXXX,WW1ABC*,WIDE1-2:whatever

Another digipeater would decrement the hop count to 1 and send:

WB2OSZ>XXXX,WW1ABC,WW2DEF*,WIDE1-1:whatever

Notice how there is only a single “*” after the last digipeater name. The next digipeater would decrement the hop count to send one of these two forms:

WB2OSZ>XXXX,WW1ABC,WW2DEF,W3GHI*:whatever

In the first example, the WIDE1-0 has been all used up so we can discard it and make the packet smaller. Some implementations leave the generic alias after it has been used up:

WB2OSZ>XXXX,WW1ABC,WW2DEF,W3GHI,WIDE1*:whatever

This creates an ambiguous situation. Did we hear W3GHI or did we hear some other station that did not identify itself? I think the first approach makes a lot more sense and follows the digipeater path intent in the AX.25 protocol specification:

> *"The destination station can determine the route the frame took to reach it by examining the address field and use this path to return frames."*

Anyhow, the digipeater addresses have been all used up so this packet can't be digipeated again.

5.2 Delay from clear channel to start of transmit

Usually, when we are preparing to transmit, we wait for a clear channel, and then wait a random amount of time to minimize the chances of transmitting at the same time as someone else. This is based on the SLOTTIME and PERSIST parameters.

For the default values, we have delays with the following probabilities:

Delay, mSec	**Probability**	
100	.25	= 25%
200	.75 * .25	= 19%
300	.75 * .75 * .25	= 14%
400	.75 * .75 * .75 * .25	= 11%
500	.75 * .75 * .75 * .75 * .25	= 8%
600	.75 * .75 * .75 * .75 * .75 * .25	= 6%
700	.75 * .75 * .75 * .75 * .75 * .75 * .25	= 4%
etc.	...	

If a signal is detected during this random wait time, we go back to the top and start over.

In the case of digipeating, we start transmitting **immediately** when the channel becomes clear. Rather than trying to avoid a collision, digipeaters immediately start transmitting at the same time on top of each other. The AX.25 protocol specification refers to these as "expedited" frames. Due to the FM capture effect, the strongest signal should win.

Traditional TNCs often have a parameter, called **UIDWAIT**, which needs to be **off** for this to work properly.

Using a KISS TNC for a digipeater is a bad idea because the KISS protocol has no way to apply "nowait" for individual packets. If you use a KISS TNC, and a separate application, for a digipeater, it will wait until after the well behaved digipeaters transmit at the same time, and possibly more poorly behaving digipeaters, and then cause extra unnecessary congestion.

5.3 Duplicate Suppression

The third part of the solution is to avoid sending duplicates within a certain amount of time, usually 30 seconds. A digipeater must remember everything it transmits and not transmit the same thing within 30 seconds. The comparison involves only the source, destination, and information part. In other words, the varying digipeater path is ignored when checking to see if two packets are the same.

A digipeater should not repeat a packet with its own callsign in the SOURCE field because that could result in a loop.

6 Digipeater Algorithm Summary

Digipeater configuration contains:

- Its own callsign.
- An optional set of aliases, additional names to which it will respond.
- An optional set of "generic" addresses where the SSID is a remaining hop count. Typically this will be WIDE1-N for a short range digipeater. It should be possible to had at least one more for geographical region or other special purpose.

6.1 Decide if it is eligible for repeating:

(a) Look for the first "unused" address in the digipeater addresses. Return NO if none found.

(b) Is the source my station address? If so, return NO.

(c) If the first unused digipeater address is my station address, return YES.

(d) If the first unused digipeater address is one of my aliases, return YES.

(e) If the first unused digipeater is of some generic form ***XXXXn-N*** matching a rule in my configuration, return YES.

The legacy UITRACE allows only a single prefix such as “WIDE” which would match “WIDE”, 1 thru 7, “-“, 1 thru 7.
Some later implementations offer greater flexibility including multiple prefixes and distinguishing “WIDE1” from “WIDE2”.

(f) Otherwise, return NO.

6.2 Suppress any duplicates.

(a) If we retransmitted this packet recently (typically within 30 seconds) then return NO.

The comparison is based on SOURCE, DESTINATION, and INFORMATION part.
It does not include the digipeater addresses.
Implementations typically use a 16 or 32 bit hash, rather than keeping the entire string, to reduce storage space and compare time.

6.3 If it should be resent and the address is my call or one of my aliases:

(a) Replace the address with my address.
(b) Mark the address as used.
(c) Transmit it.

Or, if it should be resent and of the form XXXXn-N:

(a) If N >= 2, the N value is decremented. The digipeater callsign is inserted before it and marked used if we have less than the limit of 8 addresses.

Example: W9XYZ >APRS,WIDE2-2
Becomes: W9XYZ >APRS,WB2OSZ*,WIDE2-1

(b) If N = 1, we don't want to keep WIDEn-0 in the digipeater list so the generic address is replaced by the digipeater callsignand marked as used.

Example: W9XYZ >APRS,WIDE2-1
Becomes: W9XYZ >APRS,WB2OSZ*

(c) If N = 0, the hop count has been used up and the packet is not digipeated. This is an error condition that we should not encounter. If the address count was all used up, we would expect the has-been- used “H” bit to be set. There is at least one defective implementation out there which produces this.

6.4 Transmit and remember it

(a) Transmit the modified packet, WITHOUT the normal random delay after the channel is clear.

(b) Add to list of recently retransmitted packets along with a timestamp

7 Inconsistent behavior of different implementations

The APRS protocol specification says nothing about how a digipeater should work so it is not surprising that different implementations behave differently.
Based on observations, some implementations ***always insert*** their call rather than replacing when the hop count is all used up. Example:

	Unconditional insert	**Adaptive insert / replace**
Original digipeater path	WIDE1-1,WIDE2-2	WIDE1-1,WIDE2-2
After 1 hop	W1ABC,WIDE1*,WIDE2-2	W1ABC*,WIDE2-2
After 2 hops	W1ABC,WIDE1,W2DEF*,WIDE2-1	W1ABC,W2DEF*,WIDE2-1
After 2 hops	W1ABC,WIDE1,W2DEF,W3GHI,WIDE2*	W1ABC,W2DEF,W3GHI*
Implemented by	KPC-3+	Dire Wolf

The unconditional insert approach has a rather unfortunate consequence. The final packet looks like it was relayed by **five** different digipeaters.

- W1ABC
- Unknown station not implementing tracing.
- W2DEF
- W3GHI
- Unknown station not implementing tracing.

The packet is longer than it needs to be and wastes radio channel capacity.

This also creates an ambiguous situation where we are not sure about the path taken.

Here is another confusing example observed on 145.825 MHz:

```
K4KDR-6>CQ,PSAT,ARISS*::NB3T     :Hey Mal!!!
N2UFM-1>APWW10,PSAT,ARISS*::ALL      :Hello de N2UFM via PSAT
K0KOC-7>3Y2S1Y,PSAT,ARISS*:'i4wl #/]=
```

It looks like a few people are carefully timing their transmissions, and setting the via path, to go through two space digipeaters. Source station → PSAT → ARISS → heard on Earth.

Why don't we hear any of them, after the first hop, like this?

```
K4KDR-6>CQ,PSAT*,ARISS::NB3T     :Hey Mal!!!
N2UFM-1>APWW10,PSAT*,ARISS::ALL      :Hello de N2UFM via PSAT
K0KOC-7>3Y2S1Y,PSAT*,ARISS:'i4wl #/]=
```

After a little research, I was disappointed to learn that PSAT uses the unconditional insertion approach, creating this confusing situation.

I would argue that it violates the AX.25 protocol specification section 3.12.5:

> *"The destination station can determine the route the frame took to reach it by examining the address field…"*

The via path part of the address field, up through the address marked with "*" should contain the path that the packet has traveled.

Here is a real example that demonstrates the different cases and something new and unexpected.

We start off with the original packet. There is no "*" in the header, so we are hearing the originating station.

```
N1TBN-9 audio level = 9   [NONE]
[0] N1TBN-9>T2SU5U,WIDE1-1,WIDE2-1:`c.<m>Lk/]"4G}449.075MHz=<0x0d>
MIC-E, truck, Kenwood TM-D710, In Service
N 42 35.5500, W 071 18.3200, 15 MPH, course 48, alt 157 ft, 449.075MHz
```

Next we see the same packet (below) after it was digipeated by WB2OSZ-5 and AB1OC-10. Notice how the original WIDE1-1 was replaced by WB2OSZ-5 because the remaining hop count was all used up.

```
Digipeater WIDE2 audio level = 11   [SINGLE]
[0] N1TBN-9>T2SU5U,WB2OSZ-5,AB1OC-10,WIDE2*:`c.<m>Lk/]"4G}449.075MHz=<0x0d>
MIC-E, truck, Kenwood TM-D710, In Service
N 42 35.5500, W 071 18.3200, 15 MPH, course 48, alt 157 ft, 449.075MHz
```

The "*" appears after WIDE2 so that is what the radio is hearing. If we didn't know the earlier history, we wouldn't know whether WIDE2-0 (the -0 is not displayed) was left there by AB1OC-5 or a different later station that did not identify itself.

Here is something totally unexpected. Below we see the packet was digipeated twice and we are hearing W1HML, as indicated by the "*" after it.

```
Digipeater W1MHL audio level = 13   [NONE]
[0] N1TBN-9>T2SU5U,WB2OSZ-5,W1MHL*,WIDE2:`c.<m>Lk/]"4G}449.075MHz=<0x0d>
MIC-E, truck, Kenwood TM-D710, In Service
N 42 35.5500, W 071 18.3200, 15 MPH, course 48, alt 157 ft, 449.075MHz
```

The really strange part is a WIDE2-0, at the end, which is not marked as being used. When the remaining count is reduced to zero, the digipeater should be marked as being used.

Here is another case of the same thing that showed up in the discussion forum. There was a question about why a certain packet was not getting digipeated.

> *Under the usual rules, this would not get digipeated.*
>
> *NA7Q>APX202:MEGLER*,WIDE1,WIDE2-1*
>
> *The Digipeater looks for the first address, in the via path, which has not been used. This would be the one after "*".*

The digipeater decides whether "WIDE1" would be eligible. It is not because the remaining hop count (SSID) is 0.

Seeing "WIDE1" without the "" is rather odd. You would not expect to see this. When the remaining hop count is 0 you would expect the address to be marked as being used. There are two possibilities here.*

(1) The originating station used something like "WIDE1-1,WIDE1,WIDE2-1" which is not a sensible thing to do. or

(2) The first digipeater is not behaving properly. Suppose the originating station used "WIDE1-1,WIDE2-1" which is fairly common. The first digipeater should change it to "MEGLER,WIDE2-1". There are a couple implementations that change it to the form "MEGLER,WIDE1*,WIDE2-1" which is confusing. This looks like it went thru 2 digipeaters and the second one did not identify itself.*

Both "MEGLER,WIDE2-1" and "MEGLER,WIDE1*,WIDE2-1" would be eligible for digipeating.*

Mystery solved! Both stations, exhibiting the unexpected behavior, report software type APN382 which translates to KPC-3 or KPC-3+, ROM version 8.2 from 9/1997. This is a known bug, mentioned here: http://www.aprs.org/kpc3/kpc3+82WIDEn.txt

> *When packets were decremented to n-0, the path was not marked as being used up.*

Maybe it is time to let go, of the 20 year old software, and use something more modern that works properly.

In version 1.0, we start to list the **possible** actual station heard when "*" is after something of the form WIDEn-0. Example:

```
Digipeater WIDE2 (probably AB1OC-10) audio level = 10   [NONE]
[0] KB1DDC>TR3R8X,W1XM,WIDE1,AB1OC-10,WIDE2*:'c&*l <0x1c>-/]"3r}<0x0d>
MIC-E, House, Kenwood TM-D700, En Route
N 42 32.8800, W 071 10.1400, 0 MPH, alt 0 ft
```

Of course, this is just a guess and could be wrong.

8 Other, less common options

8.1 Preemptive Digipeating

Normally the digipeater function looks only at the first unused item in the digipeater list. The preemptive option allows processing of any unused field, not just the first one, if my call or an alias matches. Note that the option does not apply to the "generic XXXXn-N" specification.

Example: The received packet contains these digipeaters:

CITYA*, CITYB, CITYC, CITYD, CITYE

The first one has already been used. My alias list includes CITYD.

Normally, this would not be retransmitted because CITYB is not in the alias list. When the preemptive option is selected, "CITYD" is matched even though it is not the first unused. As you would expect, CITYD is replaced by my call before retransmission. What happens to CITYB and CITYC? That depends on the option specified:

- DROP – All prior path data is lost. (misleading, bad)
- MARK – Prior path data is marked as being used. (misleading, bad)
- TRACE – Prior path data will reflect the actual path taken. (good)

Results, for this example, are summarized below.

Option	**Path after digipeating**	**Comment**
OFF	(none)	No match. Not digipeated.
DROP	WB2OSZ*, CITYE	Erases history before getting here. Gives incorrect impression that original station was heard directly rather than via CITYA. (BAD)
MARK	CITYA, CITYB, CITYC, WB2OSZ*, CITYE	Gives incorrect impression that packet traveled through CITYB and CITYC. (BAD)
TRACE	CITYA, WB2OSZ*, CITYE	Accurate tracing of path used to get here. (GOOD)

The AX.25 protocol specification states:

> *As a frame progresses through a chain of repeaters, each successive repeater will set the H bit in its SSID octet, indicating that the frame has been successfully repeated through it. No other changes to the frame are made (except for the necessary recalculation of the FCS). The destination station can determine the route the frame took to reach it by examining the address field and use this path to return frames.*

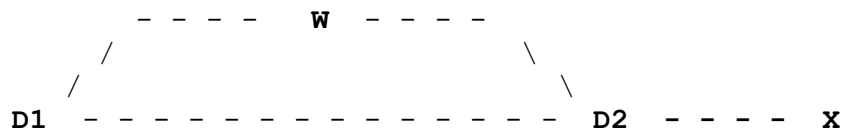
The DROP and MARK options would violate this principle. The used digipeater addresses should accurately reflect the path taken.

8.2 Viscous Digipeating

The normal mode of operation is that all digipeaters will transmit at the same time. Other stations should receive only the strongest one due to the FM capture effect.

Another controversial strategy is to wait a while and retransmit the frame only if you don't hear it from anyone else in a certain amount of time. This is known as "viscous" digipeating.

You can certainly construct examples, where redundant transmissions don't accomplish anything. For example a low power fill-in digi when a high powered mountain top station will overpower everyone else. There are other cases where it will make the situation worse. Consider the following example:



W is a weather station.
X is someone interested in hearing weather information.
D1 and D2 are digipeaters.
The lines indicate who can be heard by whom.
(i.e.
W, D1, and D2 can hear each other.
X and D2 can hear each other.)

Normal digipeater operation:

Station W transmits a packet.
Both D1 and D2 retransmit it at the same time.
X is able to receive it from D1.

Suppose D1 was using "viscous" digipeating:

Station W transmits a packet.
D1 (normal behavior) retransmits it immediately.
D2 (viscous strategy) waits a little while, notices the retransmission by D1, and decides not to resend it.
X is NOT able to receive the weather information.

Here is the last mention of it in APRSSIG in 2013: http://www.tapr.org/pipermail/aprssig/2013-March/041554.html The consensus is that it probably is not a good idea.

As far as I know, "aprx" is the only APRS application that has implemented this.

8.3 Packet Filtering

Modern implementations often have some sort of filtering to allow or deny digipeating of a packet depending on properties such as source address or information type. There was no precedent to follow from the previous Century so there is no standard approach.

Dire Wolf Configuration Differences

Rather than clinging to the past, and mimicking inadequate configuration options from about 40 years ago, Dire Wolf avoids these limitations and allows very flexible configuration options to handle a wide variety of situations.

9 APRS Digipeater Configuration

APRS digipeater configuration is achieved with commands of the form:

DIGIPEAT *from-chan to-chan aliases wide [preemptive]*

where,

from-chan	is the radio channel where the packet is received. Must be a number.
to-chan	is the channel where the packet is to be re-transmitted. Must be a number.
aliases	is an alias pattern for digipeating ONCE. Anything matching this pattern is effectively treated like WIDE1-1. 'MYCALL' for the receiving channel is an implied member of this list.
wide	is the pattern for normal WIDEn-N digipeating where the ssid is decremented.
preemptive	is one of the preemptive digipeating modes: OFF, DROP, MARK, or TRACE. Default is off.

Pattern matching uses "extended regular expressions." Rather than listing all the different possibilities (such as "WIDE3-3,WIDE4-4,WIDE5-5,WIDE6-6,WIDE7-7"), a pattern can be specified such as "^WIDE[34567]-[1-7]$". This means:

^	beginning of call. Without this, leading characters don't need to match and ZWIDE3-3 would end up matching.
WIDE	is an exact literal match of upper case letters W I D E.
[34567]	means ANY ONE of the characters listed.

- is an exact literal match of the "-" character (when not found inside of []).

[1-7] is an alternative form where we have a range of characters rather than listing them all individually.

$ means end of call. Without this, trailing characters don't need to match. As an example, we would end up matching WIDE3-15 besides WIDE3-1.

Google "Extended Regular Expressions" for more information.

(Note: "Connected" mode digipeating uses a different algorithm and is configured with **C**DIGIPEAT.)

As a typical example, you might have a dual port digipeater between the national standard APRS frequency and a local frequency for a special event. You could use the "t/m" filter (described in a later section) to allow only "Message" packets to be forwarded to the special event frequency.

Duplicates are not transmitted if the same thing was transmitted within the DEDUPE number of seconds. The default is

```
DEDUPE  30
```

Duplicate checking is performed by comparing the source, destination, and information part. In other words, the via path is ignored.

Packet filtering can be used to limit what gets retransmitted for each combination of from/to radio channel. For example you might want to transmit only position reports from W2UB when digipeating from channel 0 to 1.

```
FILTER  0  1  t/p & b/W2UB
```

Complete details are in the Packet Filtering section of the User Guide.

10 Digipeater - Typical configuration

Enable digipeating by editing the configuration file (direwolf.conf) and modifying the two lines that look similar to this:

- `MYCALL  NOCALL`

  Obviously, you would want to change this to your own call.
  For example: MYCALL WB2OSZ-5

- `#DIGIPEAT  0  0  ^WIDE[3-7]-[1-7]$  ^WIDE[12]-[12]$`

Remove the "#" character at the beginning of the line. Lines beginning with "#" are comments and they are ignored. This is a good default for general purpose use.

Restart Dire Wolf so it will read the modified configuration file.

What does this all mean?

- The first 0 means the rule applies to packets received on radio channel 0.
- The second 0 means anything matching the rule is transmitted on channel 0.
- Next we aliases that need to match exactly. This gets replaced by MYCALL when digipeateed. It does not apply the rule of decrementing the last digit of WIDEn-n. We use ^WIDE[3-7]-[1-7]$ to "trap" larger values of N as discussed in

  Fixing the 144.39 APRS Network
  The New n-N Paradigm
  http://www.aprs.org/fix14439.html

  If you wanted to process WIDE3-n normally, instead of "trapping" it, you could use this instead:

  ```
  DIGIPEAT 0 0 ^WIDE[4-7]-[1-7]$ ^WIDE[123]-[123]$
  ```

- The final parameter specifies patterns to be processed with the new n-N paradigm if not caught by the aliases. If the last digit is greater than zero it is decremented by 1 when retransmitted.

If you wanted a "fill-n" digi, that responded to only WIDE1-1, you could use this:

```
DIGIPEAT 0 0 ^WIDE1-1$ ^WIDE1-1$
```

11 Digipeater – example 2 – routing between two states.

In this hypothetical example, we are on top of a tall hill between Massachusetts and New Hampshire.

- Radio channel 0: Directional antenna towards MA
- Radio channel 1: Directional antenna towards NH

Each channel does its normal digipeating out to the same channel. Anything with **MA***n-n* in the path should be sent to channel 0 regardless of where it came from.

```
DIGIPEAT 0 0 ^WIDE[3-7]-[1-7]$ ^WIDE[12]-[12]$|^MA[1-7]-[1-7]$
DIGIPEAT 1 0 ^WIDE[3-7]-[1-7]$ ^WIDE[12]-[12]$|^MA[1-7]-[1-7]$
```

Similarly we want anything for NH to be digipeated only to radio channel 1.

```
DIGIPEAT 0 1 ^WIDE[3-7]-[1-7]$ ^WIDE[12]-[12]$|^NH[1-7]-[1-7]$
DIGIPEAT 1 1 ^WIDE[3-7]-[1-7]$ ^WIDE[12]-[12]$|^NH[1-7]-[1-7]$
```

12 The Ultimate APRS Digipeater

http://www.aprs.org/digi-ultimate.html describes the "Ultimate APRS Digipeater" with 3 radios and different rules for forwarding packets from one channel to another. It's no longer necessary to lament that, "such a digipeater does not yet exist." It's available now. Using the digipeating rules described above, and packet filtering, described in the User Guide, you can do all of this and more.