Packet Radio Relay Stations
Digi-peaters vs Nodes
By Steve Wuelfing K8BZ

In my opinion the single most misunderstood aspect of packet radio is the difference between a digi-peater and a node. Both digi-peaters and Nodes are used to relay signals between distant stations that could not communicate directly due to distance or obstacles. But the way they do it is very different. Depending on the specific circumstances involved one method may provide much better results than the other. In order to have a meaningful understanding of the difference between the two methods some technical background is required.

Packet Radio is a digital communication mode that can employ error checking methods similar to that used by computers to verify the accuracy of data transferred between computer components. If the receiving component detects that the received data contains an error, it doesnôt ñself correctò the data. Itôs simply requests that the data be re-sent until no error is detected. Computer systems and packet radio have their own method or protocol for error checking. Early computers used a protocol known as X.25. A group of technically adept hams in and around Tucson, AZ modified this protocol for use on amateur radio and it became known as AX.25. AX.25 protocol is employed between two packet stations that are ñconnectedò via radio to exchange information between them while employing error detecting techniques.

When using AX.25 protocol in packet radio communications, the transmitting equipment (TNC) constructs a ñframeò or packet of information that is subdivided into the following fields:

- Addressing information (callsigns of the transmitting and receiving stations and any digi-peaters in the path)
- Control Information that identified the type of packet (information packets that contain message information, or administrative packets that send connect or disconnect requests, packet acknowledgements, packet rejects)
- Information field, If the packet is not an administrative packet, this is where the message information resides
- Field Check Sequence (FCS) frame. This is where the error checking is accomplished. This is a 16 bit number that the transmitting station calculates based on the contents of the Information field. The receiving station will also calculate what the FCS should be based on the received I field. If the receiving stationôs calculated FCS matches the transmitting stations calculated FCS then an Acknowledgement administrative frame (ACK), or a Receive Ready frame (RR) is set back to the transmitting station to let it know that the packet was received accurately. If the two FCSôs do not match then the receiving station will ignore the frame. If the transmitting station does not receive an ACK or an RR frame from the receiving station it will continue to retransmit the same packet up to the limit set by the tncôs RETRY command. By default a tnc will attempt 10 retries. If an ACK or a RR frame is not received after the 10^{th} retransmission of a frame

then the sending station will assume the path between the stations is lost and it will disconnect.
- An additional field known as the Protocol Identifier Field (PID) is used to identify the type of network layer protocol (if any) is used. This field is absent in Un-numbered or unconnected packet frames.

It is also possible to send Unconnected or Un-numbered Information packets when you are transmitting information that is not addressed to a specific station. Examples would be Automatic Position Reporting (APRS) data, beacons, calling CQ, or when several stations are involved in a ñround tableò discussion and take turns transmitting to the group. In these unconnected packet transmissions no error checking occurs. The packets are displayed, as received, including errors if any, from the transmitted station.

The above is the long description. The Readerôs Digest short version is as follows:

Connected packets use error checking; unconnected packets do not.

What does this have to do with Digi-peaters and Nodes?

If two stations desiring to communication via packet are so far apart that they need, lets say 3 relay stations between them they may opt to use these relay stations as digi-peaters or they may use them as Nodes instead, if those stations have their node function enabled. I will describe how both examples would be used.

A three hop path via digi-peaters
Lets say W8ABC needs to communicate with W8XYZ and needs to use ham stations A, B, and C as digiôs to relay the messages between them. W8ABC would send a packet that would be received Digi A; Digi A would send the packet to Digi B; Digi B would send the packet to Digi C; Digi C would send the packet to W8XYZ. Then W8XYZ would compare its calculated FCS with the received FCS. If the two numbers agreed, W8XYZ would send an ACK back to W8ABC in reverse order through the same digiôs. If the FCSôs did not agree then W8XYZ would ignore the packet and wait for a re-transmission of the same packet.

If and when the ACK was received from W8XYZ, then W8ABC would send the next packet. During the above example, no error checking is done by any of the digiôs. The packet was just re-transmitted as received along with any errors that may have been encounters along the way. The only error checking that takes place is when the packet finally reaches W8XYZ. Also, No subsequent packets are sent by W8ABC until each ACK is received. Subsequently, there are lots of opportunities for errors or lost packets. If W8ABC did not received an ACK it would re-send the same packet up to 10 times before ñretrying outò and disconnecting. The amount of time it waits for an ACK before re-sending the packet is multiplied by the number of digiôs in the path to allow time for the packet to reach its destination, and for the ACK to come back via the reverse path.

A three hop path via nodes

We will use the same scenario as above but relay stations A, B and C will now be used as Nodes and not Digiôs. Stations who have their node function enabled will assign a unique callsign to the node. In many cases this would be the primary callsign with the Secondary Station Identifier (SSID) of -7. In my case, my node call is K8BZ-7 to distinguish it from my primary station call or my station Bulletin Board call of K8BZ-1.

Now back to our example of Nodes. In our example W8ABC would first connect to Node A. After connecting to Node A, W8ABC would command Node A to connect to Node B. Then W8ABC would command Node B to connect to Node C, and then he would command Node C to connect to W8XYZ. After all these connections were made the two stations could exchange packets as normal. The difference here is that error checking occurs between each node and packet station at every step in the path. If errors are detected the packet re-transmission only needs to occur between the two points where the error occurred. The ACK of the packet by W8XYZ does not need to follow the reverse path all the way back to W8ABC, as the packet was error checked and acknowledged at each node in the path to the destination. Furthermore, as soon as the first packet from W8ABC is acknowledged by Node A and passed on to Node B, then W8ABC will send the next packet to Node A. In other words, there will be a continuous line of ñnext packetsò on route to W8XYZ.

The bucket brigade comparison
Another way to visualize the difference between communication via digi-pearter or nodes is to think of fire fighters trying to put out a fire using the old Bucket Brigade method. In the Bucket Brigade method one fireman is at the water well, and another fireman is at the fire. Between them is a line of firemen to pass buckets of water from the well to the fire.

The didi-peater example
If they were fighting the fire by the digi-peater method there would only be one bucket. The fireman at the well would fill the bucket and pass it on to the next fireman and so on until it reached the fire. After the water was thrown on the fire the bucket would be passed back to the well in reverse order, to be refilled and sent back to the fire. If the bucket was dropped and the water spilled along the way, the fireman at the well would only refill another bucket and send it to the fire after he waited long enough for the empty bucket to get back to him. If it didnôt arrive after a sufficient waiting period he would refill a new bucket and send it back down the line

The node example
If the same firemen were fighting the fire by the node method, the fireman at the well would have an unlimited supply of buckets. He would fill the first bucket and send it to the next fireman. He would then fill the next bucket and pass it on as soon as the first bucket was passed one more ñnodeò down the line. If a bucket was spilled anywhere along the path, each fireman would be able to magically re-fill the bucket up to ten times and pass it on. Each fireman along the path would in effect have a bucket of water in his hands waiting for the next fireman in the path to take it from him and pass it on. As soon as the next fireman takes his bucket, he can then get the next bucket thatôs already to be given to him. Unlike the didi-peater example, the ACK administrative packets only need

to be given to the previous packet station or node in the path. The ACKôs donôt need to be sent all the way back to the transmitting station as they are error checked and ACKôs are exchanged in each step along the path.

THE BOTTOM LINE

The default setting for tncôs off the shelf is that up to ten digi-peaters can be used for any CONNECTED path between two stations. In actual practice any more than two digis in any CONNECTED path makes for a very slow, if not completely undependable path between two CONNECTED stations. If you are communicating on a connected path requiring more than two packet relay stations, you will generally have much better results if you use nodes and not digis. The practical limit on digis for a given path is typically 2 or 3. A five or six node path will still provide reliable communications on a long haul path. For leaving or reading message from a BBS this is very satisfactory performance. If you are communicating keyboard to keyboard with a real person in a five node path you will find that your transmitted and receive packets will come and go at a steady rate. But the other operators responses to your questions will seem delayed as the question makes it way through the nodes in the path and the response makes it way back. It makes for ñherkey jerkeyò conversations but it works.

Un-numbered (AKA Un-connected or Unprotocol UNP) packets

I emphasize CONNECTED paths in the above example because there are many types of Unconnected communications where digis work fine. Beacons, ID packets, APRS packets and others are all unconnected packets, where nodes always involve connected communications between two specific stations.

If you are sending APRS packets from a mobile installation, the unconnected packets containing the GPS coordinates and any message information will be sent in Un-numbered Information (UI) frames and will almost always involve didi-peaters in an APRS Wide Network. If one packet in a wide digi APRS network is ignored due to errors, another packet will follow shortly and the receiving stations map will be updated at that time with the transmitting stations new location. You may miss a dot on a map from time to time but the overall goal of tracking the transmitting station movement is still maintained.

Advantages and disadvantatges of Digis vs Nodes

Packet TNCôs are ready for digi-peating service right out of the box and require no set up. On the other hand, if you want to allow your packet station to be used as a node it requires setting up the appropriate commands to enable the node function. In a Kantronics TNC, at a minimum you may have to set the following commands at the command propmpt (cmd:)

INTERFACE TERMINAL
MYNODE (CALLSIGN) Usually your primary call and SSID-7 (W8ABC-7)
NUMNODES (NUMBER) The number of stations that can simultaneously use the node (Zero disables the node, but can be up to 10. 2 or 3 would be more realistic)

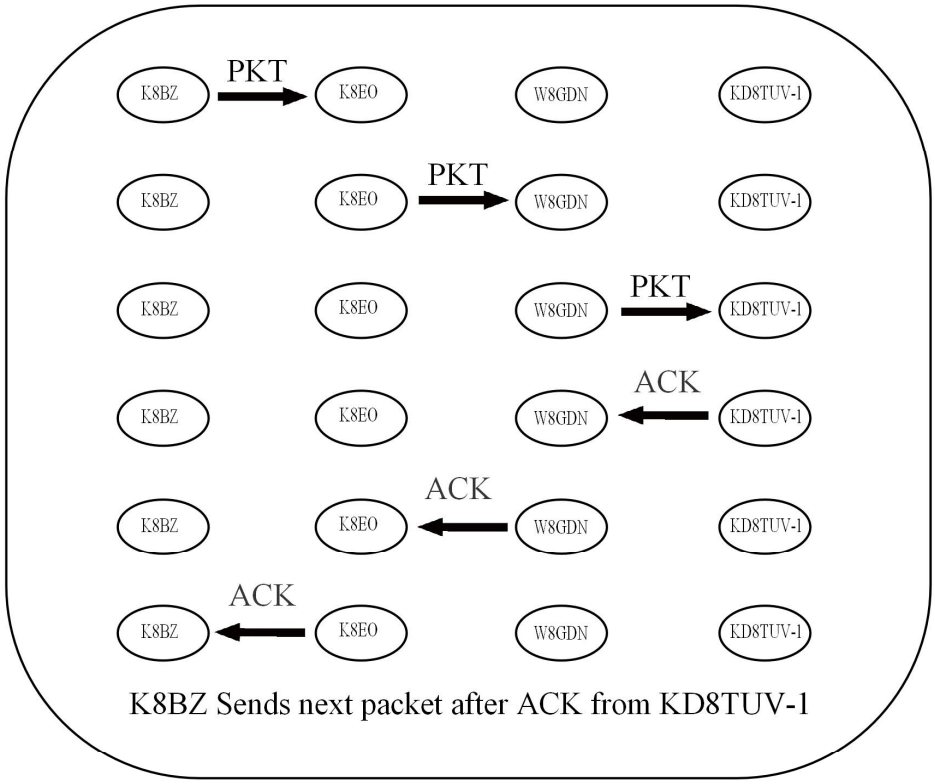
Making a connection via digis can be done in one command. If W8ABC wanted to connect to W8XYZ and use digis K8AA, W8BB, NA8CC as digis in the path the following connect request would be typed at the command prompt:

C W8XYZ via K8AA,W8BB,NA8CC

If W8ABC wanted to make the same connection using the relay stations as nodes he would have to connect to the K8AA node, and from there connect to the W8BB node, and then connect to the NA8CC node and finally connect to the W8XYZ station. The initial connect would require a little more effort, but after the connection was made, the path would be much more reliable.

Although APRS and general text or message type packet radio both use the packet operating mode they are generally incompatible uses on the same frequency. APRS activity would generally take place on an APRS specific frequency (144.390 in this area) and Keyboard QSOôs, BBS operations and general packet radio communications will generally be on another frequency (145.090 Mhz locally).

Digi operation. For K8BZ to connect to KD8TUV-1 via digis use the connect command:
cmd: c kd8tuv-1 via k8eo, w8gdn



To connect via nodes, connect to each node individually until the Destination is reached. Then the packets will flow as follows: