Net/Rom Node Information for the Node Sysop

This is a series designed to help node operators understand some of the more arcane commands associated with the **TheNet X-1JR4** node firmware that's in wide use throughout our state. It's a work currently in progress, so as of now, it is incomplete.

It's meant to be a plain-American-English way for node operators to learn about a very complex system. The original documentation is good, but was written by folks from other countries whose English usage is a tad different than ours. Some people might find that a little difficult to weed through, and that's why I wrote this series.

As each part is completed, it will appear in two places - here and in the Badger State Smoke Signals . It'll appear here first, of course, as my publication deadline is on the 19th of each month.

Here's the Table of Contents:

Part One - Introduction and the start of a discussion of the complex **ACL** Command

Part Two - Continuing and concluding the discussion of the **ACL** Command

Part Three - The **ADC** Command

Part Four - The **ALIAS** and **ARP** Commands

Part Five - A look at **ARPTIMER** and **AUDIT**

Part Six - The **BBS**, **BBSALIAS**, **BRATE** and **BTEXT** Commands

Part Seven - Discussing **CALIBRATE**, **CLOSEDOWN**, **CTEXT**, **DXCALIAS** and **DXCLUSTER**

Part Eight - How to configure **HOST**, **HOSTALIAS**, **INFO**, and **IPADDRESS**

Part Nine - The **IPBROADCAST** and **IPROUTE** commands

Part Ten - Discussing the **IPSTATS**, **L3MHEARD**, and **LINKS** commands

Part 11 - About the **MANAGER**, **METER**, **MHEARD** and **MODE** commands

Part 12 - Covers the **MTU**, **NODE** and **PARMS** commands

Part 13 - Dealing with the **RESET**, **ROUTES** and **STATS** commands

Part 14 - We finish the series with the **SYSOP**, **TALK**, **UI** and **USERS** commands.

Net/Rom Node Information for the Sysop - Part One

by Andy Nemec, KB9ALN

Introduction

This series is designed to help node operators understand the more complex aspects of node configuration and operation, perhaps explained in an easy-to-understand fashion. In addition to the explanations, you will find some WAPR recommendations on these more arcane settings so that you will find interfacing to the network a little easier.

Our discussion is centered around the popular TheNet X-1J series of nodes, which are in wide use throughout Wisconsin. It's assumed you know some of the basics of node operation, such as what an alias is. We won't discuss much of the security aspects of node operation, just some general security concepts. While this series is not specifically aimed at the node user, it can be a good resource for the casual user who wants to know more about nodes. That said, we'll begin (and continue) alphabetically.

The first and second installments deal with one of the least understood node configuration commands - **ACL**. We'll take two parts to discuss this command as it is so complex and requires a lot of attention to function properly.

What it is

ACL is a mechanism for controlling access to nodes by users or other nodes. Although the X-1J manual does not explain what the letters mean, I call it "Anti-Connect List". This may or may not be correct, one will have to ask the authors of this firmware to be sure.

What it Does

ACL has a list of call-signs and a numerical value associated with each call-sign. This numerical value is checked against two other numerical values, a "**Default value**" and a "**Mask value**". If a match is made, a particular action is taken.

Numerical values are based on "bit values" that correspond to a particular action that you wish the node to take. These bit values, and the action associated with them, are shown in the table below:

0 - Prohibit incoming Level 2 connections

1 - Prohibit outgoing Level 2 connections

2 - Ignore node broacasts from this particular station

3 - Prohibit Level 3 (node packets) operation from this particular station

4 - Prohibit Level 4 incoming connections associated with a particular station

5 - Prohibit Level 4 outgoing connections associated with a particular station

6 - Ignore a particular station's SSID on the ACL list

Now it's time for some additional discussion before we go any further. Forgive me while I delve into this, however, it is important to understand just what these levels mean in order to use ACL effectively.

Level Two connections can be considered the AX.25 connection between a regular user and a node. The ACL command can prohibit a user from making a connection to or from the node. This corresponds to the bits **0** and **1**. **0** prohibits regular users from connecting to the node. **1** will allow a user to connect to a node, but will only allow

connections to other nodes (*not other users*).

Level Three connections are basically network management and relay. Setting the **3** bit will disallow relay of network node packets from or to the corresponding station on the ACL list.

Level Four connections are node-to-node connections. If a node call-sign has the **4** ACL bit set, this call-sign won't be able to connect. If this node call-sign has the **5** bit set for a node on the ACL list, it will be able to connect to your node, but your node will not be able to connect to it.

The remaining bits should be self-explanitory, ignoring node broadcasts and the **SSID** (Secondary Station Identification) bit do pretty much what you might think.

How it works

When the node receives a connect request, or a request to connect to another call-sign, it checks the ACL list for call-sign entries. If the call-sign is present on the list, it takes action based on what ACL value is assigned to the call-sign.

When it comes upon such a call-sign, it first looks at a "**Mask**" value that determines what the node should check for. The Mask value is designed to speed things up by restricting the number of checks that would need to be made for a particular call-sign.

For example, if we only need to ignore nodes broadcasts from a particular node, we would only set bit number **2** in the mask. This way, the computer has to only check for one thing instead of 7 things.

The other check that is made when a station attempts a connect is the "**Default**" value. The Default value is also chosen by the node operator and may be virtually any number. It's best, however, to keep it low, for reasons we will cover in our next part.

Net/Rom Node Information for the Sysop - Part Two

by Andy Nemec, KB9ALN

This series is intended to assist TheNet (Net/Rom) node operators understand the complexities of the X-1J series of nodes used widely throughout Wisconsin. We'll discuss each of the commands intended for node Sysops alphabetically. Part one started with the ACL command, and we continue our discussion in Part Two.

To Re-Cap

ACL is a mechanism for controlling access and functions of the node based on call-signs. This is done based on two numerical values, the "**Mask**" value and the "**Default**" value.

Permit or Deny?

The ACL list of call-signs can be configured as an "**Access**" list or a "**Deny**" list. It all depends on the Default and Mask values you use. Naturally, one must be very careful when using it, a Sysop could very well lock him or herself from the node if the command is not entered correctly. This means that a trip to the node site may be necessary to reset the node before radio access can again be gained.

The Default Value

This is a number chosen that reflects a "no action taken" in response to call-signs that match this value.

For example, if you enter your own call-sign with the default value, you get access under all circumstances. If you enter a call-sign with a value higher than the Default value, it is subject to evaluation with the Mask Value.

The Mask Value

This value tells the node which ACL functions are to be used. This is based on the bit values mentioned in Part One. Mask values are these bit values plus 1. Look at the list of these values in part one while we discuss the Mask.

The Deny List Configuration

Let's say that we want the **Default Value** to be **0**, and we want a **Mask** value of **127**. Now let's suppose we have a distant node that is not a good path, one that we wish to prevent from connecting to the node. In our example, the node call is WX9BAD-5. We will enter a default value and then assign a value of 127 to this call-sign with these commands (executed in the Sysop mode):

ACL * 0 *< Assigns an ACL Default value*

ACL & 127 *< Assigns an ACL Mask value*

ACL WX9BAD + 127 *< Enters this call-sign and value to the ACL list*

This means that WX9BAD will not be able to make an incoming or outgoing AX.25 connection (necessary to initiate a Net/Rom circuit), and will ignore the SSID. This corresponds to bit 0 +1, bit 1 +1, and bit 6 +1, which is 127.

The **Mask** means that your our node will only bar incoming and outgoing AX.25 connections, and will ignore the SSID of the listed station when checking the call-sign. Now we can get a little tricky.

I mentioned before that it is possible to keep everyone (including the sysop) from connecting to the node. Let's say, for example, that I am the Sysop of this node. I can prevent this from happening by the addition of this command:

ACL KB9ALN + 0

This places may call-sign at the Default level, and the Mask value (hence ACL) has no effect when the node encounters my call-sign.

The Access List Configuration

We can also select Default and Mask values to achieve the opposite effect. Let's say that our node wishes to communicate with another node for Level 3 and 4 connections. We'll call this node WX9GUD-5. Here are what the commands look like for ACL access list:

ACL * 7 *< Set the Default value at 7*

ACL & 0 *< Set the Mask value at 0 - No Mask used*

ACL WX9GUD-5 + 64 *< Set the ACL Value for this station to 64*

Now we are guaranteed to be able to connect and make Level 3 and 4 connections to WX9GUD-5. Note that we have not set the "Ignore the SSID" Bit, so this entry only applies to WX9GUD-5. WX9GUD-3 would not be permitted these priveleges. Why?

There are reasons for doing this, mostly for network management. In the first example, we denied access to and from a node that is unreliable. In the second, we guaranteed access to a known, reliable node and made sure it was part of the network.

Removing an ACL Entry

Call-Signs are added to the ACL List with the "+" modifier, and removed with the "-" modifier. No other information need be entered when removing a call-sign from the list.

To change or deactivate the Default or Mask values, just enter a new value of 0. Here are a few examples:

ACL WX9BAD - *< Removes WX9BAD from the ACL list.*

ACL * 0 *< Removes the Default value.*

ACL & 0 *< Removes the Mask value.*

ACL - *< Shuts off all ACL functions.*

WAPR Recommendations

ACL is difficult to understand and can be tricky to implement. For that reason, WAPR recommends you not use it if there is another way to accomplish your goal. As was cautioned before, you can effectively *lock yourself and everyone else out of the node* if you don't get it right. This has happened, and it made for quite a problem when trying to change anything on this node.

If you must use it, be sure to make certain that you use the above example of **0 default**, your desired mask, and set several call-signs as 0 so that selected Sysops can connect to the node. I say "selected Sysops" just in case something happens when you are out of town or something prevents you from operating the node. You should have at least two other people set up to be able to Sysop the node if something keeps you from maintaining the node. Otherwise, your node may be inaccessible to the rest of the network. In Part 3, we'll continue our discussion with **ADC**.

Net/Rom Node Information for the Sysop - Part Three

by Andy Nemec, KB9ALN

This is part 3 of a series designed to help node Sysops learn more about the popular TheNET X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this installment, we look at **ADC**.

What is ADC?

ADC is both a user command, and a Sysop command that can take four forms. A user typing "ADC" can retrieve data that is gathered by an Analog-to-Digital Converter channel wired to the TNC. There are four such channels, and they can be used to set up an S-meter, a Deviation meter, a thermometer or a voltmeter, among other things.

Actually, all ADC channels are really voltage meters, they are just calibrated and set up to dispense a particular type of meter reading. The sysop ADC commands are longer, and work in conjunction with the **METER** command. Giving you a complete run-down on how to construct, calibrate, and adjust an S-meter is beyond the scope of this article. We'll only touch on this for those who may be entertaining the thought of adding something interesting to their node stack. The complete instructions can be found in the NETX1JR4 package available at the TAPR software library (www.tapr.org).

Basic Setup

The Analog-to-Digital converter inputs accept a voltage range of 0 to 3 volts maximum. In the example of an S-Meter, one finds a suitable voltage from the reciever, or constructs circuitry that produces it. The documentation in the X1JR4 package has some very helpful ideas on how to do this.

Once a suitable voltage has been found or created, you determine how the voltage relates to the signal strength, and what voltages correspond to minimum and maximum signal strengths. This is done with a signal generator and a step attenuator. A graph of these values is made.

Once this is known, a decibel multiplier factor is calculated. After this is obtained, and the zero-signal (noise floor) reading is known and graphed, a "multiplier figure" is found on a trial-and-error basis. After this, the Analog-to-Digital Converter is wired into the TNC (again, detailed instructions are contained in the X1JR4 package).

The approprite wiring is put into place, a particuilar ADC channel is selected, and the configuration process can begin.

Sysop Commands

In addition to the user ADC command, there are 4 ADC commands for the Sysop. They are ADC1, ADC2, ADC3, and ADC4. These commands are used to enter a text string that follows the ADC channel output. This text string is limited to a maximum of 8 characters. These have to be entered into the node after a cold start and are not "burned into" the node's EPROM.

To enter the text for the ADC1 channel, first this command is issued to clear the text that may already be in the buffer:

ADC1 *

Then we enter the text that tells us what we are measuring. If we want this to be an S-Meter, we would enter this:

ADC1 S-Units

The most complex command associated with setting up the ADC metering systems is the **METER** command itself. Basically, the METER command dictates what the ADC voltage will be displayed as, how to make this voltage conform to a particular scale, and how to multiply the ADC channel reading so that the meter reaeds correctly.

The **METER** commands configure:

- The Mode of the meter; whether it will be an S-Meter, a Voltmeter, or whatever.

- Scaling factor for deviation.

- The S-Meter or dbm meter multiplier factor and noise floor values.

- The voltmeter multiplier value.

- The voltmeter offset values on a per-channel basis

Conclusion

We probably won't talk more about the METER command later in this series. Showing you examples of how to set up one of these meters is complex and cannot cover every application. For that, I can only refer you to the X1JR4 package - it has extensive documentation not only on the commands themselves, but also has circuitry examples. That's all for this part. We'll continue next time in our series by discussing the **ALIAS** and **ARP** commands.

Net/Rom Node Information for the Sysop - Part Four

by Andy Nemec, KB9ALN

This is part 4 of a series designed to help node Sysops learn more about the popular TheNET X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. This month, we look at **ALIAS** and **ARP**.

ALIAS

Most every node has an alias - a name other than it's call-sign - that it is known by. TheNet X-1J revision 4 nodes can have the standard network node alias plus 3 other aliases. These are **BBSALIAS**, **DXCALIAS**, and **HOSTALIAS**.

The **BBSALIAS** is simply another name that the node is known by for BBS purposes. If a station attempts to connect to the BBSALIAS, the node will respond and carry out the BBS connect command (automatically connect to a specified BBS).

In the case of **DXCALIAS**, the node connects the user to a DX Cluster. **HOSTALIAS** will connect the user to a local TCP/IP "Host" computer. You don't have to set these other aliases if you don't have these services in your area.

Of course, you would want to set the network node alias. The command syntax is the same for all commands involving an ALIAS. Naturally, you need to be connected to the node as a sysop in order to use these commands. Here's a sumarry:

ALIAS *

Clears the alias entry.

ALIAS *your_alias_name_here*

Will give your node an alias, simple as that.

Note that there are certain checks the node makes to see if your proposed alias is permitted under it's "rules". Changing aliases without careful consideration of the problems it may cause others is discouraged.

Most commonly, the alias is "burned" into the node's EPROM so you probably would only re-enter a node alias if something really bad happened, or it was changed.

ARP

Arp is a command closely related to other TCP/IP commands for TheNet X-1J. One of the most recognizable feature of these nodes is the ability to act as a TCP/IP router. That is, it can correctly and automatically route TCP/IP packets (called ***datagrams***) to their intended destination.

What is "ARP"?

ARP means "**A**ddress **R**esoloution **P**rotocol". In the world of **TCP/IP**, computers are known by their names (called "***hostnames***") as well as a number (called an ***IP** address*). In the Amateur Radio World, Call-signs are used to differentiate stations. ARP bridges this gap by associating an IP address to a Call-Sign. ARP allows the node to properly direct TCP/IP connections toward a particular call-sign-equipped radio transmitter and TNC, eventually reaching the computer.

The Node Sysop maintains a small database in the node (called an "***ARP Table***") that it uses to remember which call-

sign goes with which IP address. ARP, when used as a node command, is used to manipulate this database. Sound confusing? It's really pretty simple when you take it step-by-step.

For now, let's just assume that WX9APR has an IP address of 44.92.232.5. All we have to do is to tell this to the node. Let's put WX9APR into a node's ARP Table. We'd use the command form of:

ARP 44.92.232.5 + P AX25 WX9APR DG

The First number appearing after the ARP command, **44.92.232.5**, is the IP address belonging to WX9APR.

The **+** indicates you want to add it to the list.

The **P** indicates that you want the node to *Publish* this arp entry - tell it to other computers that are listening on the frequency.

AX25 tells the node to use the AX.25 protocol set (the Amateur Packet Standard) to talk to **WX9APR**, which is the call-sign used to connect.

The Last entry, **DG**, means to use the "Datagram" mode. The Datagram mode uses end-to-end packet acknowledgement. There is one other choice, VC. This means that the connection is to be a "***Virtual Circuit***" type. These connections are hop-to-hop acknowledged. There are special uses for both. The general rule seems to be to use DG on reliable paths, and VC when the path is error-prone.

This is fine, we have WX9APR on the list. What happens when he decides to change call-signs, and we have to remove the old one from the list? That one is easy enough:

ARP 44.92.232.5 - AX25

From there, you can add that address on a different call-sign using the **ARP +** variant of the command.

Any incorrect syntax using the ARP command can make things interesting. One common mistake is not entering enough information when dropping a station from the list. Forgetting to enter the IP Address, for example, will turn ARP off entirely. Turning it back on is simple: **ARP +** Will do it.

ARP -

by itself, as you have probably guessed, totally disables ARP.

That's all for this part. Two of the final installments in this series will sumarrize what it takes to get the TCP/IP functions operating on TheNet X-1JR4 nodes. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Five

by Andy Nemec, KB9ALN

This is part 5 of a series designed to help node Sysops learn more about the popular TheNET X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. This month, we look at **ARPTIMER** and **AUDIT**.

ARPTIMER

This command configures part of the node's **ARP** system. You may remember from our last installment that ARP associates a call-sign with an IP address in the ham radio implementation of **TCP/IP**. This information is kept in a list on the node called an ***ARP Table***. The **ARPTIMER** command allows an automatic update of the ARP table that the sysop has set up, allowing new stations that are not in the ARP table to be added to the ARP table automatically.

This command can also specify how long a station that is automatically added to the ARP table will remain in the ARP table. Here's a command usage sumarry:

ARPTIMER *

clears the current ARPTIMER settings.

ARPTIMER 0

Shuts off the ARPTIMER function. The manually-configured ARP will continue to work, but no automatic updates to the ARP table will be performed.

ARPTIMER 1

Enables automatic updating of the ARP table.

ARPTIMER *(number from 15 to 1440)*

This specifies how long the automatic updates in the ARP table will remain in the table, in minutes. For example:

ARPTIMER 30

Means that any ARP update will remain in the ARP table for 30 mintes. Again, this does not affect the entries made manually to the ARP table.

We'll recap **ARP** and it's related settings in the final installments of this series when we look at how to set up a TheNet X-1J node for TCP/IP operation.

AUDIT

This command configures the amount of information dispensed to node operators and managers when they issue the **AUDIT** command when not logged in as a sysop. The node keeps track of a number of statistics, and the Sysop or node manager can pull these statistics when needed. This is as close to a station log that the node can come, and is handy for checking node performance and for troubleshooting.

The information that can be dispensed includes:

- The percentage of time a Data Carrier was detected, and how long the transmitter has been keyed in the last 10 minutes.

- A mini-log of the various connections that have been made to the node, both AX.25 and Net/Rom frames.

- Node CPU load and buffer usage

- Usage of the SYSOP command

A few other types of info can also be dispensed as well.

One important note about the AUDIT command - it can slow down a node considerably and consume considerable memory when the node has a lot to do. The only time it should ever be needed is when troubleshooting.
Here's a command summarry:

AUDIT 0

This turns off the Audit function completely. This is really the preferred normal-use setting.

AUDIT ***(number from 1 to 255)***

Enables auditing of one or more parameters. The number is based on a "**Mask**" value determined by what Audit functions you wish to enable.

There is considerable information that can be dispensed with the AUDIT command when used in this manner. Due to the infrequent usage of this command, I won't delve into all of the options. Consult the X-1J manual for details, or mail me with specific questions and I'll help you the best I can.

That's all for this part. In our next part, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Six

by Andy Nemec, KB9ALN

This is part 6 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we'll look at the **BBS**, **BBSALIAS**, **BRATE** and **BTEXT** commands.

BBS

If you have a BBS that is accessible through your node stack, this command makes life easier for your users. When **BBS** is entered as a command by a user, they'll be connected to a BBS set by the Sysop. No call-sign or alias needs to be remembered by the user, which certainly makes it easy for out-of-town packet operators.

When the Sysop enters the command when the node is in the Sysop mode, the Sysop has options to set the BBS, as well as to clear an entry that was previously made. Here are the various Sysop forms of the command:

BBS *

Clears the node's entry for the BBS, if one was set. Usually, a Sysop only needs to enter this command if he or she wishes to clear or change a current designated BBS. If the BBS is cleared, the command is disabled.

BBS ?

Asks the node what call-sign has been entered as the BBS.

BBS (*call-sign*)

Sets the call-sign of the BBS to connect to. This can be either the actual call-sign or the Node alias of the BBS. Only valid node aliases are allowed if the call-sign validation is active.

BBSALIAS

This is another alias that the node can be known by. It does not get broadcasted as a separate node, which means it won't conflict with the BBS's node broadcast itself if the BBS is acting as a node, too. It's used as a shortcut for users on a LAN that can connect to a node directly. For example, if the BBS is hard-wired to the node's RS-232 port, one has to access a node to connect to the BBS. A user normally would first connect to the node, and then type "BBS", or the call-sign or alias of the BBS/Node. This eliminates a step, allowing what the user sees as a direct connection to his or her station. It operates in conjunction with the BBS command. If the BBS and BBSALIAS are both set, then the seemingly direct connection will take place. The BBSALIAS command takes two forms:

BBSALIAS *

clears the current entry.

BBSALIAS (*call-sign or Node-type Alias*)

will set this additional node alias to whatever call-sign is used. One note of caution here - you must make certain the BBS is not on the same frequency as the node and using the same alias. For example, if a BBS has a port on the LAN, and the node is also on the LAN, two stations will try to answer a connect request. This will, in all likelihood, generate a "*Frame Reject*" in response to a connect request, and no one will be able to connect to the BBS.

BRATE

This command is specific to the PK-96 TNC when TheNet X-1J firmware is installed. It sets the serial port speed used to communicate with other TNCs on a node stack. If the firmware is installed in a TNC-2 or clone, it is not used.

BRATE *

will clear the current serial speed.

BRATE ***(choose 300, 600, 1200, 2400, 4800, 9600, 19200)***

Sets the serial port to the corresponding speed. TheNet X1-J authors caution that 19200 bps may not function well, as it is untested.

BTEXT

This sets the beacon text - what you want it to say when it beacons - for the node. While it is a good thing to beacon the node's existence from time to time, it's easy to overdo it. Naturally, the beacon is only seen on the radio port, and not the RS-232 port. Here are the various forms of this command:

BTEXT

sent to the node by itself by a Sysop while the node is in Sysop mode will simply display what the beacon text, the beacon message, is set to. alternatively, the Sysop can send

BTEXT ?

to find out what the current BTEXT is.

BTEXT *

As you may have guessed, clears the beacon text and sets it to nothing, blank.

BTEXT ***(beacon text message)***

will set up the beacon to send the text as a UI Beacon packet. The total amount of text that can be sent is limited to 160 bytes (80 characters including spaces, control characters and carriage returns), and can be separated by multiple lines.

The Sysop can even start a message with a blank line. There is a bit of a trick to this, however, and it's really not a blank line. The Sysop sends a control-character combination (*Control-A* is suggested in the documentation) at the start of a message and this appears on most packet terminals as a blank line.

One note concerning **BTEXT** - if you forget to send the **BTEXT *** command before programming a new message, the "new message" will simply be ***added*** to the old one, if the total doesn't exceed 160 Bytes.

Which is how you get a beacon text message to be displayed in multiple lines. The Sysop would enter each line of text in succession, each preceded by BTEXT. In this example, we start by clearing the current message:

BTEXT *

Now we'll put three lines of beacon text into the node:

BTEXT *This is the first line.*

BTEXT *This is line 2.*

BTEXT *This is line 3.*

When the beacon is sent, most packet stations monitoring the frequency will see this:

NODE-CALL-SIGN > BEACON UI:
This is the first line.
This is line 2.
This is line 3.

This method works for other types of text the node sends, such as **CTEXT** (the text users see when they connect to the node). We'll discuss that in a future part of this series. That's all for this installment. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Seven

by Andy Nemec, KB9ALN

This is part 7 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we'll look at the **CALIBRATE**, **CLOSEDOWN**, **CTEXT**, **DXCLUSTER** and **DXCALIAS** commands.

CALIBRATE

The **CALIBRATE** command is used in initial setup and maintenence of the node. This command keys the transmitter up to 60 seconds with alternating audio tones, which would allow you to set the transmitter deviation on a node. It would also allow you to remotely key another X-1J node, allowing you to look at the receiver audio, which can be quite helpful in diagnosing receiver problems.

Here's a little bit of "review" information that will be helpful in understanding one form of the command; packet data is sent by alternating between two audio tones when transmitting in ***A****FSK* mode. In *FSK* mode, it alternates between two radio frequencies. The **CALIBRATE** command allows you to send alternating audio or radio tones for up to 60 seconds.

Additionally, you can specify how long to alternate the tones within the transmit time period of the command. The syntax of the CALIBRATE command is:

CALIBRATE *(total transmit time, **1-60** seconds) (toggle interval)*

The command can also be shortened to **CAL**. Here's an example of usage:

CAL 20 10

This would set the node to transmit for **10** seconds at one of the audio tones, and **10** seconds of the other tone for a total of **20** seconds.

One special note concerning this command - remember that your TNC has a time-out timer to keep the node from transmitting too long should the node lock up. If that timer is set for 30 seconds and you specify a calibration trasnmission of 60 seconds, you'll only get a 30 second transmisison.

CLOSEDOWN A

This command is almost self-explanatory. When issued, the node will prompt you for the Sysop password, just as if you had entered the **SYSOP** command. I will not detail exactly how the SYSOP command works for security reasons.

Once you have successfully given the correct SYSOP response, the node will stop transmitting and can't be accessed. No routing tables or other operating parameters will be removed from the node's memory. However, in order to reactivate the node, a power-up reset will need to be performed. This means you will have to go to the node site and cycle the power off and then back on. Therefore, caution is in order - you will want to carefully consider whether you really want to use the **CLOSEDOWN** command.

CTEXT

This command sets or clears the text that users will see when they connect to the node. It takes the same form as other text commands, such as **BTEXT**. For example,

CTEXT *

will clear the existing connect text from the node's memory.

CTEXT *Welcome to the WAPR Network Node!*

Causes the node to return "**Welcome to the WAPR Network Node!**" when a user connects to the node.

There are two things to consider when entering the connect text. Like other text commands, the connect text is limited to 160 bytes, which is enough for 80 characters (including letters, spaces, and control characters). If you forget to clear the existing text, whatever you enter will be added to the existing text.

DXCLUSTER

This command specifies the call-sign of a DXCluster node or station that a user can connect to by issuing the DX command when connected to the node, or when a user sends a connect request to the **DXCALIAS** (see below). It operates in the same manner as the BBS command, and it takes the same forms.

DXCLUSTER *

Clears the current DXCLUSTER setting.

DXCLUSTER ?

When entered, the node will return the call-sign of the current DXCLUSTER.

DXCLUSTER *WX9APR-1*

This will set the DXCLUSTER to WX9APR-1. A node user will be connected to this station when the DX command is sent to the node.

DXCALIAS

This oerates in the same manner as the **BBSALIAS** command that we covered in our last installment. This allows the node to act on behalf of a DX Cluster node or station.

For example, if a DX Cluster node or station is not operating on a local LAN frequency and is accessable to the node through the network, a user can simply send a connect request to the DX Cluster alias and be connected via the node. This appears as a transparent connection to the user, as though he or she were connected directly to the DX Cluster station. In fact, the node is acting on behalf of the node, "faking it".

As with DXCLUSTER, the command takes three forms:

DXCALIAS *

Clears the current alias.

DXCALIAS *?*

Displays the current alias, and

DXCALIAS (*call-sign or alias)*

will set the DXCALIAS to the specified call-sign or alias. Here's an example:

DXCALIAS *WAPRDX*

will allow a user to send a connect request to *WAPRDX* and get connected to a remote DX Cluster as set by the **DXCLUSTER** command. In the above example we used for the DXCLUSTER command, this would be *WX9APR-1*.

One thing that should be noted concerning any alias-based command. If you are going to be using a someone else's call-sign rather than an alias, you should inform the owner of that call-sign and get his or her cooperation. After all, the call-sign belongs to someone else, and they should know that you're node will respond to that call-sign.

That's all for this installment. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Eight

by Andy Nemec, KB9ALN

This is part 8 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we'll look at the **HOST**, **HOSTALIAS**, **INFO**, and **IPADDRESS** commands.

HOST

This command is an interesting one in that it offers a couple of interesting options. You can think of it as being exactly like the BBS command, but it also has an additional option that can make for an interesting node configuration.

In the customary multiple-node ("*Nodestack*") configuration, the RS-232 ports of each TNC are connected together so that one node can connect to another. For example, you can have a VHF LAN node connected to a UHF Backbone node, and a user can connect from one node to another. The serial port can also be connected to a computer, so that the node's parameters can be manipulated on-site.

One can also connect a serial printer up to this port. In this case, a remote user can ussue the "**HOST**" command and can send a message to the attached computer or printer. This feature, while interesting and novel, probably would not see any use in typical node installations.

It is more likely to be used in the same manner that the BBS command is used. When this command is issued by a user, they will be connected to a designated BBS or TCP/IP station. The connection will be made to the AX.25 or Net/Rom (Node) port of the BBS or IP station. The Sysop command syntax is identical to the BBS command syntax for the Sysop, when connected to the node in the Sysop mode. Here's a command summarry:

HOST *?*

will return the current call-sign of the designated host or BBS, or return with the response "***HOST is not set***".

HOST **

will clear the existing HOST entry. As a matter of habit it is probably best to issue this command first, to make certain you flush out the existing entry.

HOST *WX9APR*

would set the designated host to *WX9APR*. A user who issues the HOST command would be connected to this BBS. If this is set to an alias, Net/Rom connection is made, otherwise it is a standard AX.25 connection.

HOSTALIAS

This operates identically to the **BBSALIAS** and **DXALIAS** commands. If the node is set up to respond to additional aliases, it will answer on behalf of the designated **HOST** and connect the user to it, acting as an intermediary. To the user, this will appear as a direct connection, even though it really isn't. Here's the command syntax:

HOSTALIAS ***

clears the current **HOSTALIAS** entry. You can use it to flush out the old entry when changing to a new HOSTALIAS for example, or when you've made a typo and need to re-enter the HOSTALIAS.

HOSTALIAS *WX9APR*

Sets the **HOSTALIAS** to *WX9APR*. If a user's packet station attempts to connect to WX9APR, the node will answer for WX9APR and make a connection to that station. A valid alias can also be used, for example, *IPBBS*. This way, the node can make connections to 3 different service-oriented BBSs - the customary "normal" BBS, a local DXCluster, and an IP-oriented BBS named as a Host.

INFO

This is familiar to most Node operators. It sets what the user will see when he or she issues the **INFO** (or "**I**" abbreviation). Usually, this is the location, sponsor of the node, maybe the grid square and perhaps other information.

The command syntax is identical to the **BTEXT** and **CTEXT** commands, and also has the same size limitation - 80 characters including spaces and control-characters. The command syntax is the same:

INFO *

clears the current text. If you want to completely change the text, rather than just add to it, this is the command you would use.

Sending the following to the node:

INFO *WAPR Network Node in Thimble Creek, Wi - 145.090 VHF LAN*

changes the info text so that the user will see this when he or she issues the **Info** (or "**I**") command:

WITHIM:WX9APR-1} WAPR Network Node in Thimble Creek, Wi - 145.090 VHF LAN

It's helpful for packet operators to have this information when the node sysop has taken the time to enter it. In instances where it hasn't been set, the traveling packet operatior does not know where they've "landed", so to speak.

IPADDRESS

As you'd expect, this sets the **IP Address** of the node, quite simply. IP Addresses are assigned by IP address coordinators to packet operators that request them. They are free - there is no charge for obtaining one.

In Wisconsin and the Upper Peninsula of Michigan, we have local IP address coordinators and a State coordinator. If you'd like to obtain an IP address, Mail Me or send a packet message to: KB9ALN@KB9ALN.#GRB.WI.USA.NOAM.

I will either assign you an address, or refer you to your local IP Address coordinator.

IP addresses contain 4 sets of numbers seperated by periods. The numbers range from 0 to 255. IP addresses in the Amateur Packet world all start with 44. In Wisconsin and the UP of Michigan, they all start with 44.92.

For example, my first IP address is 44.92.20.9. The addresses are assigned based upon location and operating frequency. It is important that you do not pick one at random - they must be assigned. In addition, there are certain conventions that follow the internet conventions, and your IP Address coordinator will assign addresses based upon the function of the node. If you do obtain an address for your node, here's how you would set it:

IPADDRESS *44.92.254.1*

Sysops or users can see the IP address of the node simply by typing

IPADDRESS

with no other text. The node would then respond with:

My IP address is: 44.92.254.1

Take note that this is example address is not assigned, and it's very important that you not use this particular address - you must only use an address that is assigned to you.

A reminder - we'll cover all of the aspects of setting an X-1J node up for TCP/IP operation at the end of this series. This discussion will center on understanding TCP/IP and how the node treats it. Don't let TCP/IP parameters intimidate you, they're not as difficult to understand as you'd think.

That's all for this part. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Nine

by Andy Nemec, KB9ALN

This is part 9 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we'll look at the **IPBROADCAST** and **IPROUTE** commands.

IPBROADCAST

This command sets an IP address that will be ignored by the node. The most common reason may be some IP packet programs that send out routing information that isn't used by the node and may confuse it. First, some background info.

A Network Address is used for routing purposes, and is derived from a block of IP addresses used in a particular ***sub-network***. Here's a brief explanation of the **sub-net** concept as it relates here.

You'll notice that you and your neighbors have similar IP addresses. For example, the Green Bay area IP addresses range from 44.92.20.1 to 44.92.20.254. They all have a **common element** to their addresses - the dotted number combination of **44.92.20.** When referring to the Local Network Address for routing purposes, it is written with a 0 at the end of this dotted number combination.

For example, the Green Bay **Local Network Address** shown above is written as **44.92.20.0**. When entering the IPBroadcast address in the node, you will need to be in the sysop mode and you'll enter the local sub-net's Network Address. The command syntax is:

I**PBroadcast** ***(local network address)***

In the case of our Green Bay example we used above, this would be:

IP Broadcast 44.92.20.0

For those of you who have some experience with IP networks, you may be a confused a bit. Normally, a broadcast address is different than what is shown here. Yes, this command could have been named more appropriately. However this is what the X-1J node needs, and this is what we give it.

IPROUTE

This command tells the node how to route IP packets to their destination. At first, this task can seem rather daunting, but once you get the hang of it, it all makes sense.

In order to effectively route IP packets, we need to use 2 forms of the **IPROUTE** command. One is intended for individual hosts with their own IP addresses. The IPROUTE command syntax would look like this to enter a specific host with it's unique IP address:

I**PROUTE** *(Destination IP Address) (+) (Port 0 or 1) (Gateway IP, if used)*

Let's say we have a station with an IP address of 44.92.130.10. It is on port 0, the radio port. We wish to add this, but need to use 44.92.130.9 to get to the 44.92.130.10 host. First, we need to enter a route to the gateway, or relaying station address. In this case, our IPROUTE command would look like this:

IPROUTE 44.92.130.9 + 0

This adds a route to the 44.92.130.9 host on the radio port, port 0. Now we can add the final destination with this version of the command:

IPROUTE 44.92.130.10 + 0 44.92.130.9

This will add the route to 44.92.130.10, (and use 44.92.130.9 to get to it) via the radio port, Port 0. If we wished to remove this route, we would send the command:

IPROUTE 44.92.130.10 -

We would not need to remove the gateway entry, unless specifically needed.

The other IP routing task is meant for accessing other networks, represented by blocks of IP addresses and the number of significant bits. Here is a brief explanation of networks and the number of significant bits.

First off, IP addresses are, for the purposes of our discussion, 32 bits long. When you have an IP address of 123.234.123.234, each section of the address is 8 bits long. Each of the four groups of numbers separated by dots are 8 bits long. The entire IP address is 32 bits in total.

What IP routing in a node does is send IP packets to a particular place based on how many bits of the address are specified. We can specify all 32 bits - as we did in the example above - and send an IP packet to a particular station address, or we can send a group of IP addresses on to another station or node. The other station or node handles it from there.

OK, so how do we specify a particular network address block for routing purposes? Here is a guide with examples that specify the number of significant bits:

44.0.0.0 has 8 significant bits and specifies any address that begins with 44. This is the 44 network. All Amateur Packet Radio stations with IP addresses begin with 44 (except in special cases).

44.92.0.0 has 16 significant bits and specifies any address that begins with 44.92. This is the 44.92. sub-network, which is Wisconsin and the UP of Michigan.

44.92.130.0 has 24 significant bits, and specifies any address that begins with 44.92.130. This block of addresses is currently not assigned, and is only used for example purposes. It, too is a sub-network. So in order to route to a block of IP network or sub-network addresses, we use this form of the IPROUTE command:

IPROUTE (*ip address)(/bits) (+) (port number) (gateway address)*

Here's an example of how to route to the 44.92.130 block of addresses, using 44.92.130.1 as the gateway to all other 44.92.130 series of IP addresses. We'll be using the Radio Port 0.

We need to add a route to a gateway first, before we can route through it. We do that like this:

IPROUTE 44.92.130.1 + 0

That takes care of the station we will be using as a gateway to our fictional 44.92.130.0 network. Now we add the network route with the command:

IPROUTE 44.92.130.0/24 + 0 44.92.130.1

Similarly, we can route other networks, using the network addresses, the number of significant bits, the port number and the gateway address.

To remove a route to a network, we need to specify the network address, and the number of significant bits. Here's an example:

IPROUTE 44.92.130.0/24 -

We don't need to specify a port number or gateway when deleting a route.

For a more complete explanation of networks and sub-networks, see the Wisconsin/UP IP coordinator's page at: www.gbonline.com/~kb9aln/4492-ip/ (currently under construction).

That's all for this part. Next month, we'll continue our alphabetical exploration of the Sysop commands for these nodes. Until then, 73 from Andy.

Net/Rom Node Information for the Sysop - Part Ten

by Andy Nemec, KB9ALN

This is part 10 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we will discuss the **IPStats**, **L3MHeard,** and **Links** commands.

IPStats

This command can best be compared to one that is familiar to most veteran Node operators - **Parms**. Parms is general parameter dump command that dispenses the settings of various node operating parameters, such as TXDelay and routing timers.

In the case of **IPStats**, it dumps the TCP/IP specific operating parameters and statistics. It will give a node operator a good idea of how well the node is operating, and what kind of load it may be experiencing. It can be very helpful to occasionally save this information dump and evaluate it from time to time to see just how well the node is operating.

When you view the **IPStats** information dump, it will appear as a string of 20 numbers separated by spaces. There is no text associated with it, you will need to have a cheat-sheet handy to "decode" it. Here's the general form of the information dump after you issue the command "IPStats", with output from the node listed as N1, N2, etc:

N1 N2 N3 N4 N5 N6 N7 N8 N9 N10 N11 N12 N13 N14 N15 N16 N17 N18 N19 N20

And here is what each of the numbers you'll see means.

N1 displays what the default AX.25-level IP connection type is - Virtual Circuit or Datagram- for both the radio and the serial ports. It is a number from 0-3. Here's what these numbers mean:

0 = Mode Virtual Circuit is used on both ports, radio and serial
1 = Datagram mode on radio port, Virtual Circuit on serial port
2 = Virtual Circuit on radio port, Datagram Mode on serial port
3 = Datagram mode on both ports, radio and serial.

N2 displays if IP Routing is enabled on the node. The number 0 indicates that routing is disabled, 1 indicates enabled.

N3 is the "Time-to-Live" value. This is the maximum number of hops, or relaying nodes, that the IP packet will pass through. Once this value is exceeded, the packet is unceremoniously dropped, ie., it "dies".

N4 is the number of received IP frames

N5 is the total number of IP headers the node has received

N6 shows the number of IP address errors the node received

N7 is the count of IP Datagrams (packets) that have been forwarded.

N8 shows a count of protocols that the node has heard, but can't identify.

N9 is another counter, showing the number of dropped IP frames

N10 counts the number of IP frames delivered

N11 is the number of IP requests for output

N12 shows a count of the IP datagrams dropped by the receiving nodes

N13 is the number of datagrams refused due to no route available

N14 is a count of IP reassembly timeout errors

N15 tells how many times it experienced an "IP Reassembly Required" error

N16 is how many times an IP datagram was successfully reassembled

N17 Counts how many times an IP datagram was unable to be reassembled

N18 is the number of times a datagram was successfully sent in fragments

N19 shows the the opposite - IP fragmentation failures

N20 shows us how many times an IP fragmentation was initiated by this node

This is a lot of information that might get you thinking about the complexity of IP in general. We will clear this up a bit for you when we conclude this series with a special section detailing how to configure a node for TCP/IP use.

L3MHeard

This command will yeild a list of connections that the node has heard at the Net/Rom level - level 3. Here's a typical output one would expect when the command "**L3MHeard**" is sent to the node.

```
#WAPR1:WX9APR-1}
Callsign        Pkts   Port Time
KE9PW-5    40248   1    2:5:1     KA9JAC
KA9JAC-5    2374   0    2:5:1     KE9PW-5
KA9JAC       276   0    2:5:27    KE9PW-5
```

The first 4 columns of output should be pretty easy to figure out - they show the originating node, the total L3 packet count from that node, the port number the node is heard on and time since the last L3 packet was heard from the originating node. The last column is not labeled and idicates the destination of the L3 packet last heard.

The first line, for example, shows us that the node KE9PW-5 was heard relaying 40,248 L3 packets on Port 1, the last one being 2 hours, 5 minutes and one second ago, and that these packets were destined to KA9JAC.

Links

This command displays the current L3 links that the node is taking part of. It shows the two stations it is linking, the state of the link, the port number used for the link and the retry count. Here's a sample output when the command

Links

is issued:

```
#WAPR1:WX9APR-1} Links:
WX9APR-1 KB9ALN 4 0 0
WX9APR-1 KB9ALN-15 4 0 0
```

Note the third column, link state. This is not defined in the documentation included with the X-1Jr4 firmware package. 4 seems to indicate that the stations are currently linked. Other numbers may indicate that a station is in the process of setting up or closing a link. In our example, WX9APR-1 is linked to KB9ALN on port 0 and there have been no

retries associated with this link.

That's all for this part. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Net/Rom Node Information for the Sysop - Part Eleven

by Andy Nemec, KB9ALN

This is part 11 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we will discuss the MANGER, METER, MHEARD and MODE commands.

MANAGER

This command is intended to be a secondary Sysop command, but it does have some interesting (and potentially troublesome) "Gotchas" involved. For that reason, and a few others, it's recommended that you not use it.

The main purpose of the **MANAGER** command is to collect data that is displayed with the **AUDIT** command. It's intended as another way a user can do some minor Sysoping, but it somehow got expanded to include all Sysop priveleges.

The main "Gotcha" with this command is that you need to connect to the node with a level 4 connection in order to utilize it. Due to a timer discrepancy, the connection may not properly time-out. The Sysop commnad does not have this linitation and provides the same functionality.

The **MANAGER** command takes the following forms:

MANAGER [+] or [-] or [password]

MANAGER -

disables the MANAGER command, while

MANAGER +

enables it.To enter the **MANAGER** mode, the Sysop password is sent on the same line as the **MANAGER** command.

In order to keep confusion to a minimum, it is recommended that you disable this command, and instead use the SYSOP command to collect any data or make an changes that require Sysop privelges.

METER

The **METER** command is used in conjunction with the **ADC** command, and is used in the setup of the S-meter, Deviation meter or other analog measurement "Channel". The command syntax is a little too involved to fully explain here, and it requires some interaction with the **ADC** command. It's recommended that you consult the X-1J documentation if you plan on setting up a metering circuit of any kind.

Basically, it involves setting up a particular metering circuit's characteristics, such as how it will scale, so that it can render acurate readings to the users. The Meter outputs are shown when the **MHEARD** command is issued.

Of course, this is a Sysop-only command, and in order to set up any kind of meter parameters you will have to be connected in the Sysop mode and enter the password correctly.

MHEARD

The **MHEARD** command displays a comprehensive list of stations that have been heard by the node. It can be abbreviated as **M** and when issued, the format will look similar to this:

```
 WAPR1:WX9APR-5}

 Callsign      Pkts    Port    Time    Type
WX9APR-4      22084     1      0:2:4   Node TCP/IP
WX9APR-4      20830     1      0:2:9   Node TCP/IP
KE9LZ-4           5     0    0:24:40   Node
KB9ALN-7       8988     0     0:54:3
KE9LZ             1     0    1:12:13   TCP/IP
KB9ALN-5          1     0    1:12:14   TCP/IP
KE9LZ-3          16     0    1:16:43
N9PAV-2        2253     0    1:28:53   TCP/IP
KB9MWR        10350     0    2:39:43   TCP/IP
```

While this may be thought of strictly as a User-level command (as it can be successfully issued by anyone connected to the node), it's included because it's so helpful to Sysops in troubleshooting network problems. As you can see, it shows the heard stations call-sign, followed by a count of packets, the port that the station was heard on, the time since it was last heard and the type of traffic it last heard from that station. If an S-Meter (or other analog measurement) was set up with the ADC and Meter commands, it would appear after the packet type.

MODE

The **MODE** command is used to set node parameters, such as ***TXDelay*** and the like. Think of it as the command used to set what is read with the PARMS command. It can use two different types of input - one is the original TheNet syntax, the other is a shorter and therefore easier form.

Before we look at an example of how to use the command, we'll need to review. Node parameters are read with the PARMS command, and are displayed as a line of numbers. Unless you happen to know what order these appear, you won't make much sense of them. Here's a list of them, with each parameter numbered, representing the order they appear from left to right:

1 - Host mode enabled/disabled
2 - CWID interval
3 - The CWID Keyer speed
4 - Port nodes broadcast control
5 - The Crosslink KISS control
6 - TXDelay value
7 - Full Duplex on/off
8 - RS-232 serial port node broadcast interval
9 - Node Broadcast algorithm
10 - Beacon interval
11 - The "Connect" redirector
12 - Help message enable flag, case sensitivity
13 - TALK 8-bit flag
14 - "Hash" node broadcast port control
15 - Node uses extra aliases on/off
16 - Remote disconnect causes reconnect to this node on/off
17 - Controls "slime" trails
18 - Digipeating uplinks or downlinks control

In order to be able to set a parameter with the **MODE** command, we'll need to know the parameter number. In the examples we'll use, we'll set the ***TXDelay*** to 350 ms, which is typical for a VHF 1200 bps node. We'll show two examples, as two forms of command syntax are permitted.

The ***TXDelay*** command is parameter number 6. In the first example, which has been used since TheNet 1.01, we use the ***** character with the **MODE** command. We insert 5 ***** characters as filler, and include the actual parameter value as the 6th parameter. Here is what that style of command would look like:

MODE *****35

Now compare that with the other form:

MODE /6 35

Both variations are acceptable and perform the same function - set the ***TXDelay*** to 350 milliseconds. Naturally, both types require you to be connected in the Sysop mode

That's all for this part. Next time, we'll continue our alphabetical exploration of the Sysop commands for this node firmware. Until then, 73 from Andy.

Net/Rom Node Information for the Sysop - Part Twelve

by Andy Nemec, KB9ALN

This is part 12 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. We'll skip over the more common user commands and devote our discussion to commands used by the Sysop. In this part, we will discuss the MTU, NODE, and PARMS commands.

MTU

MTU is a contraction of "***M****aximum* ***T****ransmission* ***U****nit*". This is the maximum amount of TCP/IP data that is permitted in an AX.25 packet. It can be thought of as kind of a "Paclen" for IP. In TheNet X-1JR4, the default is 256 bytes.
There is a lot of thought that must go into changing this value. Here are some things to know if you think you may wish to change this setting:

1) Short packets get through best.

2) The node sends an error message if it is asked to handle an IP packet that is larger than it's permitted MTU value.

3) Some users may set their MTU incorrectly, and you may get flooded with mail from those users asking why they are not able to use the node.

4) The node is more likely to crash if it is asked to handle packets that are substantially larger than the default value.

5) Most digipeaters in TNCs will not resend a packet larger than the standard 256 bytes.

6) The calculation used to determine MTU is a little different than most may be accustomed to. Most IP stations set the MTU to 236, subtracting the 40 bytes in the IP header from 256. In the X-1J, they ask you to include all of the AX.25 call-sign and digi data, which is 38 Bytes plus 2 control bytes.

The bottom line is that 256 is a good default value, and it's best to keep it. However, if you have a user who can't seem to be able to send IP through the node, you may want him or her to check their MTU setting (and perhaps change it to 236 if it's a NOS or Linux station, 256 if it's another X-1J node).

If you do find that you need to change the MTU on the node, you'll find that it operates similar to the **MODE** command. Like MODE, you have a choice of syntax you can use - the old style, or the newer one. Here's a list of the MTU changes you can make with either:

Parameter	**Default**	**Function**
1	256	Sets MTU for the Radio Port (Port 0)
2	256	Sets MTU for the Serial Port (Port 1)
3	236	Sets MTU for Net/Rom IP routing
4	257	Sets the MTU that triggers an error response
5	328	Sets the MTU level governing the discard of packets (they'll be ignored)

The first form you can use to enter data has the * character as a filler until you get to the parameter you want to change. Let's say you want to change the MTU on the Serial Port to 512. Here's how that command would look:

MTU * 512

Or the newer syntax makes it look like this:

MTU / 2 512

Both of the above examples tell the node to change the MTU on the serial port to 512 bytes.

NODE

This is really both a user and a Sysop command. It not only lets the user and the Sysop know how this node gets to another node, it's how the Sysop sets up a node's static routing table.

"Neighbour nodes" are ones that you can reach directly, and it's a good idea to have your reliable neighbour nodes "locked in" your node's routing table. It's also not a bad idea to enter known reliable nodes that are easily reachable through these neighbor nodes.

Each node is entered into the table with both an alias and call-sign, assigned a route quality, Obsolescence Count, Port number and digipeaters used to reach it, if necessary.

Here's how a neighbor node, WX9APR-2 with an alias of WAPR2, a route quality of 192, Obsolescence count of 6 on port 0 is added to the routing table:

NODE WX9APR-2 + 192 6 0

Note that this node is a neighbour node - we can reach it directly. If it weren't, we would have added the neighbor node call-sign at the end of that line.

To remove the same node from the routing table:
NODE WX9APR-2 - 192 6 0

Of course you can check routes by using the long form of the user NODE command. Let's say we have a node WAPR3 that we reach from WAPR2, the node shown in the above examples. If we send this to the node:

NODE WAPR3

You'll see this in response:

Routes to WAPR3:

WX9APR-3 192 6 0
WAPR2:WX9APR-2

The first number, 192 is thr route quality, the second is the obsolescence count, the third is the port number.

PARMS

The **PARMS** command is used for reading and setting various basic node parameters. When the PARMS command is issued by itself, the node responds with a string of 26 sets of numbers that tell what each value us. You'll need a key, such as this one below, to figure out what each set of numbers is telling you. Count the 5th number set from the left, for example, and you'll see the current setting of the initial obsolescence count. Here's the list of parameter numbers and what they indicate:

1 Size of destination node table
2 Minimum auto update quality
3 HDLC (radio port) default quality
4 RS-232 (Crosslink) default quality
5 Initial obsolescence count
6 Min Obs to broadcast

7 Nodes broadcast interval
8 Initial time-to-live
9 Transport FRACK timeout (seconds)
10 Transport RETRY counter
11 Transport acknowledgement delay
12 Transport busy delay
13 Transport window size
14 Transport overfill limit (frames)
15 No-Activity time-out (seconds)
16 Persistance (n/256)
17 Slottime
18 FRACK (T1) time
19 AX.25 windowsize (MAXFRAME)
20 AX.25 retries
21 ACK (T2) time (L2 RESPTIME)
22 Activity check (T3)
23 Digipeat
24 Callsign validation
25 Beacon mode control
26 CQ broadcasts

These numbers can also be used to set various parameters by the Sysop. The Syntax can use either of two types of syntax, just like the MODE and MTU commands.

Let's say you want to change the Persistence to 128. Persistence is the 16th set of numbers that you can read or manipulate. In the older style syntax, this would look like:

PARMS * * * * * * * * * * * * * * * 128

In the newer style syntax, it would look like:

PARMS / 16 128

The PARMS command, when used by a non-sysop user, will only allow the display of paramters. The node can be set to not display the parms to users with the command:

PARMS - D

It can be enabled with:

PARMS + D

That's all for this part. Next time, we'll continue our alphabetical exploration of the Sysop commands for these nodes.

Until then, 73 from Andy.

Net/Rom Node Information for the Sysop - Part Thirteen

by Andy Nemec, KB9ALN

This is part 13 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. This month, we'll skip over the user **QUIT** command, and move on to the **RESET**, **ROUTES**, and **STATS** commands.

RESET

This does exactly what you may think - it resets the TNC. There are two ways that the TNC can be reset - a *"Cold Start"* and a "*Warm Start"*.

A *Cold Start* will erase the current node list and routing information. Dynamically obtained ARP information will also be lost. In effect, it is very much like turning the power to the TNC off and then back on. Therefore, any setting that was modified from the original ROM setting (such as TXDelay or Persist) will be lost.

A Warm Start is like typing the command "RESTART" on a conventional TNC. Any parameters that are changed from the original ROM parameters will still be there. It's suggested that you try a warm start first when trying to reset the TNC. The RESET command for a warm start is simply:

RESET

For a cold start reset, add a space and any arbitrary text, such as:

RESET NODE

ROUTES

Users have access to this command in it's most basic form - it will show a table of "locked-in" node routes when entered. When the Sysop uses this command (connected to the node in Sysop mode, of course), it can be used to add, delete, or change the node's routing table.

A node sysop can use the command to "lock-in" a route to a neighbor node, or unlock it. This is used most frequently when the neighbor node is on the same stack, or reliably reached via radio. A sysop can also specify a particular route quality, allowing the node to pick the most reliable path. Routing control is an art, and locking in nodes should be done carefully and sparingly. Here are the various forms of the ROUTE command:

ROUTE

By itself will show you a table of locked-in routes that the node knows about. It will look something like this:

```
WAPR1:WX9APR-1}Routes:

   1  #WAPR9:WX9APR-4 255 28
   1  #WAPR2:WX9APR-7 255 28
>  0  WBBS:WX9APR 225 1
!  0  WAPRDX:WX9APR-2 130 1
```

This is pretty familiar to most experienced node users. The chevron **>** mark indicates that the route is in use. The **!** indicates that the node is a "locked in" route.

ROUTE [***port-number***] [***Node-callsign/SSID***] [***digi-list***] [+or-] [***quality***]

Will add a locked-in route to the specified node call-sign on the specified port. Digipeaters can also be used to access a remote node. For example:

ROUTE 1 WX9APR-2 + 254

will lock a route in to WX9APR-2 on port 1 (the RS-232 port) with a route quality of 254. Conversely, :

ROUTE 1 WX9APR-2 - 254

will "unlock" a route to this node and will rely on node broadcasts to obtain a quality number and other routing information.

STATS

This command can retrieve valuable information about the health of the node and it's performance. There are no parameters to specify when issuing this command. It's output consists of 6 sets of numbers from the node's 4 levels of operation. Those marked L1 are the receiver and transmitter (the physical layer), L2 (the protocol layer) that shows AX.25 statistics, L3 and L4, the layers responsible for network transport. Here's a typical output from the STATS command:

#WAPR2:WX9APR-4} Statistics
L1 Tx % : 1 0 1 2 0 0
L1 DCD% : 3 0 1 4 0 0
L1 RxOvr: 0 0 0 0
L1 TxUnd: 0 0 0 0
L2 RxCRC: 5 0 5 0
L2 heard: 69 67 57 103
L2 recvd: 22 0 0 0
L2 sent : 67 45 61 58
L2 RxRNR: 0 0 0 0
L2 RxREJ: 0 0 0 0
L2 TxRNR: 0 0 0 0
L2 TxREJ: 0 0 0 0
L2 fails: 0 0 0 0
L3 g'wyd: 0 0
L4 recvd: 11 0
L4 sent : 8 0
Buffers : 613 634 614 594 632 633
CPU loop: 1218 1259 1253 1244 1259 1259
Timers : 13159 44169
L1 flush: 0 44

Let's look at the L1 statistics first. The first two statistics show activity over the past hour, divided into six 10 minute groups. Here's the definitions of the abbreviations you see above.

L1 Tx % - the percentage of time the transmitter was active
L1 DCD% - the percentage of time the DCD was active
L1 RxOvr - the number of receiver overruns
L1 TxUnd - the number of transmitter underruns

L2 statistics are divided into two groups of two numbers. The first pair shows activity over the past hour, with the first number indicating port 0, the second indicating port 1.

Ditto for the second group, except that these numbers apply to the second hour. Here's what the abbreviations mean:

L2 RxCRC - The Frame Checksum Error count

L2 heard - Total number of packets received
L2 recvd - The total count of heard packets that were sent to the node
L2 sent - Total number of sent packets
L2 RxRNR - "Receiver Not Ready" Packets received count
L2 RxREJ - Number of "Reject" packets received by the node
L2 TxRNR - Count of "Receiver Not Ready" packets sent by the node
L2 TxREJ - number of "Reject" packets sent by the node
L2 fails - Timed-out AX.25 links count

L3 and L4 statistics are displayed in two number groups, one for this hour, the other for the past hour. There are two:

L3 g'wyd - shows the number of packets that were handled as network node connections and transported to other nodes (gatewayed)

L4 recvd - Number of packets that were received for node connections that originate of terminate at this node

L4 sent - Number of packets that were sent for node connections that either originate or terminate at this node

The two statistics that follow the L4 stats show the TNC's load. These are displayed the same way that the L1 stats are shown, 6 representing 10 minute intervals. Here's their definitions:

Buffers - Shows the number of free buffers, indicating buffer overruns

CPU loop - Displays the time it took for the CPU to make it through a "loop" of the X-1J program.

Buffers normally run somewhere around 600. The fewer the available buffers, the better chance that data will get "lost" by the node.

CPU Loop values vary depending on the clock speed of the TNC. For a 5 MHz clock and no activity, this will be about 470. With about 75% activity, this will drop to the 300's. If the loop value drops much more than that, the CPU is running as fast as it can and there is a probability of data loss. A 20 MHz Node model Paccomm TNC shows a loop value of about 1200 on a lightly loaded channel.

The ***Timers*** display shows the number of hours of operation on each port. The first set is for Port 0 (the radio port), the second is port 1.

Lastly, there is the ***L1 Flush*** value, also shown for both radio and serial ports. A "Flush" happens when data that is queued and ready to be sent in stead is "flushed out" and not sent.

That's all for this part. In the next part, we'll continue our alphabetical exploration of the Sysop commands for these nodes. Until then, 73 from Andy.

Net/Rom Node Information for the Sysop - Part Fourteen

by Andy Nemec, KB9ALN

This is part 14 of a series designed to help node Sysops learn more about the popular TheNet X-1J series of nodes. This month, we'll finish up our alphabetical exploration of these commands with the **SYSOP**, **TALK**, **UI** and **USERS** commands. While TALK and USERS are technically user commands, there are some additional features associated with these commands that you may want to be familiar with.

SYSOP

This command allows one to enter the Sysop mode to change node parameters. Node operating paramters can't be changed by users unless they enter into this mode. A sysop enters this mode by sending the appropriate password in the appropriate format.

First, a password phrase has to be worked out and "burned" into the EPROM that the X-1J firmware resides on. The password phrase can be nearly any combination of letters, numbers or printable symbols. No limit is specified on the length of the password, although 80 characters seems to be the limit. When picking the password, don't use characters that the node uses for command, like *** ? / . , +** or **-**.

Each letter of the password phrase is numbered. When the sysop prompts the node for the password by sending "***SYSOP***", the node returns with a set of numbers. If the Sysop sends back the appropriately numbered characters of the password phrase (this is case-sensitive, by the way), the node then allows Sysop commands to be accepted and acted upon.

There are two ways to make the password phrase virtually undetectable. First, choose a long one that is not real words. Second, you can send additional "fooler" characters between the legitimate password characters. The node scans the incoming text for the correct characters that can be embedded in text, in the correct sequence (not necessarily next to each other). We won't use any examples here, for obvious reasons. Consult the X-1J documents to learn more about this.

TALK

This is a nice conference feature that allows packet operators to engage in a "round table". To enter the talk mode, the user sends the phrase:

TALK

and the node responds with:

WX9APR-1:WAPR1} Entering Talk. Send /EX to exit.

Other participants will see your call sign and a message that you've joined them. Text from other stations is preceded by their respective call-signs. When a station leaves this mode, a message informs the group of this fact. Unlike some converse bridges you'll find on BBSs and the like, there is only one conference "channel", used by everyone who is engaged in the "Talk" mode. That's the simple, straightforward way to use TALK. There is another way to use it, however.

You can send a broadcast message inviting other node users who aren't engaged in the conference session to join you. When you're engaged in the session, type:

TALK Conference in session, if anyone wants to join in, type 'Talk'

Anyone who is connected to the node but idle will see the message.

There are some limitations to this conference feature, however. First involves the number of users that can successfully engage in the conversation. This seems to be limited to about 4 or 5 at 1200 baud, depending on the settings of timers on the participant's TNCs. There is simply not enough time to send and acknowledge every packet sent in a typical keyboard conversation before the appropriate timers expire and the connection is closed. For that reason, it's best to suggest to your users that they use the TALK feature at off-peak times.

Another limitation is the text buffer. Anything past the 80th character will be dropped into the "bit bucket". For those terminal and packet programs that do not have "word wrap", this may present a problem. You may want to tell your users to change their "Paclen" to 70 or so. Also suggest that they find the command to add a Carriage Return/Line Feed character to each packet. This is necessary for the node to know how to format the text.

UI

This is another user command that is useful for the sysop to know about. **UI** allows a connected user to send an unproto packet through the node. The packet appears as another SSID of the user. This can be good for sending out a LAN-wide announcement or invitation for a connection. This command, like most user commands, can be disabled from the sysop mode with:

UI -

Likewise, it can be enabled with:

UI +

All text intended to be sent out as UI needs to end with a Return (Carriage Return/Line Feed), or it won't be sent.

USER

This command is a user-level command from which Sysops can get some useful information. When the **User** command is given, you may see something like this in return:

WAPR1:WX9APR-1} TheNet X-1J4 (633)

Uplink (WX9ABC)
Downlink (WX9DEF)
Circuit (WAPRBBS:WX9APR KB9ALN)

First, you see that the node firmware version is displayed along side the node's call and alias. Secondly, you see the type of connection, "Uplink", "Downlink", or through "Circuit". In the above example, you see that WX9ABC is connected to WX9DEF through the node, and that KB9ALN is connected to WAPRBBS through the network, using this node as one of the links. Normally, you'd see something like this.

However, there is something else you may see from the User display, the node's "***Choke***" status. A node connection is "Choked" when there is data waiting to be sent, but can't be accepted by the destination station or node for one reason or another.

In uplink or downlink situations, you will see the letter "C" next to the choked station or node. For circuit connections, you will see something like this:

Circuit (WAPRBBS:WX9APR <----+> GRBBBS:KB9ALN)

Notice the + sign? This means that the connection to GRBBBS is choked on that end. In reality, this isn't always a + sign, it can be some other character. The importance is the difference between the --- and some other character. If there is a different character on both ends, then the connection is choked on both ends.

This can be very useful when trying to find the cause of slow connections. One choked node can slow an otherwise

good network path. Choking can also be caused by a poor path, of course. Or it can be caused by a "slow" or faulty TNC.

That's all for our alphabetical exploration of the Sysop commands for these nodes. Next time, we'll explore what is needed to set up a node for TCP/IP operation. Until then, 73 from Andy.