Setting up a Packet–Internet Gateway

Warren Toomey VK1XWT – April 1994

Email: wkt@csadfa.cs.adfa.oz.au

Abstract

This paper discusses the reasons for setting up an amateur packet radio to Internet gateway, technical and hardware features of such a gateway, and security aspects of maintaining a gateway.

Several sections of this paper require some knowledge of amateur packet radio operation, Internet site operation, connecting hardware, and Internet security. These sections will be marked, and I will try to summarise each section in a manner that does not require this knowledge.

This paper is intended for people who have to install a packet-Internet gateway, people who must maintain a packet-Internet gateway (both at the software and hardware level), people who manage a set of Internet machines, and people who provide the facilities for a set of Internet machines (such as a University). This is a wide range of readers, so I apologise in advance if you find some of the material too far below or above your level of understanding.

1 Introduction

The Internet is a large network of academic and commercial computers in the United States that provide a number of communication services (such as electronic mail and file transfers) to its users, via the TCP/IP network protocols. This network is connected to other networks, such as the AARNet in Australia, and networks in Europe and Japan, to provide a global computer network.

The amateur radio service consists of a large number of individuals (known as amateurs or Hams) throughout the world who use radio communications to pass messages, chat, do research into aspects of radio communications and exchange information with other amateurs on several radio frequencies that are allocated for their use. To become an amateur, an individual must undergo an examination which tests the individual's knowledge of the technical operation of radio equipment, radio propagation, electronic theory, as well as legal and other aspects of amateur radio. Thus amateurs form a highly technical, capable and responsible group of people, who are able to provide communication during emergencies, natural disasters, as well as for their own use.

The amateur radio service is also actively involved in finding new and more efficient forms of radio communication - techniques such as single sideband radio and reduced spectrum radio were invented by amateurs. In the last decade or so, amateur radio operators have been involved in researching new forms of digital radio communications. This research has led several amateurs to experiment with different communication protocols for digital radio communications, including the TCP/IP protocols used on the Internet (see Appendix A).

The Internet protocols have been found to be very useful for radio networking, and are now used widely by amateurs. To this end, a block of 16,387,064 Internet addresses has been allocated to the worldwide amateur radio service.

Fast and effective digital radio communications is limited over large distances, due to the distortion and noise introduced by the atmosphere, and the curvature of the Earth. Amateurs have sought several methods to overcome these limitations, such as amateur satellites, which are designed and built by amateurs, and provide store and forward message capabilities as well as digital repeater services.

Another method of overcoming the distance limitation is to use the Internet itself to route digital messages between distant amateur stations. This is easy to accomplish as amateurs are able to use the same network protocols, TCP/IP, as used on the Internet. This method of long-distance amateur communication has been implemented at several places in the world: Australia, America, Switzerland, the Netherlands and Canada, and has been found to be very simple to set up and operate, and is very reliable.

There are two legal/policy aspects of these amateur radio–Internet gateways. The first is the amateur radio service enforces that amateur communications must be between two (or more) licensed amateurs: therefore, non-amateur Internet users must not be able to use the gateways to transmit messages. The second is that many AARNet/Internet sites' policy is to prevent non-registered users from accessing local/Internet resources: therefore amateurs may only pass messages to other amateur gateways. The software that controls the packet–Internet gateways has been written to solidly enforce these two conditions.

2 The Gateway

A packet–Internet gateway obviously must be able to communicate to both the Internet, and the amateurs in its local area. Therefore, it must have at least two **interfaces**, one to the Internet, and one to the amateur community. This is shown in Figure 1.

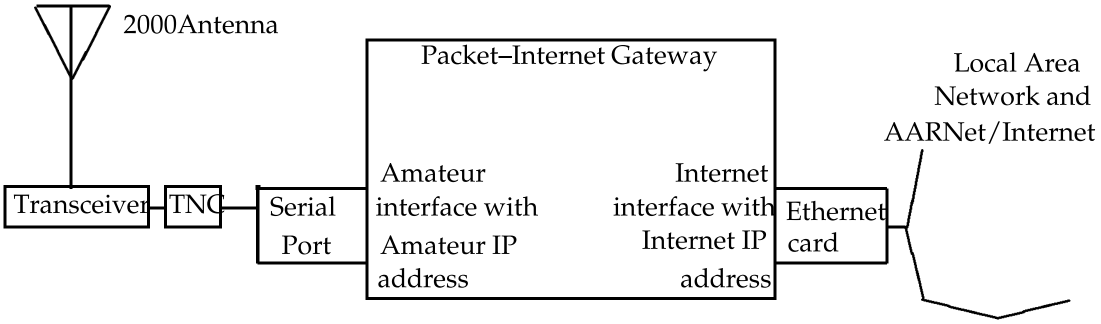

Figure 1: Packet–Internet Gateway Block Design

The gateway is a machine that runs the software that enables it to perform its function. Currently, this means that it must be an IBM PC computer or clone, running the latest version of the NOS software by Phil Karn, Gerard van der Grinten, and other flavours. The software itself is freely available, but is copyright by the software authors. For performance reasons, I would recommend a computer with at least the performance of a 10MHz IBM XT, with 640K or memory. An AT of course would be better. If you plan on providing file transfer or mail access, a hard disk is needed.

Each interface must have a unique IP address, so that both the local amateurs and the other gateways can communicate with it. The Internet interface's address is assigned by the AARNet/Internet administrator of the institution, and the amateur radio interface's address is assigned by the local amateur IP administrator.

Each interface must also have the appropriate hardware needed for it to communicate with its destination. On the amateur radio side, a **Terminal Node Controller**, a **transceiver** and an **antenna** convert the packets from the gateway into radio transmissions that can be received by amateur stations. On the Internet side, there are two main options: an **Ethernet card** connects the gateway to the local area network, which is part of the AARNet/Internet, or a **serial line** connects the gateway to another machine in the institution which is on the local area network, and hence is part of the AARNet/Internet. The serial line can run either the SLIP or PPP protocol to communicate with the other machine.

That completes the overview of the hardware setup of the gateway. This leads on to several considerations:

- The gateway and other hardware must be reliable, and be readily available to the maintainer of the gateway to fix unforseen problems.

- The gateway must be placed in a physical position to suit both its connection to the AARNet/Internet, and to the TNC/transceiver.
- The transceiver and antenna must be in a position to satisfy both the needs of the local amateur community, and the wishes of the institution where they are to be situated.
- If using Ethernet, the gateway's position must allow it to be physically connected to the local area network.

At this point, it should be mentioned that at least one of the people maintaining the gateway must have a current amateur radio license. The institution providing the Internet connection may also wish to appoint another person to ensure the gateway meets the policies of the institution; this of course may be the same person.

3 How the Gateway Works

At an elementary level, the gateway works as follows: Messages are broken up into **packets** when they are sent over the Internet. When the gateway receives a packet on one interface, it checks the destination IP address in the packet, and according to its table of **routes**, it retransmits that packet on the interface that will send the packet to its destination. For example, a packet from the local amateur radio community to a gateway in another country will be received by the local gateway, retransmitted by the gateway on its Internet interface, and will travel along the Internet to the destination gateway, where it will be received and retransmitted by the transceiver at that gateway.

In practice, this is not as simple, because of the following reason. Even though the global amateur community has been allocated a block of valid Internet addresses, none of the Internet machines in the world know how to send packets to these addresses. This is because up until recently, no amateur computers were connected to the Internet.

The easiest solution would be to advertise the routes to these addresses to all Internet machines, but this was not done for the following reasons:

- Amateur stations are not as geographically fixed as most Internet computers, as their computers are generally smaller, and the radio medium allows amateurs to move around and still communicate with each other, and
- Advertising amateur computer addresses to all Internet machines greatly increases the possibility of non-amateurs accessing amateur transmitters.

The solution adopted was to **encapsulate** amateur packets received by the gateways inside another packet, transmit this bigger packet to the destination gateway, which unwraps the packet and transmits the contents [Woodburn et al. 1991]. This is illustrated in Figure 2.

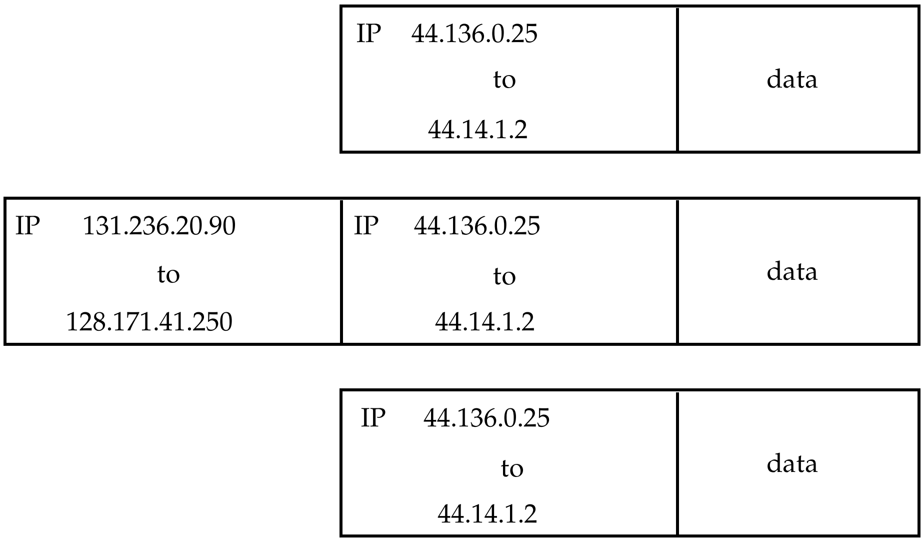

Figure 2: Amateur Packet Encapsulation

The first packet shown is the packet transmitted by an amateur and received by the local gateway. At this level, the source and destination address are both amateur addresses. The local gateway determines which remote gateway this packet must be passed to, and **encapsulates** the packet inside a new packet, with source and destination addresses being the Internet addresses of the local and remote gateway. The packet is then placed on the 2000Internet, where it is passed to the remote gateway.

This encapsulated packet (the second packet in Figure 2) is seen by the Internet to have a valid source and destination address, and can be passed to the remote gateway. The Internet sees the wrapped packet as data, which it ignores.

At the remote gateway, the packet is received and unwrapped; this is the third packet in Figure 2. The remote gateway determines which interface to transmit the unwrapped packet, according to the destination address. Reply messages are encapsulated in exactly the same manner.

4 AX.25 Encapsulation

One of the first digital protocols used by amateurs, and still the most widely used, is AX.25, a reliable protocol based on X.25. One of its limitations is that AX.25 packets cannot pass through more that 7 intermediate machines before reaching their destination. This limits the distance AX.25 packets can travel.

The latest versions of NOS include AX.25 encapsulation, which operates in exactly the same way as normal IP encapsulation, thereby allowing long distance AX.25 communication while still obeying the 7 step limit [Kantor 1991].

5 Preparing the Gateway Machine

This section discusses the aspects of preparing the gateway for the software that will make the gateway work. It is assumed that:

- You have a fairly good idea of the Internet and how it works.
- You have a reasonable understanding of the NOS software. If you don't, you can either obtain a 'plug 'n play' disk from me by mailing me a note to my email address at the beginning of the paper, or you can anonymous ftp to the machine `minnie.cs.adfa.oz.au`, cd to `hamradio/packet/plugplay`, and get the software from there.
- You have a copy of the NOS software. At the time of writing, the NOS flavour most used on gateways is Johan Reinalda's JNOS, and the current version is 1.07. Other NOS versions used are Phil Karn's KA9Q, and Gerard van der Grinten's PA0GRI. The latest version of these NOS flavours can be obtained from `ucsd.edu` or `grivel.une.oz.au`.
- You have a gateway that meets the hardware specifications given in the Introduction, and the hardware needed to form the interfaces – that is, a TNC, a transceiver, an antenna, and an Ethernet board or a serial link to a machine on the Internet that can talk SLIP or PPP. The gateway will also need around 1 Megabyte of free disk space if it is to provide file transfer or mail access.
- You have a local amateur IP address for the machine, and a local Internet IP address for the machine.

Your first step is to give the gateway two domain names, a name for the other gateways to know, and a name for the amateurs to know. Your Internet name should follow any rules specified for your institution. Your amateur domain name should be of the following form: `name.callsign.ampr.org` where the callsign is the callsign of the amateur that will be operating the gateway, and the name is whatever name you want to call the machine. An example of Internet and amateur domain names is `minnie.cs.adfa.oz.au` and `minnie.vk1xwt.ampr.org`.

You need to register the Internet name on the Domain Name Server for your local area network. You then need to register your amateur domain name with the Domain Name Server for the amateur service. This is the DNS server at `ucsd.edu`. To do this, send email to the following address: `ampraddr@ucsd.edu`. The subject line should be empty, and the email should have the following line:

```
name.callsign             IN    A   amateur.radio.ip.address
```

For example, the line I sent in for my gateway was:

```
minnie.vk1xwt             IN    A    44.136.7.129
```

Aside: For those who understand DNS records, `ucsd.edu` will also accept CNAME and MX records. Also, you can delete previous entries by sending the same line, with the last field on the line the word 'delete'. Additions and changes to the DNS at `ucsd.edu` usually take a few days to be entered, so make sure you get things right first.

6 Installing the NOS Software

Installing the NOS software can be done in three stages:

- Install the software, and set the gateway up as a local amateur radio station; this includes setting up the TNC, transceiver and antenna. This should not be too hard to do, as the local amateur community can provide information on how to do this.
- Alter the setup to include an interface to the AARNet/Internet, using either an Ethernet board, or a serial link running SLIP or PPP. You will need an appropriate Ethernet driver if you are using an Ethernet board – these can be obtained from `ucsd.edu` in the directory `hamradio/drivers`. Read the documentation that comes with the driver package.
- Finally, set up encapsulated routes to the other amateur gateways, and email their operators so that they will set up routes back to your gateway. Use the 'ping', 'telnet' and 'ftp' commands in NOS to ensure that the encapsulation is working.

Example setup files are given in the appendices , and are currently in use at the `minnie.cs.adfa.oz.au` gateway. This section will discuss aspects of a gateway's setup that I think need attention.

To begin with, the hostname of the gateway should be the gateway's amateur domain name, as it will be used mainly by amateurs. Ensure that the gateway has a valid amateur callsign. Every interface used by the 2000gateway should use the amateur IP address, except for the single interface to the Internet, which should use the Internet IP address. The default route should be the Internet interface, as badly addressed packets will not affect anybody there. Ensure that routes are as specific as possible: for example, if your local amateur address are of the form $44.136.0.x$, then you should add a route to `44.136.0/24`.

6.1 Adding Encapsulated IP Routes

Adding the encapsulated IP routes to other gateways is very easy, as is shown in the example setup file in the appendices. Note that you must specify the Internet IP address of the gateway, and not the amateur IP address.

6.2 Adding Encapsulated AX.25 Routes

Due to the implementation aspects of AX.25 encapsulation, each encapsulated AX.25 route appears as a new **interface** under NOS, and as such, needs to have a new **SSID** (Station Sequence ID), as well as a new interface name. For example, the pseudo-interfaces used by the `minnie` gateway have SSIDs vk1xwt-2, vk1xwt-3, vk1xwt-4, vk1xwt-5 and vk1xwt-6. These are different than the callsign used for the gateway, vk1xwt-1. According to the AX.25 protocol, this allows up to 15 pseudo-interfaces, with SSIDs of –1 to –15, as well as the local amateur interface.

For AX.25 encapsulation to work, you need to already have an encapsulated IP route defined to the destination gateway, and vice versa.

When using the encapsulated AX.25 service, you must remember that you need to use the SSID of the interface you want to connect through. For example, to connect to `minnie` locally, you would do:

```
net> connect vk1xwt-1
```

However, to connect to a machine in Hawaii through `minnie`, you need to use `minnie`'s SSID of –4:

```
net> connect ah6bw-1 vk1xwt-4
```

Similarly, a user in Hawaii connects **to** `vk1xwt-4`, but connects **through** `vk1xwt-1`.

7 Security Considerations – Packet Filtering

One obvious problem with a gateway is that packets from Internet machines may escape out onto amateur frequencies. This problem is usually minimised by not advertising routes to amateur frequencies to Internet machines. However, this is security through obscurity.

New versions of NOS have solved this problem by adding packet filtering, known as 'IP Access'. WIth IP access, the gateway can allow or refuse to transmit a packet on any interface according to the source IP address in the packet. Thus, access to amateur frequencies can be denied to all packets who have non-amateur IP addresses, i.e IP addresses that are not of the form 44.X.Y.Z.

IP Access is described more completely in the appendices and in the NOS documentation.

8 Security Considerations – Services

If you use your gateway with no services running, you can guarantee the security of your gateway. However, with several services, the mailbox, SMTP, RIP and the converse service, your security can be breached. Fortunately, these services can be secured as described below.

8.1 RIP

RIP is the routing protocol that is used to pass routing information around between machines on the Internet. Strictly speaking, a packet–Internet gateway does not need to run RIP, as all of its routes are statically defined. The only reason for a gateway to run RIP is to inform and/or learn from other gateways when new gateways become available.

If you think that using RIP is a good idea, you must ensure that:

- You do not broadcast your routing table to your local network; if you do, non-amateur machines will learn about routes to the amateur gateways, and may allow non-amateurs to use amateur transceivers.
- You refuse all RIP broadcasts from your local network; if you do not, amateurs will learn about the Internet addresses on your local network, which are non-amateur addresses.
- You only send your routing table to destination amateur gateways, as they are the only machines who are interested in your routes.

I would receommend **not** running RIP at all.

8.2 The Mailbox

The mailbox service can be accessed in two ways: by a local amateur connecting to your machine using the AX.25 protocol, or by an amateur or non-amateur connecting to your machine by the Telnet protocol.

The mailbox provides features that you do not want amateur or non-amateur users to access:

- A Telnet capability which can connect to both amateur and non-amateur machines.
- A Gateway capability which can connect using AX.25 to amateur machines.
- A Netrom capability which can connect using NETROM to amateur machines.
- A Remote operation capability which allows a user to change the current setup of the gateway.

Fortunately, these are easily disabled by including an entry in the `ftpusers` file for the user 'anonymous', with permissions '1', no password, and a home directory the directory `/pub`. Make sure that the `ftpusers` file is not kept in the `pub` directory, or any of its subdirectories.

You may wish to add other users entries, so that you can remotely Telnet to the gateway and modify its state. These entries should have a password, and you should never Telnet to the gateway from another amateur machine, as the password will be broadcast over the airwaves for other people to hear.

8.3 SMTP

SMTP is the another way to bypass security, as it allows non-amateurs to send mail to amateurs, and vice versa, and SMTP cannot be easily set up to prevent this.

You may initially have SMTP enabled, and monitor the use of email to see if there is any current security problems; however, this leaves the gateway open for email interchange which may be illegal.

8.4 The Converse Service

The converse service is similar to a DX-Cluster or an Internet Relay Chat: local users login to the converse server, which is itself connected to other server, and the text sent by local users is passed to all the other converse server users.

Again, this is a security problem, especially as many converse servers are run on amateur-only machines. Thus, Internet users could log into one converse server, and then transmit text via amateur frequencies to amateur-only machines.

9 Security Considerations – Connection Filtering

You will be relieved to know that in new versions of NOS there is connection filtering as well as packet filtering. Connection filtering allows the gateway to accept or refuse a connection to a single or range of services according to the IP address attempting the connection.

This filtering, known as 'TCP Access', allows insecure services such as SMTP and the converse service to be restricted to amateur-only IP addresses, i.e only 44.X.Y.Z addresses. Of course, if you believe that other services are insecure for one reason or another, you can use TCP Access to deny connections to them as well. More details on TCP Access are given in the appendices and in the NOS documentation.

10 Conclusion

To summarise, the packet–Internet gateway provides fast and reliable long-distance digital amateur radio communication, which up until recently has been difficult if not impossible. The use of TCP/IP protocols allows easy connection to the AARNet/Internet, as well as the local amateur community. This long-distance connection not only provides reliable communication for amateur and emergency use, but gives the amateur radio community another avenue for research into radio communication techniques.

The gateway software provides the method to connect amateur gateways together, while at the same time ensures the gateway can be operated correctly from both the amateur and non-amateur point of view.

I have operated and maintained a packet–Internet gateway in Canberra since April 1991, and have found it to be very reliable, very easy to maintain, and very secure. Moreover, due to the communication speeds used in the amateur community, the gateway has added a negligible burden to the AARNet/Internet use of my local network.

I would be glad to answer any questions that are emailed to me at the Internet mail address given at the beginning of this paper, although I cannot guarantee the promptness of my replies. I would also greatly appreciate comments, criticisms and suggestions about this paper and how it can be improved.

Warren Toomey vk1xwt.

A Bibliography

This bibliography includes the papers describing the encapsulation protocols used by a packet radio–Internet gateway, as well as some example papers published by amateurs in the field of packet radio.

[Diersing et al. 1989] R.J.Diersing, J.W.Ward. *Packet Radio in the Amatuer Satellite Service.* IEEE J Selected Areas in Communications, Vol 7 No 2. February 1989.

[Enmore 1990] G. Enmore n6gn. *Physical Layer Considerations in Building a High Speed Amateur Radio Amateur Network.* ARRL/CRRL Amateur Radio 9th Computer Networking Conference. London, Ontario. Canada. September 1990.

[Flaherty 1988] P. Flaherty n9fzx. *Digital Radio Networks and Spectrum Management.* ARRL/CRRL Amateur Radio 7th Computer Networking Conference. Columbia, Maryland, America. October 1988.

[Geier et al. 1990] J. Geier, M. DeSimio wb8mpf and B. Welsh kd8wg. *Network Routing Techniques and Their Relevance to Packet Radio Networks.* ARRL/CRRL Amateur Radio 9th Computer Networking Conference. London, Ontario. Canada. September 1990.

[Ioannidis et al. 1991] J. Ioannidis, D. Duchamp and G. Maguire Jr. *IP-Based Protocols for Mobile Networking.* ACM SIGCOMM '91 Conference. Zurich, Switzerland. September, 1991.

[Kahn et al. 1985] P.R.Kahn, H.E.Price, R.J.Diersing. *Packet Radio in the Amateur Service.* IEEE J Selected Areas in Communications, Vol 3, No 5. May 1985.

[Kantor 1991] B. Kantor wb6cyt. *Internet Protocol Encapsulation of AX.25 Frames.* Request for Comments memo 1226. May 1991.

[Karn 1990] P. Karn ka9q. *MACA – A New Channel Access Method for Packet Radio.* ARRL/CRRL Amateur Radio 9th Computer Networking Conference. London, Ontario. Canada. September 1990.

[Neben 1983] B. Neben k9bl. *Packet Radio for Emergency Communications.* ARRL Amateur Radio 2nd Computer Networking Conference. San Francisco, California. March 1983.

[Sproul et al. 1990] M. Sproul kb2ici and K. Sproul wu2z. *Long Distance Packet Mail via Satellite.* ARRL/CRRL Amateur Radio 9th Computer Networking Conference. London, Ontario. Canada. September 1990.

[Woodburn et al. 1991] R. Woodburn and D. Mills. *A Scheme for an Internet Encapsulation Protocol.* Request for Comments memo 1241. July 1991.

B Existing Packet–Internet Gateways

Following is a table of the existing gateways as at the 28th of April, 1992.

Area	Operator	Email Address
Canberra, Australia	Warren Toomey vk1xwt	wkt@csadfa.cs.adfa.oz.au
Newcastle, Australia	Dave Walmsley vk2xpx	ccdrw@cc.newcastle.edu.au
Sydney, Australia	Terry Dawson vk2ktj	terryd@extro.su.oz.au
Melbourne, Australia	Peter Hallgarten vk3ave	vk3ave@csource.oz.au
Queensland, Australia	Andy Joyce vk4kiv	joyce@qut.edu.au
Central Virginia, USA	Jim De Arras wa4ong	jmd@emperor.handheld.com
Parts of Europe	Marco Zollinger hb9cat	marco@srztm601.alcatel.ch
Ottawa, Canada	Barry McLarnon ve3jf	barry@dgbt.doc.ca
Ontario, Canada	Roger Sanderson ve3rks	rsanders@sunee.uwaterloo.ca
Hawaii, USA	Antonio Querubin ah6bw	tony@mpg.phys.hawaii.edu
Pennsylvania, USA	Joe Reinhardt af2j	jmr@ecl.psu.edu
South Texas, USA	Kurt Freiberger wb5bbw	kurt@cs.tamu.edu
San Antonio, USA	Jack Spitznagel kd4iz	kd4iz@giskard.uthscsa.edu
Chicago, USA	Bob Van Valzah ke9yq	bob@imsa.edu
Indiana, USA	Dwight Hazen wb9tlh	hazen@hazen.ucs.indiana.edu
Illinois, USA	Chuck Henderson wb9uus	chuck@bradley.bradley.edu
Illinois, USA	Jay Freeman wt9s	freeman@eagle.sangamon.edu
Colorado, USA	Bdale Garbee n3eua	bdale@gag.com
Houston, USA	Remi Hutin fe6cnb	hutin@asl.slb.com
nevada, USA	Bill Healy n8khn	healy@moriah.unr.edu

C Example NOS Configuration Files

This section examines a set of example configuration files for NOS on the gateway minnie.cs.adfa.oz.au. Note that not all gateways have the same requirements or offer the same services, and not all use the same variety of NOS. Therefore, take the examples given here as a guide for your own files.

C.1 Introduction

In order to understand the configuration files presented here, we need to first examine the purpose of the gateway, its services, and the software it uses.

Minnie.cs.adfa.oz.au is an Internet-AMPRnet gateway which routes packets between itself and other gateways via encapsulation. It also offers the following services:

- Anonymous file transfer with the **ftp** service.
- Logins via AX.25 or telnet to the NOS mailbox.
- Mail delivery using the **smtp** service.
- Determination of the current users of the gateway with the **finger** service.
- A **converse** service that is the TCP/IP equivalent to a DX-Cluster.

It should be noted that the mailbox in particular provides more services than just sending and receiving mail; mailbox users may be permitted to telnet or AX.25 connect out of the mailbox, use the finger and converse services, and upload/download files. Ensuring reasonable privileges to mailbox users will be dealt with later.

The most important service is that of encapsulating AMPRnet packets destined for other gateways, and vice versa. Unlike most ordinary TCP/IP hosts, a gateway has two or more interfaces, and at least two IP addresses. Minnie has the following addresses and domain names:

- minnie.cs.adfa.oz.au [131.236.20.70]
- minnie.vk1xwt.ampr.org [44.136.7.129]

and three interfaces:

- An ethernet interface (`ec0`) for Internet users
- A KISS AX.25 interface (`ax0`) for AMPRnet users, and
- A pseudo-interface (`encap`) which performs the encapsulation.

With the encapsulation correctly set up, minnie can be accessed by all Internet users as minnie.cs.adfa.oz.au, and by all AMPRnet users as minnie.vk1xwt.ampr.org.

Packet encapsulation is a two-way process. Minnie services the range of AMPRnet addresses 44.136.7.128 to 44.136.7.255, i.e 44.136.7.128/25. Other gateways who wish to route packets to this AMPRnet address range must include an **encapsulated route** so that packets will be encapsulated and then routed to minnie's Internet address, i.e

```
route addprivate 44.136.7.128/25 encap minnie.cs.adfa.oz.au
```

Similarly, minnie must have a corresponding set of encapsulated routes back to all the other gateways, e.g

```
     ...
route addprivate dest_AMPRnet_range encap dest_gateway
     ...
```

The list of encapsulated routes to all known gateways is available via anonymous ftp from minnie.cs.adfa.oz.au in the 'gateways' directory, as the file 'gateways.XXX', where 'XXX' are digits.

C.2 Booting the Gateway

Gateways are usually run as turn-key machines; the box is turned on and the NOS software is automatically started. Therefore, in the gateway's boot file you need to do several things:

- Set up some environment variables used by NOS
- Install the packet drivers you need for NOS
- Run NOS

Here is an example `autoexec.bat`:

```
@echo off
PATH c:\bin;c:\ka9q\bin;
rem
rem     Set up some environment variables for NOS
rem
rem set TMP=C:\tmp
rem set TEMP=C:\tmp
rem
rem     Set up miscellaneous environment variables
rem
set USER=warren
se!2000!t TERM=nansi
loadhigh \ka9q\bin\wd8003e 0x70 -o 5 0x280 0xc800
nos
```

On minnie, all of the gateway files and programs are kept under the `KA9Q` directory. The programs and batch files are kept in `KA9Q\BIN`. NOS uses the `TMP` or `TEMP` variable to determine where it creates temporary variables: I can't remember which one so I set both.

The packet driver for minnie's WD8003E card is then loaded. Each packet driver has a slightly different calling convention: here the driver is loaded to use software interrupt `0x7e`, hardware interrupt 5, with the card's registers at `0x280` and the card's buffer memory at `0xc800`. Only the interrupt numbers are of interest to NOS.

Finally, a batch file kept in `KA9Q\BIN` called `NOS.BAT` is executed.

C.3 The NOS Batch File

The NOS batch file usually runs as an infinite loop, restarting NOS if ever it is terminated: this is useful for when the gateway must run unattended. The batch file cleans up the lock files from the mail subsystem (in case NOS was abruptly terminated). If the gateway has logging turned on, the last log file is appended to a cumulative log file, and then zeroed. Thus a gateway administrator can look in `nos.log` for logging information since the last NOS restart, and in `nos_old.log` for the remaining log information.

```
@echo off
:Again
cd \ka9q
set TZ=EST14
del \ka9q\spool\mqueue\*.lck
del \ka9q\spool\mail\*.lck
type nos.log >> nos_old.log
del nos.log
copy dom.txt domain.txt
rem \ka9q\bin\oldnos.exe -d /ka9q minnie.net
\ka9q\bin\jnos107b.exe -f minn107.cfg
goto Again
```

One line to note is that the `domain.txt` file (which holds TCP/IP name/address pairs) is recreated from the `dom.txt` file. This is done because some versions of NOS (for example GRINOS), would rewrite the file, removing any comments. This ensures that an original version is kept intact.

Finally, the NOS program is started. This can be done in two ways. With some NOS flavours, you specify the directory where all the configuration files are kept, and the NOS initialisation file, e.g

```
\ka9q\bin\oldnos.exe -d /ka9q minnie.net
```

With these flavours, all of the NOS service configuration files are kept in fixed locations in the NOS directory, which can be inflexible. In new NOS flavours, e.g JNOS 1.07, a directory configuration file is first read to inform NOS of the location of the NOS service configuration files, e.g

```
\ka9q\bin\jnos107b.exe -f minn107.cfg
```

If you have a NOS flavour that performs this, go on to the next section. If not, skip the next section.

C.4 The Directory Configuration File

As mentioned, this file is read by newer versions of NOS to allow you to set up a flexible directory system. I won't attempt to explain every line; see the NOS documentation for your NOS flavour for more information. The most important line is the one which tells NOS where the NOS initialisation file is kept:

```
Startup = /ka9q/minnie.net
```

Once this file is read, NOS reads the initialisation file. The rest of this file is left for your perusal.

```
#SAMPLE: these are the defaults used.
#These are new names for the various files and directories used in nos.
#To use them, run nos as 'nos -fnos.cfg'
#If you don't change a particular filename, you should comment out each
#unneeded line, since they allocate memory for the new name.

#lines need to be either comments (starting with #)
#or have a valid 'file=filename' format.
#(lines are NOT case sensitive.)
#all others are ignored; this allows for different compiles to
#use the same files-configuration file...
#both spaces or tabs can be used as separators.
```

```
#921125 - WG7J

#the autoexec file containing system setup
Startup = /ka9q/minnie.107

#the user permission file
Userfile = /ka9q/ftpusers

#the ftp host file for auto-login
#Hostfile = /net.rc

#the mail log file
Maillog = /ka9q/spool/mail.log

#the directory where local mail gets delivered
Mailspool = /ka9q/spool/mail

#the directory where mail gets queued for the smtp daemon to handle
Mailqdir = /ka9q/spool/mqueue

#this should have same path as the above!!
Mailqueue = /ka9q/spool/mqueue/*.wrk

#if you route mail, here is goes
#Routeqdir = /ka9q/spool/rqueue

#the mail alias file
Alias = /ka9q/alias

#the domain.txt file
Dfile = /ka9q/domain.txt


#directory where finger files go
Fdir = /ka9q/finger

#the file where the finger database is maintained
Fdbase = /ka9q/finger/dbase.dat

#the list of areas on the system
#Arealist = /ka9q/spool/areas

#mailbox message of the day
Motdfile = /ka9q/spool/motd.txt

#mail rewrite rules
Rewritefile = /ka9q/spool/rewrite

#user signatures go here
#Signature = /spool/signatur

#Bulletin ID's go here
#Historyfile = /spool/history

#Help files go in this directory
```

```
Helpdir = /ka9q/spool/help

#the user defaults file (created by system)
UDefaults = /ka9q/spool/users.dat

#backup of the above
UDefbak = /spool/ka9q/users.bak

#Convers user info file; notice that default is the same as Fdbase file !
Cinfo = /ka9q/finger/dbase.dat

#pop users are listed in this
#Popusers = /popusers

#FTP message of the day
Ftpmotd = /ka9q/spool/ftpmotd.txt

#NNTP directory
Newsdir = /ka9q/spool/news

#BBS forward file
#Forwardfile = /spool/forward.bbs

#saved netrom routes go here
#Netromfile = /netrom.sav

#these commands get executed on exit
#Onexit = /onexit.nos

#expire command file
#Expirefile = /spool/expire.dat

#NNTP active file
#Active = /spool/news/active

#NNTP pointer file
#Pointer = /spool/news/pointer

#NNTP info
#NInfo = /spool/news/info

#NNTP help
#Nhelp = /spool/news/help

#NNTP message history file
History = /ka9q/spool/news/history

#NNTP forward
#Forward = /spool/news/forward

#NNTP poll
#Poll = /spool/news/poll
```

C.5 The Domain File

Before going on to look at the NOS initialisation file, we need to first examine the domain file. Although NOS can use **name servers** to look up TCP/IP name/address pairs it doesn't know, these are not used while processing the initialisation file. Therefore the domain file (usually `domain.txt`) must contain **all** of the TCP/IP names used by the initialisation file. You may also want to include names that you use frequently, so that NOS won't have to query any name servers to resolve them. Note however, that the domain file is static, and you may need to keep it up to date periodically to ensure that the name/address pairs are correct. Here is minnie's `domain.txt`:

```
# Last updated: 10:00am 921209
#
# Canberra packet stations
#
gw.vk1bud.ampr.org.     IN      A       44.136.0.40
minnie.vk1xwt.ampr.org. IN      A       44.136.7.129
#
# Gateways to other packet stations
#
#
# ADFA machines
#
cserve.cs.adfa.oz.au.   IN      A       131.236.20.1
cs_gate.cs.adfa.oz.au.  IN      A       131.236.20.2
cspyr1.cs.adfa.oz.au.   IN      A       131.236.20.7
joruth.cs.adfa.oz.au.   IN      A       131.236.20.69
minnie.cs.adfa.oz.au.   IN      A       131.236.20.70
convex.cc.adfa.oz.au.   IN      A       131.236.1.1
ccadfa.cc.adfa.oz.au.   IN      A       131.236.1.2
sserve.cc.adfa.oz.au.   IN      A       131.236.1.17
#
#
# Other machines
#
munnari.oz.au.          IN      A       128.250.1.21
#
# Aliases
#
minnie.cs.      IN      CNAME   minnie.cs.adfa.oz.au.
csadfa.         IN      CNAME   csadfa.cs.adfa.oz.au.
cspyr1.         IN      CNAME   cspyr1.cs.adfa.oz.au.
ccadfa.         IN      CNAME   ccadfa.cc.adfa.oz.au.
sserve.         IN      CNAME   sserve.cc.adfa.oz.au.
cserve.         IN      CNAME   cserve.cs.adfa.oz.au.
convex.         IN      CNAME   convex.cc.adfa.oz.au.
minnie.         IN      CNAME   minnie.vk1xwt.ampr.org.
```

C.6 The NOS Initialisation File

The NOS initialisation file is where the gateway's network addresses and interfaces are set up, and where the services on the gateway are started. Some commands must occur before others; in general, I recommend performing things in the following order:

- Define the gateway's main hostname and IP address.
- Describe the memory buffers and memory parameters.
- Specify the interfaces used and their soft/hardware interrupts.

- Specify the interfaces addresses, MTUs and broadcast addresses.
- Detail what domain name servers are to be used.
- Set up routes to the gateway's interfaces, and to other gateways.
- Set up TCP and IP parameters.
- Set up service-specific parameters.
- Start the gateway services.
- Any miscellaneous commands.

Once the initialisation file is finished, NOS falls to the command line prompt as usual. For the rest of this section, we will look at an example initialisation file, minnie's initialisation file, broken into the parts given above.

The gateway's domain name and IP address is given first. You must choose one of the two name/addresses that the gateway has. I use the AMPRnet name so as not to confuse the amateur users; the Internet users can cope.

```
#               A U T O E X E C . N E T     F O R     W G 7 J
#
#
#       Local Definitions
#
hostname minnie.vk1xwt.ampr.org
ip address minnie.vk1xwt.ampr.org
```

The number of interrupt buffers should be set so as to minimise the number of lost packets, especially from an Ethernet. Minnie uses 10 interrupt buffers. Each buffer should be at least as large as the maximum packet size; the default buffer size is 2048 bytes. Minnie uses the 'efficient' form of memory allocation which helps to keep the amount of available memory high.

```
#
#       Memory Parameters
#
# nibufs - number of interrupt buffers
# eff - efficient.  It's a better memory allocation algorithm.
mem nibufs 10
mem eff on
```

The interfaces used by the gateway are now attached. Minnie has one Ethernet interface `ec0` which is a packet driver, and a KISS interface `ax0` using the built-in NOS KISS driver. Note that the soft/hardware interrupt values used by the packet driver must be the same as used when the packet driver was loaded in the `AUTOEXEC.BAT` file.

The largest packet size is also specified on the attach lines, 1500 for the `ec0` interface, and 576 for the `ax0` interface. You should read the NOS documenation for the other parameters.

```
#
#       Attach ports
#
# We have 2 interfaces:
#       ec0 connects minnie to the ADFA net
#       ax0 connects minnie to the local AMPR subnet
attach packet 0x70 ec0 5 1500
attach asy 0x2f8 3 ax25 ax0 1024 576 9600
```

With the interfaces attached, their addresses etc. must now be given. A gateway has two other interfaces: `loopback` is used to route the gateway's packets back to itself (for debugging purposes),2000 and `encap` encapsulates packets and passes them back to the gateway for further routing.

For each interface you should specify the IP address, the maximum packet size and the broadcast address. The latter should have all non-network address bits as 1's. For example, 44.A/16 would have broadcast address 44.A.255.255, and 44.A.B/24 would have broadcast address 44.A.B.255. The encap interface should have the gateway's AMPRnet address, but broadcast should not be specified (it doesn't make sense anyway).

```
#
#       Interface addresses
#
# Although minnie's default ip address is minnie.v1kxwt, we need to use a
# `real' Internet address for the ec0 connection. We also use minnie's ampr
# ip address for encapsulation.
ifc ec0   ipa minnie.cs.adfa.oz.au
ifc ec0   broadcast 131.236.20.255
ifc ec0   mtu 1500
ifc ax0   ipa minnie.vk1xwt.ampr.org
ifc ax0   mtu 576
ifc ax0   broadcast 44.136.7.255
ifc encap ipa minnie.vk1xwt.ampr.org
ifc encap mtu 576
```

With the above done, the only important thing to do next is to set up the routes for packets. However, I usually describe the name servers that are available for domain name to IP address lookups. This was before I found out that NOS doesn't use name servers while in the initialisation file.

You can specify several domain name servers. There are two local domain servers at ADFA, which are specified, as well as the authorative name server for the AMPRnet, ucsd.edu. A maximum lookup wait of 30 seconds is specified. Note that NOS uses the last named server first, so be sure to place your local nameservers last!

Minnie also acts as a caching name server itself: nameserver requests to minnie are passed on to the nameservers minnie knows, and the result is kept in minnie's memory in case it is again requested in the future. Minnie caches up to 200 domain name results.

```
#
#       Domain Servers
#
domain cache wait 30000
domain cache size 200
domain cache clean off
domain addserver ucsd.edu
domain addserver ccadfa.cc.adfa.oz.au
domain addserver cspyr1.cs.adfa.oz.au
dom maxwait 30
dom start
dom trans on
```

Finally, routing! A gateway needs the following routes:

- A route to the local AMPRnet subnet.
- A route to the local LAN subnet.
- A default route via the local Internet gateway.
- Lots of encap routes to other AMPRnet gateways.

Note that the default route is out to the Internet. This is done so that mistakes hit the Ethernet and not the airwaves. The first three routes are usually static, and are included in the initialisation file. The encap routes frequently change, and so minnie loads them from a second file, `encap.txt`. This file can be generated by extracting only those lines with the word 'encap' from the 'gateways.XXX' file. You **must** remember to remove the encap route to your own gateway from `encap.txt`, otherwise weird things will happen.

Routes are specified as private. This prevents your gateway from advertising routes to the Internet, in the unlikely event that you are running RIP.

```
#
#       Routing
#
# The default route is via ec0. ax0 takes the subnet 44.136.7.128/25. Other
# routes are via IP encapsulation to other 44. networks via the Internet.
# See the file 'gateways.xxx' for more details. We now load from a 2nd file,
# encap.txt; this makes altering much easier.
#
route addprivate default         ec0 cs_gate.cs.adfa.oz.au
route addprivate 131.236.20/24   ec0
route addprivate 44.136.7.128/25 ax0
source encap.txt
```

With the routing specified, minnie's packet filtering (IP access) and connection filtering (TCP access) are given. The rule of thumb here is to first specify the range of addresses that are permitted access, then specify the range of addresses denied. If no denials are specified, then the default is to permit all.

The IP access is simple. All AMPRnet addresses can route packets over the `ax0` interface, but all other addresses are denied.

For the TCP access, you must decide which services should be denied and from whom. On minnie, only the converse server running on TCP port 3600 has access denied. TCP access is permitted on AMPRnet addresses and ADFA addresses on TCP ports 1 to 3600. TCP access to port 3600 is denied to all others. Thus, all addresses are implicitly allowed to connect to all ports but 3600.

```
#
#       IP Access - prevent Joe Internet from accessing ax0
#
ip access permit 44/8 ax0
ip access deny all ax0
#
#       TCP Access - prevent Joe Internet from accessing the converse server
#
tcp access permit 44/8 1 3600
tcp access permit 131.236.20/24 1 3600
tcp access deny all 3600
```

TCP and IP parameters can now be detailed. The IP maximum hop count should now be at least 32, as the Internet is large enough to need this. The maximum segment size in TCP should be set to the smallest packet size (576 on `ax0` and `encap`) minus 40.

```
#
#       Set TCP and IP parameters
#
ip ttl 32
#
tcp mss 536
tcp window 2900
tcp irtt 10000
tcp syndata on
```

With the interfaces and routes ready, we can begin on the parameters of the gateway's services. These are all readily explained in the NOS documentation, and I will omit any descriptions here.

```
#
#       SMTP parameters
#
#                       Some mailing list recipients are stuffed!b87! :-(
smtp batch off
smtp timer 500
smtp usemx on
smtp trace 0
smtp maxclients 7
#
#       Mbox parameters
#
attended off
mb attend off
mb tdisc 200
mb conv off
#
#       Ftp parameters
#
ftptdisc 200
ftype bin
#
#       Converse Commands
#
conv host Canberra
```

Minnie is an interesting gateway, in that it runs both a SMTP mail service and a converse server, both of which seem to suffer from memory haemorrhages. This means that minnie is unlikely to stay up for more than 24 hours before running out of memory and hanging. If you are able to, I suggest you offload one or both of these services to a non-gateway machine.

To try and remedy this problem, minnie uses the 'at' command in NOS to exit out of NOS every 6 hours, where it falls back to the NOS batch file, which cleans up and reruns NOS, reclaiming the lost memory. This is mostly sucessful, but under heavy use minnie will still hang.

```
#
#       AT commands
#
at 0001 exit
at 0600 exit
at 1200 exit
at 1800 exit
```

Finally, the gateway's services are started.

```
#
#       Start Servers
#
# The telnet server is activated so that people coming in from other 44. boxes
# can leave mail, and eventually see what other calls to connect to.
start echo
start discard
start ftp
start smtp
start finger
start telnet
start convers
```

Miscellaneous stuff. Logging of incoming connections and server use is appended to the `nos.log` file, and the function keys F7 and F8 have the strings ‘mem stat’, and ‘t s’ attached, so minnie’s operator can quickly review the memory status and the TCP connections to the gateway.

```
#
#       Misc
#
#  Log server use.
#  Set up some function keys.
log nos.log
fkey 65 "mem stat"
fkey 66 "t s"
strace 0
```