NTIA Report 99-360

Performance Assessment of Internet E-mail over Degraded High-Frequency Radio Channels

Christopher Redding
Robert McLean

U.S. DEPARTMENT OF COMMERCE
William M. Daley, Secretary

Larry Irving, Assistant Secretary
for Communications and Information

December 1998

PREFACE

This work was performed by the Institute for Telecommunication Sciences (ITS), Boulder, Colorado for the National Communications System's Office of Standards and Technology, Washington, DC under reimbursable order No. DNRO 66008.

ACKNOWLEDGMENTS

The authors wish to thank Mr. Terry Danielson of NRaD for providing ITS with the JNOS software, AX.25 cards and documentation to ITS for use in the tests. Also appreciated is the technical assistance from his staff, in particular Mr. Steve Barnett and Mr. Jose Ramos.

Appreciation is also given to organizations who lent ITS the HF modems that were used for the tests. Mr. Moose Hataway, Air Force Communications Agency (AFCA) provided the RF-5710 modems and Mr. Dan Roesler, Rockwell, lent ITS the MDM-3001 modems.

CONTENTS

FIGURES

TABLES

ABBREVIATIONS

ACK	Acknowledgment
bps	bits per second
dB	Decibel
FEC	Forward Error Correction
FED-STD	Federal Standard
HF	High frequency
IRTT	Initial Round Trip Time
ITS	Institute for Telecommunication Sciences
ITU-R	International Telecommunications Union Radiocommunications Sector
JNOS	Johan's Network Operating System
LZW	Lempel-Ziv-Welch
MHz	Megahertz
MIL-STD	Military Standard
MSS	Maximum Segment Size
MTU	Maximum Transmission Unit
NOS	Network Operating System
NRaD	Naval Command, Control, and Ocean Surveillance Center, RDT & E Division
OSI	Open Systems Interconnect
RTT	Round Trip Time
SDLC	Synchronous Data Link Controller

SMTP	Simple Mail Transport Protocol
SNR	Signal-to-noise ratio
SRTT	Smoothed Round Trip Time
SYN	Synchronize
TCP/IP	Transmission Control Protocol/Internet Protocol

PERFORMANCE ASSESSMENT OF INTERNET E-MAIL OVER DEGRADED HIGH-FREQUENCY RADIO CHANNELS

Christopher Redding and Robert McLean*

The transmission of E-mail over high-frequency (HF) radio channels is experiencing widespread use in the Government and military communities. This ability is made possible through the convergence of Internet and radio communication protocols, primarily by the implementation of transmission control protocol/internet protocol (TCP/IP) for radio channels. One popular implementation is a software package called JNOS, initially developed by the amateur radio community. JNOS has been adapted for use over HF channels and has been demonstrated to perform at acceptable levels. It has not been characterized in a laboratory environment under known channel conditions. This report presents data that characterizes the performance of the HF E-mail system in a controlled environment, using HF channel simulators.

Key words: E-mail, HF, HF E-mail, JNOS, TCP/IP performance.

1. INTRODUCTION

To transfer E-mail messages over radio channels, most users employ a shareware software package initially developed for the amateur radio community by Phil Karn, amateur radio call sign KA9Q. It was originally referred to as the Network Operating System (NOS), but many modifications to the original version now exist. A modified Johan's Network Operating System (JNOS) [1] was the version used for this test. The modification was done by the Naval Command, Control, and Ocean Surveillance Center, RDT & E Division (NRaD) [2] for optimized use in an HF radio environment. The modifications increased the data rate to 4,800 bps and provided additional timing functions for HF equipment. The NRaD JNOS code was used because it reflects the majority of systems in the field today.

Although JNOS implements many additional TCP/IP applications, including ping, arp, ftp, ttylink, and telnet, only the E-mail application was tested. JNOS also supports gateway functions for a radio interface to wireline connections; however this test only covers the performance over an HF link. JNOS operates on a variety of platforms and over an assortment of interfaces such as ethernet, asynchronous serial ports, and numerous packet radio cards. A customized version of the "DRSI PCPA 8530" packet radio card, a specific hardware interface supported by JNOS and subsequently modified by NRaD, was used for this test. It will be referred to as the AX.25 card throughout this paper.

---

*The authors are with the Institute for Telecommunication Sciences, National Telecommunications and Information Administration, U.S. Department of Commerce, 325 Broadway, Boulder, CO 80303.

2. TEST OBJECTIVES

The objective of this test was to evaluate E-mail throughput over degraded HF channels using the JNOS software package and configurations similar to those used in the NRaD implementation (i.e., as TCP/IP settings, AX.25 card, and HF modem). HF propagation conditions were simulated through the use of a digital signal processor (DSP)-based Watterson model HF channel simulator using degraded conditions defined by the International Telecommunications Union Radiocommunications Sector (ITU-R) Recommendation F.520-2 [3]. In the course of testing, optimum settings and anomalies were reported as they affected performance.

3. SYSTEM DESCRIPTION

In terms of the Open Systems Interconnection (OSI) Reference Model (OSI-RM), the HF E-mail system protocol stack is depicted in Figure 1. SMTP and TCP/IP operate on a computer that interfaces with the AX.25 card. This card performs the AX.25 packetization and synchronous data link controller (SDLC) framing. At the physical layer, a military standard (MIL-STD) 188-110A HF modem provides the serial tone waveform for data transmission in the HF environment.

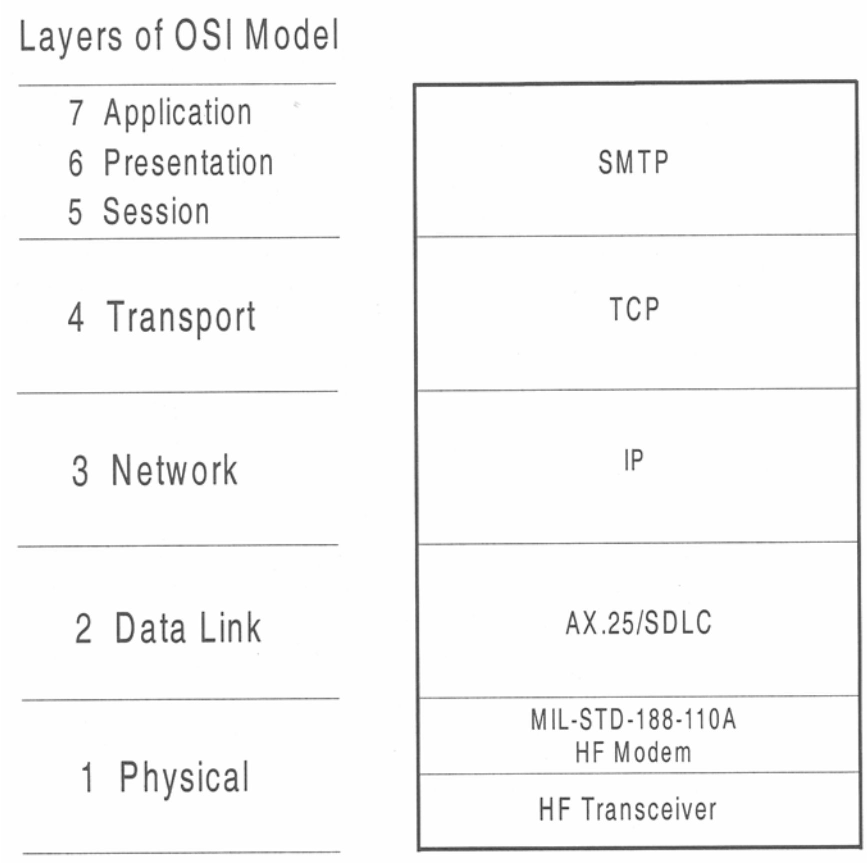

Figure 1. HF E-mail system protocol stack.

3.1 Internet Protocols

The JNOS software package performs the TCP/IP functions and contains interfaces for an application-layer protocol such as SMTP to send and receive messages. An SMTP-compliant mailer, PC Elm, was used in the test.

TCP is a protocol that works at the transport layer and provides reliable data transfer. It is also responsible for flow control through the use of a sliding window (i.e., TCP window), which ensures that transmitted packets do not overflow the receiver's buffer. TCP exchanges packets with the IP layer, a network-layer protocol that provides routing functions. IP datagrams are then exchanged with the AX.25 card, which encapsulates the TCP/IP packet with AX.25 headers, similar to radio call signs, and SDLC framing. The card also provides a synchronous RS-232 output to the modem.

Figure 2 shows the framing structure of the full data encapsulation that constitutes the output of the AX.25 card. The data stream is then sent to the HF serial tone modem.

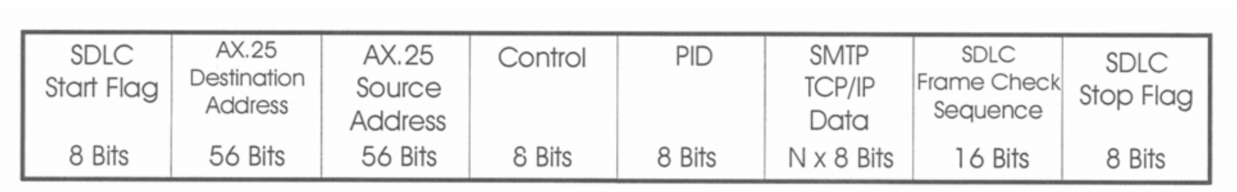

Figure 2. Internet protocol data encapsulation.

The data are first embedded from the SMTP layer into a TCP datagram, which places a 20-byte header on the frame. The TCP datagram is then encapsulated into an IP datagram, which also has a 20-byte header; this produces a total TCP/IP header of 40 bytes. Finally, the TCP/IP datagrams are encapsulated into an AX.25/SDLC frame by the AX.25 board, which adds another 20 bytes of header information. These frames are then sent to the HF modem where forward error correction (FEC) is applied and the data stream is modulated.

The maximum segment size (MSS) is the largest amount of data that can be transmitted at one time. The MSS is preset but negotiated between the sender and receiver at the start of each TCP session. The MSS plus the TCP/IP header make up the maximum transmission unit (MTU); because the TCP/IP header normally consists of 40 bytes, an MSS of 216 bytes produces an MTU of 256 bytes.

JNOS employs built-in data compression using the Lempel-Ziv-Welch (LZW) algorithm, which provides real-time compression of SMTP messages. This compression can be set and negotiated for each SMTP session. The LZW mode can be set to compact or fast[1], with the compact option performing a more efficient compression, and the fast option resulting in a quicker, but "looser," compression. The compact compression was used throughout the tests.

---

[1] J. K. Reinalda and D. E. Thompson, JNOS Commands Manual, ftp://ftp.ucsd.edu/hamradio/packet/tcpip/jnos/docs110.zip (Oct. 20, 1997).

Every time the TCP layer sends a segment, it measures the round trip time (RTT) taken for a single packet to leave one machine, reach the destination, and return an acknowledgment. The RTT plays a critical role in the performance of TCP/IP networks because timeouts and retransmissions are based on the measured RTT, also called the smoothed round trip time (SRTT). In HF networks, where high channel error rates are common, RTTs can vary dramatically and can adversely affect performance. In the absence of a measured RTT, TCP uses a preset initial round trip time (IRTT) for retransmitting unacknowledged packets. Packets will be retransmitted with the IRTT until the SRTT is measured. Each time a packet is unacknowledged, the retransmission timer doubles the SRTT [4]. This continues until the maximum retransmission timeout (maxwait) value is reached or it doubles an integer number of times (specified by 'blimit' in the 'autoexec.net' file), whichever occurs sooner.

3.2 Test Parameters

Table 1 shows the parameters in the HF E-mail that were tested. For all tests, the MSS, MTU and compression were fixed while the window, maxwait, and IRTT were varied. Other variables were the HF modem data rate, E-mail attachment size, and channel simulator conditions with varying signal-to-noise ratios (SNR). Several experimental changes to the MSS, MTU, window, and maxwait also were made to show their effect on the performance.

Table 1. Variable Test Parameters

Parameter	Set of Conditions
Channel Conditions	ITU-R Good[2] @ 0-25 dB SNR in 5-dB steps
	ITU-R Poor[3] @ 0-25 dB SNR in 5-dB steps
Modem Data Rate	300 bps
	1,200 bps
Modem Interleaver Length	0.6 s (short)
Attached Message size	Short = 1,024 byte text file
	Long = 51,200 byte text file
TCP/IP: MSS/MTU/Window	216/256/1,024
	216/256/3,072
TCP/IP maxwait	90 s for 300 bps, 360 s for 1,200 bps

[2] ITU-R Good channel consists of a multipath of 0.5 ms and a doppler spread (fading) of 0.1 Hz.
[3] ITU-R Poor channel consists of a multipath of 2 ms and a doppler spread (fading) of 1 Hz.

3.3 Simulator

The Watterson model narrowband HF channel simulator was used for the tests [3]. It consisted of two DSP-based cards in a computer, each of which simulated an independent path, and each of which was set to identical channel conditions. The simulator was manually set for each channel condition while the input and output signals were observed on an oscilloscope.

3.4 Test setup

The configuration used for this test is shown in Figure 3. The JNOS software ran on a PC-compatible computer, which also interfaced to the AX.25 card. Output from the AX.25 card was fed into the HF modem, which is connected back-to-back with another modem via the simulator. The transmit audio paths passed through the independent paths of the simulator, and into the HF modem receive line. Radios were not used because the simulators operate at baseband.

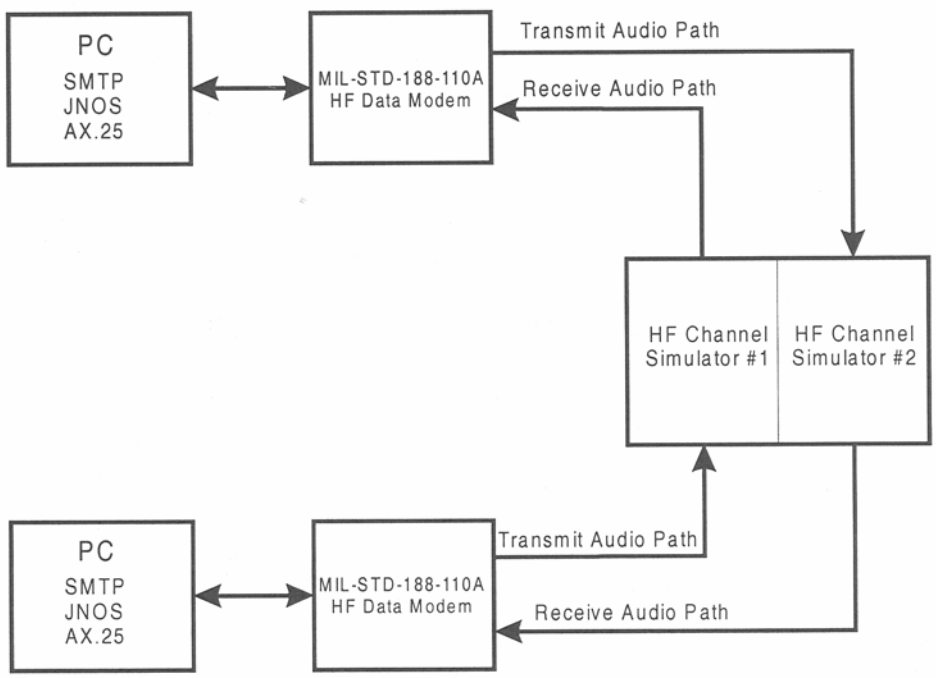

Figure 3. Equipment used for testing.

4. MEASUREMENT STRATEGY AND DATA COLLECTION

Initially, a system throughput was established by programming the simulator for a clear channel condition and transmitting messages from end-to-end. Then a series of E-mail tests was performed;

each test measured the elapsed time to transmit a message attachment under a defined set of conditions. Messages were sent through the system by using the message file attachment option. Each test consisted of configuring the desired conditions that included the simulator conditions, modem settings, TCP/IP parameters, and message attachment size. The simulator was set to the desired channel condition (either ITU-R Good or ITU-R Poor with an SNR from 0 to 25 dB in 5-dB steps). The HF modem rate was set to either 300 or 1,200 bps (always with the short interleaver selected) via the front panel controls. The message attachments consisted of a "short" file of exactly 1,024 bytes and a "long" file of exactly 51,200 bytes. Variable TCP/IP parameters consisted of the window size and the maximum retransmission time (maxwait). The window was set to either 1,024 bytes or 3,072 bytes. An MSS of 216 and MTU of 256 bytes (the default settings) were used throughout. The "maxwait" parameter was also changed depending on the modem data rate that was selected. For 300 bps a maxwait of 360 s was used and for 1,200 bps, 90 s was selected. The TCP/IP parameters were set by editing a configuration file, called 'autoexec.net,' that is used by JNOS when the program is launched. This file allows many TCP/IP and other parameters to be set. A sample of this file is included in Appendix A. During a test each mail message was completely sent before another message was transmitted. A total of ten messages were sent for each test condition; five mail messages would be sent in one direction before reversing directions for the second set of five messages.

4.1 Measured Parameters

When the actual tests were conducted, a log file was produced by the JNOS "trace" command, a function that logs TCP/IP packets to a file. The option to trace all transmitted and received packets at the machine initiating the mail transfer was selected. The log file contains TCP/IP packet information to include time of arrival, sequence number, acknowledgment number, port numbers, TCP flags, and the data size for every packet.

A special data analysis program (included in Appendix B) was written to summarize the significant results of each set of test messages. The node initiating a mail session begins by sending a synchronization (SYN) packet. To end the session a final acknowledgment (ACK) packet is sent by the node that started the session. The data analysis program finds the start and end packets for each mail message with the start and end times, counts any retransmissions of data packets or multiple SYN packets, and calculates a throughput. The TCP estimated round trip time (RTT) was manually logged at the end of each message by recording the values off the computer. Appendix C contains a sample output from the data analysis program.

The throughput formula is the mail message size (1,024 or 51,200 bytes) divided by the total message transmission time. The message size is the actual file size of the attachment and does not include headers attached to the E-mail or any of the extra packets used to establish the TCP connection, the SMTP connection, or to terminate the connections. Because the system compresses the message, it is possible to achieve higher throughput rates than the actual bit rate transmission of the modem. The time to send the message does include all of the overhead, so the start time is from the first SYN packet to the last ACK packet, calculated by the data analysis program.

Retransmissions were detected in the log files by keeping track of the sequence numbers for packets containing data. This includes the actual transmission of the mail or packets used to establish the SMTP connection. There are also packets containing no data used for acknowledgment or establishing the TCP connection. Any retransmissions of non-data packets were not counted. An example of the output of the data analysis program for a single test is shown in Figure 4.

File asbb037a.ada

Port	Start Time	Stop Time	Elap	Packets	Mails	SYNs	Retran	bits/sec
1024	11:10:49	11:17:13	384	15	1	1	0	21
1025	11:19:29	11:26:25	416	16	1	1	0	19
1026	11:27:21	11:37:09	588	16	1	1	0	13
1027	11:35:02	11:43:48	526	17	1	2	0	15
1028	11:49:39	11:58:31	532	18	1	1	1	15

Figure 4. Data analysis program sample output.

4.2 Collisions

When using the TCP/IP protocol for HF E-mail, HF radios are generally half-duplex, that is, they do not transmit and receive simultaneously; while TCP/IP packets may be asynchronously transmitted by either side. This can cause both sides of a link to transmit at the same time with the subsequent loss of some packets. Preliminary testing (in a clear channel) showed that most E-mail was delivered without occurrence of this collision problem. The establishment of an E-mail link and the delivery of the actual mail traffic is basically a sequential process where one or more packets are sent followed by a wait for response. With proper timeout values the responder will have sufficient time for a normal response before the sender tries to repeat a transmission.

However, collisions were occasionally seen under two different sets of circumstances. When collisions occur and packets are lost, mail will still be received without error. TCP/IP automatically retransmits any lost or incorrect packets and ensures that messages are received correctly. As long as collisions do not occur so frequently that transmission cannot occur at all, the problem only affects performance (i.e., decreased throughput).

The first type of collision occurred occasionally at the end of a connection. When the mail message itself is completed, there are several packets exchanged by SMTP which indicates that there is no more traffic to follow in either direction. At this point, another exchange of packets occurs and the SMTP connection is closed. There are also packets that terminate the TCP/IP connection. Sometimes during this shutdown process both computers send packets simultaneously and cause both transmitters to key at the same time, resulting in a collision and extending the completion time for the entire process. During the tests, this type of collision did occur occasionally and the results reflect a somewhat lower throughput because of it.

4.3 Message Compression

The second type of collision observed is related to message compression. The JNOS software automatically compresses mail messages to reduce the size of transmissions using the LZW compression algorithm. Optionally, the compression may be turned off or varied from LZW 9-bit to 16-bit compression. The 9-bit compression is a faster method than 16-bit, but the 16-bit compression results in the smallest messages. The JNOS Command Manual states that 16-bit compression is turned on by default; however, the default is actually 9-bit compression in the version of software that was used. With compression working sufficiently fast, no collisions occur. But when the compression cannot keep up with the transmission of packets, the sending computer will send a smaller than normal packet and then not wait for the acknowledgment from the receiving computer. This results in collisions. During the preliminary testing, Intel 486-33 MHz computers were used. No collisions occurred with these machines while using 9-bit compression, but collisions regularly occurred using 16-bit compression for 50 kbyte mail messages. This has implications for the users in the field. The type of compression chosen should match the processing power of the computers used, otherwise, any improvement in transmission efficiency caused by compression can be lost due to increased collisions. The actual compression method used is negotiated automatically by the two computers during the SMTP connection establishment phase of a mail transfer. The method chosen is the lowest common denominator of the compression methods set on the two computers. Because we desired to utilize the maximum compression possible, the computers were upgraded to Intel Pentium Pro 200 MHz machines for the actual testing. No collisions due to the use of 16-bit compression were noted during testing.

5. PRESENTATION OF MEASURED DATA

The data collected are presented in four graphs. Figures 5 and 6 show system throughput for ITU-R Good channel conditions with 1,024 and 51,200 byte files, and Figures 7 and 8 show ITU-R Poor channel conditions with 1 K and 50 K files. Table 2 shows the system settings for the four plots.

Table 2. System Settings

Parameter	Value
MSS/MTU	216/256 bytes
LZW bits	16
IRTT	60s
MRTT	90s for 1,200 bps / 360s for 300 bps
Short file	1,024 byte text file (1 K)
Long file	51,200 byte text file (50 K)
HF modem	300 or 1,200 bps, 0.6 s interleaver

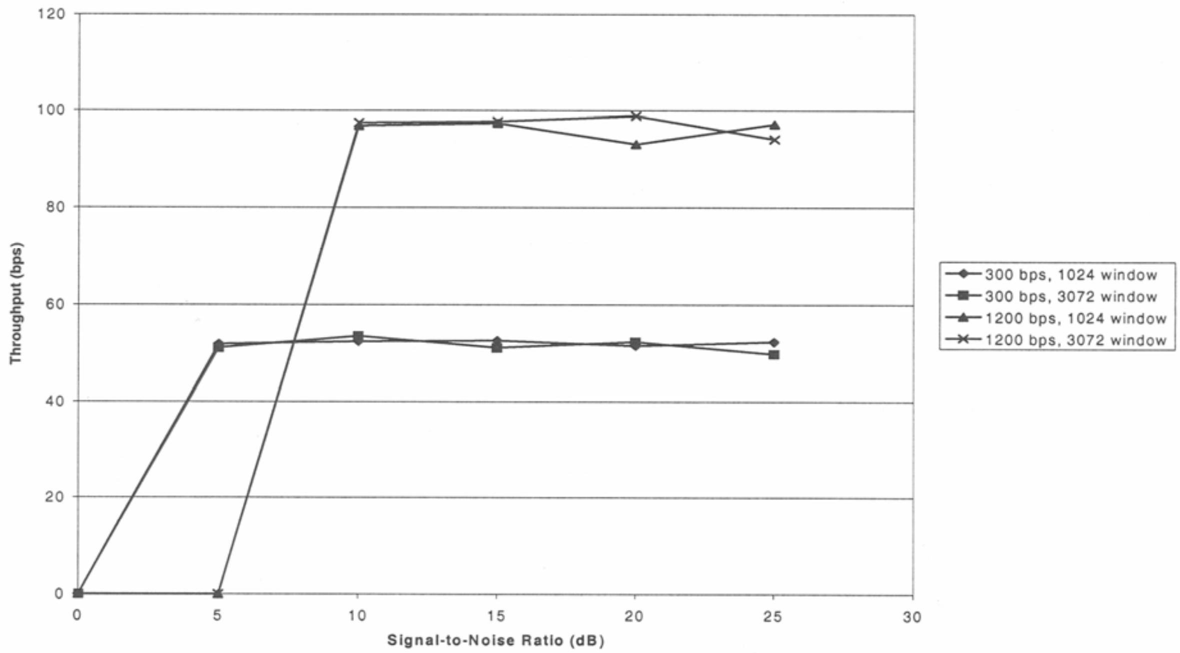

Figure 5. Throughput versus ITU-R Good channel with short file attachment.

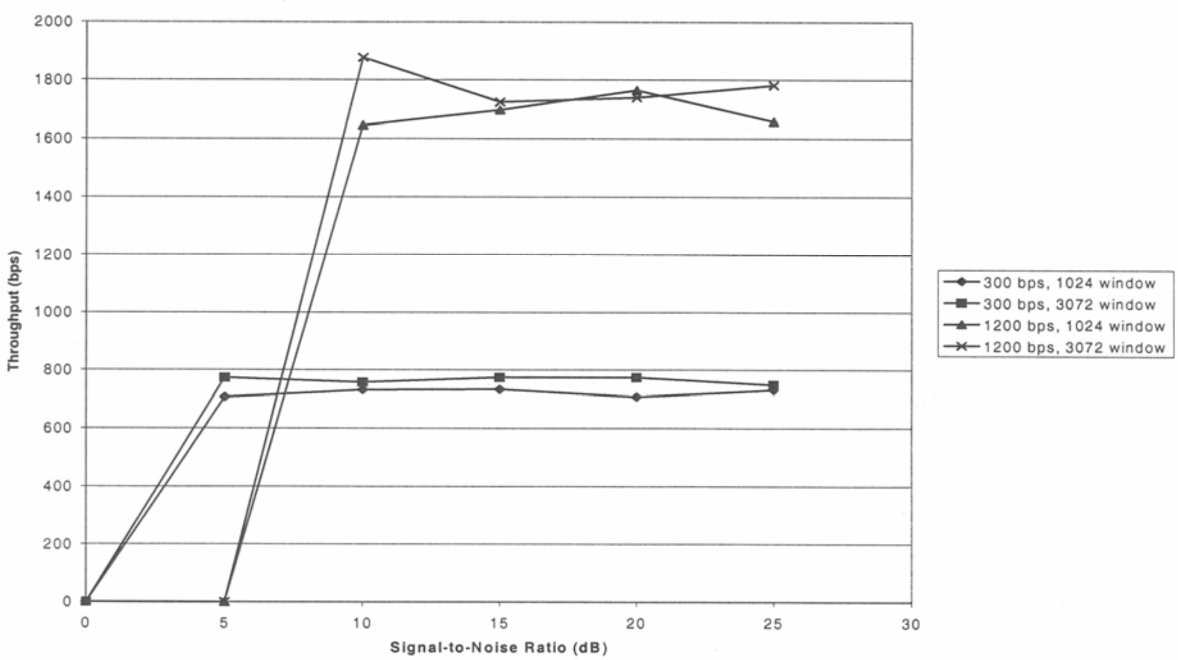

Figure 6. Throughput versus ITU-R Good channel with long file attachment.

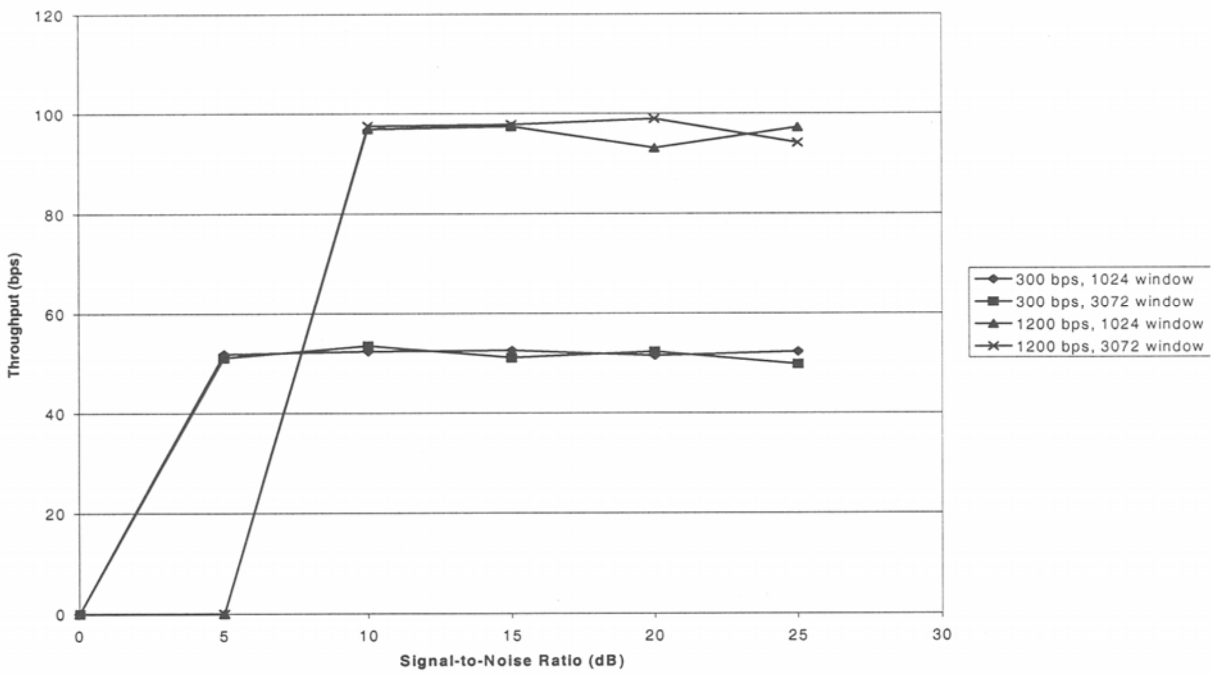

Figure 7. Throughput versus ITU-R Poor channel with short file attachment.

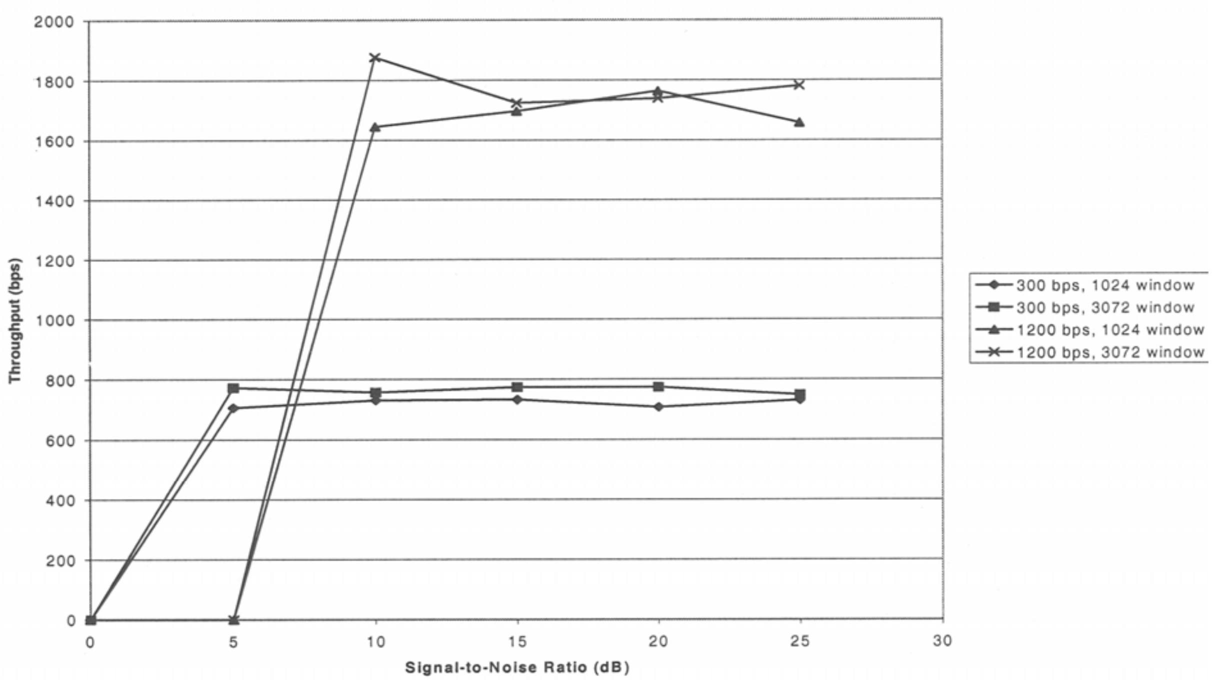

Figure 8. Throughput versus ITU-R Poor channel with long file attachment.

6. AUXILIARY TESTING

In addition to the regular test schedule, we conducted several extra tests in an effort to improve the performance of JNOS.

6.1 75 bps

Testing performed at 300 bps and 1,200 bps modem data rates was unable to deliver E-mail at the 0-dB SNR setting of the simulator. (At 1,200 bps no E-mail could be transferred at 5-dB SNR.) Under these severe conditions would the system be able to pass E-mail messages when the modems were set for 75 bps transmission?

The simulator was configured for ITU-R Good condition with 0-dB SNR, the modem was set for 75 bps with the short interleaver, and the TCP/IP settings were set the same as the 300 bps tests with the 1,024-byte window. Tests were conducted with both short and long E-mail files. The results of these tests were that mail was delivered with little trouble. When the short file was sent, the average throughput was 17.4 bps and there was one retransmitted SYN packet and one retransmitted data packet in the entire 10 mail messages. With the long file, the average throughput was 209.0 bps with a total of 6 retransmitted data packets.

6.2 2,400 bps

How would the system perform at 2,400 bps? Several tests were conducted at this higher speed to measure the effect. The channel simulator was configured for ITU-R Good channel condition with 15-dB and 20-dB SNRs. TCP/IP settings were the same as at 1,200 bps and the 1,024-byte window was used. Table 3 shows the throughput results. For ease of comparison, the table repeats the throughput results that were taken at 1,200 bps (see Figures 5 and 6 to reference the data). The data is graphically presented in Figure 9 to show performance differences.

Table 3. Average Throughput at 1,200 and 2,400 bps

Message Size	Throughput (bps) at 15-dB SNR		Throughput (bps) at 20-dB SNR	
	Modem Data Rate = 1,200 bps	Modem Data Rate = 2,400 bps	Modem Data Rate = 1,200 bps	Modem Data Rate = 2,400 bps
Short (1 K)	97.2	57.3	99.0	114.7
Long (50 K)	1,624.2	540.5	1,714.2	1,973.7

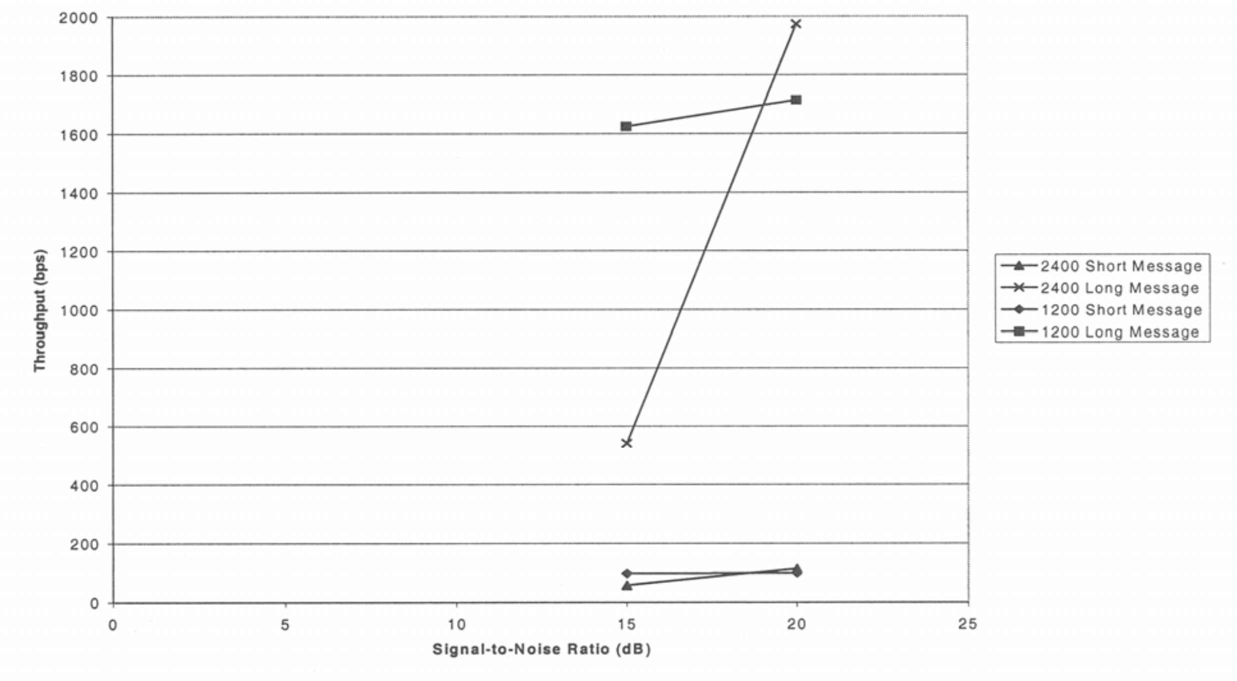

Figure 9. Throughput comparison for 1,200 and 2,400 bps with short and long messages.

Note that with an SNR of 20-dB, the overall throughput was better at 2,400 bps. There were no retransmissions during the short message test and 16 retransmissions with the long message under these conditions. However, when the channel was degraded to 15-dB SNR the results are significantly different; the throughput was much better at 1,200 bps than at 2,400 bps. This is due to the drastically increased number of retransmissions at 2,400 bps (23 retransmissions for the short message and 165 for the long message).

6.3 Optimum Settings

In discussions with engineers at the Defense Information Systems Agency (DISA)[4], we learned that JNOS performance depends heavily on the MTU, MSS and TCP window size. They have experienced good results using an MSS of 512, an MTU of 576, and a window size of 2,048 bytes. Similar testing performed by Rockwell[5] also emphasizes the importance of the same three parameters and suggests using an MSS of 472, an MTU of 512, and a window of four times the MSS value in the set up ifconfig section and two times the MSS value in the 'Set up TCP/IP defaults' section of the autoexec.net file. We wanted to know if these settings would give good results with our configuration.

---

[4] Private communications with DISA.
[5] Private communications with Rockwell.

To test these settings, parameters close to the above two sets of settings were selected, but somewhat different from each. An MSS of 472 bytes, an MTU of 512 bytes (the MSS plus 40 bytes for the TCP/IP header), and a window of 2,048 bytes (four times the MTU size which allows four maximum size packets in a window) was used. The simulator was set for ITU-R Good condition at 15 dB SNR, and tests were conducted with short and long messages at both 300-bps and 1,200-bps data rates. The results are shown in Table 4.

Table 4. Average Throughput with MSS = 472, MTU = 512 and window = 2,048

Message Size	Throughput at 300 bps	Throughput at 1,200 bps
Short (1 K)	55.5	106.5
Long (50 K)	887.0	2,178.7

For comparison, the previous results for the same HF channel conditions are shown in Table 5.

Table 5. Average Throughput with MSS = 216 and MTU = 256

Message Size	Window = 1,024		Window = 3,072	
	Throughput (bps) Modem Data Rate = 300 bps	Throughput (bps) Modem Data Rate = 1,200 bps	Throughput (bps) Modem Data Rate = 300 bps	Throughput (bps) Modem Data Rate = 1,200 bps
Short (1 K)	51.8	97.2	51.8	97.5
Long (50 K)	738.9	1,624.2	767.1	1,733.6

The recommended settings (MSS = 472, MTU = 512, and window = 2,048) clearly outperformed the earlier results. In fact these were the highest throughput results of all of the tests under any conditions. We agree that these are good settings to use at these channel conditions. Other channel conditions may vary the performance.

7. CONCLUSION

Plots of throughput versus channel conditions are presented in Figures 5-8. All the plots exhibit the same type of pattern; notably throughput increases from a minimum of 0 bps to a plateau.

Depending on file size and modem data rate, the plateaus occurred at different maximum data rates and SNR's. Minor differences in throughput occurred as the window varied from 4 times the MTU to 12 times the MTU for a particular modem data rate, indicating that a larger window has no significant effect on performance. In the cases where there is a difference in throughput, the larger window almost always produced better results. Contributing to the variances was the small number of trials performed.

As the SNR decreased, a lower data rate was required to pass traffic, as shown by the 75 bps test performed at 0 dB. This was expected since there is increased error correction coding applied at lower data rates. Conversely, higher modem data rates could be used as the channel SNR increased. This effect occurs with the three data rates that were used. The difference in maximum throughput between the 1,200 and 2,400 bps data rates was not that significant, although optimization of the data rate should be made if the channel supports higher rates. Figure 9, in conjunction with Figures 5 - 8, shows the effect that data rate has on performance for 300 and 1,200 bps data rates. It is expected that the same pattern would be observed for other data rates. In particular, throughput increases from a minimum of 0 bps to a data-rate-dependent plateau. The knee of the curve moves right, or occurs at a higher SNR, as the data rate increases. Optimization of the data rate versus channel SNR could be accomplished with an adaptive data rate modem.

Users should choose the compression method with regard to the processing power of their computer, since compression is performed in real time. The actual compression level used will be a result of a negotiation between the two stations involved in the transmission. However, users should be aware that the JNOS documentation erroneously states that the compact compression is the default setting. To be sure that the compact compression is being used, autoexec.net should contain the statement "lzw bits = 16."

Other than the effect of compression, the tests show the best performance occurred when a large file attachment was sent. There are two reasons for this effect: 1) because the overhead is relatively constant, the ratio of message size to total transmission increases significantly; 2) with longer messages the TCP window can be used more efficiently, which allows transmission up to the maximum window size before an acknowledgment is expected. When the 1,024 byte message was transmitted, only single packets were sent in a single transmission. The radio link would then have to be turned around for the response packet. When the 51,200 byte message was transmitted, the actual message part of the transmission contained several packets (up to the window size) sent together without changing the direction of the radio link. Every time the link turns around, keying the HF modem adds a synchronization overhead of 0.6 s and an average of 0.3 s of filtering at the end of the transmission (assuming the short interleaver).

Comparing the differences in throughput between the long files and the short files shows a 15 to 1 increase in throughput. Although this was not tested, mailers that support SMTP service extensions for command pipelining theoretically would increase throughput since short packets are combined and sent, reducing the number of link turnarounds.

8. REFERENCES

[1] I. Wade, *NOSintro – TCP/IP Over Packet Radio; An Introduction to the KA9Q Network Operating System*, United Kingdom, Dowermain Ltd., 1992.

[2] Naval Command, Control, and Ocean Surveillance Center, RDT & E Division (NRaD), General Battle Force Electronic Mail System Manual, Version 1.0, 1997.

[3] ITU-R Recommendation for Fixed Service, Use of High Frequency Ionospheric Channel Simulators, Recommendation F.520-2, March 1992.

[4] D. E. Comer, *Intemetworking with TCP/IP*, Volume I, Englewood Cliffs, NJ: Prentice Hall, 1991.

APPENDIX A

Listing of autoexec.net file

```
#autoexec.net       "adam"

# Miscellaneous setup ***************************************
isat             yes            # 286/386 clock
log 1            # dump/session.log
watch            off
memory thresh    14336
memory ibufsize 3000

# Set up domain defaults ************************************
domain cache size 50
domain suffix
domain retries 2
domain cache clean yes

# Station Identification ************************************
ip address       1.1.1.1
hostname         adam
# This MUST precede 'attach'
# callsign used in ax.25 header used much like ethernet address in
ethernet datagram
ax25 mycall  adam-ax

# Attach Sync interface *************************************
attach drsi 0x310 7 ax25 dr0 2048 256 1200 1200
param dr0 txdelay 45
param dr0 enddelay 45
param dr0 txtail 25
param dr0 slottime 180
param dr0 persist 176
param dr0 turn 20  #use 70 for USQ-122 modem/20 for RF-5710 and
MDM-3001 modems

# Set up ifconfig *******************************************
ifconfig dr0 tcp timertype linear
ifconfig dr0 tcp retries 48
ifconfig dr0 tcp blimit 20
#dr0 maxwait in milliseconds, make all nodes in net a few tens of
seconds different (600000-1000000)
ifconfig dr0 tcp maxwait 360500
#dr0 irtt in milliseconds, make all nodes in net a few seconds
different (10000-17000)
ifconfig dr0 tcp irtt 60500
ifconfig dr0 tcp mss 216
```

APPENDIX A

Listing of autoexec.net file (continued)

```
ifconfig dr0 tcp window 1024

# Set up TCP/IP defaults ***************************************
ip rtimer           60
ip ttl              255

tcp mss             216
tcp irtt            60500
tcp window          1024

lzw bits            16

# Start network services ***************************************
start               pop3
start               smtp

# Set up IP routing ********************************************
## this routing section should be edited to reflect static routing
## for this host
# route out to  network interface gateway router

route add default dr0 bob

# Set up the ARP table *****************************************
arp add 1.1.1.2 ax25 bob-ax dr0

# Set up the mailbox *******************************************
smtp                timer  600  #in seconds, longer the better
smtp mode           route
smtp trace 3       #use 1 for no tracing, use 3 to see smtp client
conversation
smtp batch on
#time interval used to notify user/return undelivered mail
smtp dtimeout 10  #return mail after dtimeout hours as undelivered

## default server to send mail if not addressed for local users
(popusers)
## or hosts in domain.txt file
smtp gateway bob
smtp usemx on

# >>>>>>> THIS IS THE LAST COMMAND IN THE FILE <<<<<<<<
domain dns on
# THE END
```

APPENDIX B

Data Analysis Program

```
BEGIN { MessageCnt = 0 }

NR == 1 {
  print
  print "File " FILENAME
  print ""
  print "Port  Start Time Stop Time  Elap Packets Mails SYNs Retran
bits/sec"
}
/ sent:/ {
  time = $4
  day = $3
  getline
  getline
  len = $3
  getline
  port = substr($2, 1, index($2, "-") - 1)
  seq = hexnum($4)
  # see if the port number already exists
  i = 1
  while (i <= MessageCnt) {
    if (port == Ports[i]) {
      StopTime[i] = time
      StopDay[i] = day
      Packets[i]++
      break
    }
    i++
  }
  if (i > MessageCnt) {  # port not found so add it
    Ports[i] = port
    StartTime[i] = time
    StartDay[i] = day
    Packets[i] = 1
    SYNs[i] = 0
    Retrans[i] = 0
    MailsInCon[i] = 1
    State = 0
    MessageCnt++
  }
```

```
  if ($0 ~ " SYN ") {
    SYNs[i]++
    MSS = $NF
    CurrentSeq = seq + 1
  }
  if ($0 ~ " Data ") {
       # Look for multiple messages in a connection and
retransmissions
    if (seq < CurrentSeq)
      Retrans[i]++
    else {
      CurrentSeq += $NF
      # Look for the packet that separates messages in a single
connection
      # Note that the method depends on the message length being
greater than
      #   the MSS.
      if (($NF == MSS) && (State == 0))
        State = 1
      if (($0 ~ " PSH ") && (State == 1))
        State = 2
      if ((State = 2) && ($NF >= 45) && ($NF <= 48)) {
        # This is a packet between mail messages
        MailsInCon[i]++
        State = 0
      }
    }
  }
}

/ recv:/ {
  time = $4
  day = $3
  getline
  getline
  getline
  port = substr($2, index($2, ">") + 1, length($2))
  # see if the port number already exists
  i = 1
  while (i <= MessageCnt) {
    if (port == Ports[i]) {
      StopTime[i] = time
```

Data Analysis Program (continued)

```
      StopDay[i] = day
      break
    }
    i++
  }
}

END {
  for (i = 1; i <= MessageCnt; i++) {
    split(StartTime[i], StartAr, ":")
    split(StopTime[i], StopAr, ":")
    StopT = (StopAr[1] * 60 + StopAr[2]) * 60 + StopAr[3]
    StartT = (StartAr[1] * 60 + StartAr[2]) * 60 + StartAr[3]
    ElapTime = StopT - StartT
    if (StartDay[i] != StopDay[i])
      ElapTime += 86400
    if (ElapTime != 0)
      Thruput = size * 8 * MailsInCon[i] / ElapTime
    else
      Thruput = 1000
    printf("%4d  %s   %s   %4d%6d%6d%6d%6d     %.0d\n", Ports[i],
StartTime[i],
       StopTime[i], ElapTime, Packets[i], MailsInCon[i], SYNs[i],
Retrans[i],
      Thruput)
  }
  print
}

function hexnum(hnum) {   # convert hexadecimal number hnum to
decimal
# assumes the number begins with an x
  num = 0
  len = length(hnum)
  hnum = substr(hnum, 2, --len)
  while (len >= 1) {
    char = substr(hnum, 1, 1)
    if (char == "a")
      value = 10
    else if (char == "b")
      value = 11
    else if (char == "c")
```

APPENDIX B

Data Analysis Program (continued)

```
      value = 12
    else if (char == "d")
      value = 13
    else if (char == "e")
      value = 14
    else if (char == "f")
      value = 15
    else
      value = char

    num = 16 * num + value
    hnum = substr(hnum, 2, --len)
  }
  return num
}
```

APPENDIX C

Sample Output From Data Analysis Program

File jlac026a.ada

Port	Start Time	Stop Time	Elap	Packets	Mails	SYNs	Retran	bits/sec
1024	15:48:34	15:51:58	204	51	1	1	0	2007
1025	16:14:41	16:18:06	205	52	1	1	0	1998
1026	16:19:08	16:22:32	204	51	1	1	0	2007
1027	16:25:22	16:28:46	204	51	1	1	0	2007
1028	16:31:27	16:34:58	211	52	1	1	0	1941

File jlac026a.bob

Port	Start Time	Stop Time	Elap	Packets	Mails	SYNs	Retran	bits/sec
1024	15:18:17	15:21:42	205	51	1	1	0	1998
1025	15:23:50	15:28:58	308	54	1	1	1	1329
1026	15:28:29	15:33:55	326	55	1	1	1	1256
1027	15:33:19	15:38:54	335	55	1	1	2	1222
1028	15:39:33	15:44:11	278	53	1	1	1	1473

FORM NTIA-29
(4-80)

U.S. DEPARTMENT OF COMMERCE
NAT'L. TELECOMMUNICATIONS AND INFORMATION ADMINISTRATION

BIBLIOGRAPHIC DATA SHEET

	1. PUBLICATION NO. 99-360	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE
Performance Assessment of Internet E-mail over Degraded High-frequency Radio Channels

5. Publication Date

6. Performing Organization Code

7. AUTHOR(S)
C. Redding - R. McLean

8. PERFORMING ORGANIZATION NAME AND ADDRESS
National Telecommunications and Information Admin.
Institute for Telecommunication Sciences
325 Broadway
Boulder, CO 80303

9. Project/Task/Work Unit No.

10. Contract/Grant No.

11. Sponsoring Organization Name and Address
National Communications System
Office of the Manager - N/6
701 South Court House Road
Arlington, VA 22204-2198

12. Type of Report and Period Covered

13.

14. SUPPLEMENTARY NOTES

15. ABSTRACT (*A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.*)

The transmission of E-mail over high-frequency (HF) radio channels is experiencing widespread use in the Government and military communities. This ability is made possible trough the convergence of Internet and radio communication protocols, primarily by the implementation of transmission control protocol/internet protocol (TCP/IP) for radio channels. One popular implementation is a software package called JNOS, initially developed by the amateur radio community. JNOS has been adapted for use over HF channels and has been demonstrated to perform at acceptable levels. It has not been characterized in a laboratory environment under known channel conditions. This report presents data that characterizes the performance of the HF E-mail system in a controlled environment, using HF channel simulators.

16. Key Words (*Alphabetical order, separated by semicolons*)
E-mail, HF, HF E-mail, JNOS, TCP/IP performance

17. AVAILABILITY STATEMENT	18. Security Class. (*This report*)	20. Number of pages
☒ UNLIMITED.	Unclassified	32
☐ FOR OFFICIAL DISTRIBUTION.	19. Security Class. (*This page*) Unclassified	21. Price:

*U.S. GOVERNMENT PRINTING OFFICE 1999-0-773-022/40126