Using Udpcast to IP Multicast Data over Amateur Packet Radio Networks

Paul D. Wiedemeier, Ph.D., KE5LKY
Clarke M. Williams, Jr. Endowed Professor of Computer Science
The University of Louisiana at Monroe
Computer Science and Computer Information Systems Department
College of Business Administration
700 University Avenue, Monroe, Louisiana 71209
318-342-1856 (Work) or 318-396-1101 (Fax)
wiedemeier@ulm.edu or KE5LKY@arrl.net

Abstract

Traditional data transmission software, including Winlink 2000, AirMail, FTP, and SCP, use point-to-point unicast to transmit data between two computers attached to the same network. Unfortunately significant time is required to transmit data from one computer to multiple computers when using unicast. During disasters and crises, minimizing the time to disseminate data is vital. Thus, to transmit data efficiently, point-to-multipoint multicast data transmission software should be used. In this paper we discuss how to multicast data over amateur packet radio networks using the Udpcast file transfer tool. Udpcast was written for both the UNIX/Linux and Microsoft Windows operating systems, and is widely used within the UNIX/Linux community to transmit operating system images from one computer to multiple computers attached to the same network. Additionally, any computer can use the Udpcast udp-sender and udp-receiver commands to transmit data or receive data. For these two main reasons, we advocate using Udpcast to multicast data over amateur packet radio networks during emergencies.

Key Words

Udpcast, File Transfer, IP Multicast, Data Transmission, Amateur Packet Radio Network

Introduction

During disasters and crises, the ability to transmit data to multiple destinations is paramount and amateur packet radio networks are often used to provide emergency data communication services when these situations occur. Most data transmission software, including Winlink 2000, AirMail, FTP, and SCP unicast data. Thus, using these, many point-to-point data transmissions are needed to transmit data to multiple computers. Conversely, multicast software can transmit data to multiple destinations using a single point-to-multipoint data transmission. Unfortunately, available multicast software designed specifically for data transmission over amateur packet radio networks (1) require a dedicated computer to act as a multicast server, (2) are written only for the Microsoft Windows operating system, and (3) are not all actively supported.

In this paper, we discuss how to IP multicast data over amateur packet radio networks using the Udpcast file transfer tool (e.g. data transmission software). Compared to other multicast software, we have identified three advantages to using the Udpcast file transfer tool. First, Udpcast was written for both the UNIX/Linux and Microsoft Windows operating systems, so you have the freedom to choose between operating systems. Second, Udpcast multicasts data without using a dedicated computer. This advantage will be highlighted later in this paper. Last, Udpcast is actively supported and widely used

within the UNIX/Linux community. For these three reasons, we advocate using Udpcast when data must be transmitted simultaneously to multiple computers during emergencies.

The structure of this paper is as follows. The *IP Addressing and Data Transmissions* section discusses Internet Protocol (IP) (ISI, 1981) addressing, IP unicast, IP broadcast, and IP multicast. The *Multicast Software for Amateur Packet Radio Networks* section reviews the RadioMirror and AltCast multicast software. The *Udpcast File Transfer Tool* section presents the basic structure of the udp-sender and udp-receiver commands. The *Using Udpcast to Multicast Data* section presents examples of how to use the udp-sender and udp-receiver commands to IP multicast data. The *Conclusions* section compares and contrasts the RadioMirror, AltCast, and Udpcast multicast software and reiterates our arguments as to why we believe Udpcast should be used to IP multicast data over amateur packet radio networks during emergencies.

IP Addressing and Data Transmissions

Regardless of whether data is transmitted using unicast or multicast, all computers attached to IP based networks must be assigned an IP address of the form A.B.C.D, where A, B, C, and D are 8 bit numbers in the range 0 through 255. For a thorough discussion of IP addressing the reader should refer to (Comer, 2000) and (Tanenbaum, 2003).

Once a computer receives an IP address, data can be transmitted using either IP unicast, IP broadcast, or IP multicast (Comer, 2007). Each data transmission method is discussed in the following paragraphs, but for a thorough discussion of these data transmission methods the reader should refer to (Comer, 2000) and (Tanenbaum, 2003). For the remainder of this paper, we will refer to the terms IP address, IP packet, IP unicast, IP broadcast, and IP multcast as address, packet, unicast, broadcast, and multicast respectively.

Unicast

Unicast, often referred to as point-to-point data communication, transmits packets from a source computer to a destination computer. Each packet includes the addresses of the source and destination computers. Many of the four traditional groups of Internet applications (e.g. tools) (Comer, 2007), including electronic mail, news, remote login, and file transfer, use unicast to transmit data.

Broadcast

If unicast is referred to as point-to-point data communication, then broadcast is point-to-multipoint data communication, in that all packets transmitted by a source computer are delivered to all computers attached to the same network (Mogul, 1984-A) (Mogul, 1984-B). The address resolution protocol (Plummer, 1982), the dynamic host configuration protocol (Drom, 1997), and routing table updates are examples of Internet applications that are known to use broadcast.

Actually, two forms of broadcast exist: limited and direct. The address for limited broadcast is 255.255.255.255, and packets transmitted using limited broadcast are only delivered to those computers attached to the same network as the transmitting computer.

Direct broadcast addresses are of the form A.B.C.255, A.B.255.255, or A.255.255.255. The specific broadcast address used by a computer depends upon the underlying computer network to which it is attached. Packets transmitted using direct broadcast are usually only delivered to those computer

attached to the same network as the transmitting computer, but routers may be configured to transmit direct broadcast packets across other networks. The UNIX ifconfig command can be used to obtain the broadcast address for a specific UNIX computer's network interface. Figure A1 in the Appendix shows the output generated by the UNIX ifconfig command, which queries the computer dcrl1's AX.25 ax0 network interface. The computer dcrl1 is discussed later in the *Using Udpcast to Multicast Data* section.

Multicast

Multicast is similar to broadcast in that packets are transmitted to multiple computers attached to the same network as the transmitting computer. However, multicast packets are only accepted by a subset of all attached computers. That is, only those computers that agree to accept packets from a specific multicast address do so. Multicast addresses are in the range 224.0.0.0 through 239.255.255.255 and those Internet applications that use multicast, such as the University of California, Berkeley's Streaming Media Toolkit and VideoCharger by International Business Machines (IBM) Corporation, often transmit audio and/or video data.

Multicast Software for Amateur Packet Radio Networks

As of July 2008, two software applications have been identified that were specifically developed to multicast data over amateur packet radio networks. They are RadioMirror and AltCast, and each are discussed in the following paragraphs.

RadioMirror, written by John Hansen, Ph.D., W2FS, for the Microsoft Windows 95 operating system, was introduced to the amateur packet radio community during the mid 1990's. Dr. Hansen wrote RadioMirror to provide "a general purpose mechanism for moving data from a central server to a lot of client computers, all at the same time. … In emergencies, a RadioMirror server would provide a mechanism to distribute large quantities of information (including graphics and other multimedia files) over a wide region" (Hansen, 1997-A). Regarding functionality, a dedicated RadioMirror server continuously multicasts data over a very high frequency (VHF) or ultra high frequency (UHF) amateur packet radio network for some duration of time, usually 24 hours, to guarantee that all RadioMirror clients have an opportunity to receive the data.

While the RadioMirror software is still available for download (see Table A1 in the Appendix), little web-based documentation exists. The best RadioMirror documentation found were those files included with the actual software distribution. Specifically, the files RADIOMIR.TXT (Hansen, 1997-A), RMTECH.txt (Hansen, 1997-B), RMSERVER.TXT (Hansen, 1997-C), and RMCLIENT.TXT (Hansen, 1997-D) were read. Unfortunately, it appears that Dr. Hansen is not actively supporting the RadioMirror software as versions for the Microsoft Windows XP and Vista operating systems do not exist.

AltCast, written by Walt Fair, Jr., W5ALT, for the Microsoft Windows XP operating system, was introduced to the amateur packet radio community during the early 2000's. Similar to RadioMirror, a dedicated AltCast server continuously multicasts data over a high frequency (HF) amateur packet radio network, again for a given duration of time, using one of several phase-shift keying (PSK) modes: PSK31, QPSK31 or PSK63 (Brabham, 2005). Any Altcast client can receive the data multicasted by the dedicated AltCast server. The AltCast software can be downloaded from the website shown in Table A1 in the Appendix.

The Udpcast File Transfer Tool

The Udpcast file transfer tool was written by Alain Knaff for the UNIX/Linux and Microsoft Windows XP operating systems, and released in the early 2000's under the GNU General Public License 2.0. Widely used within the UNIX/Linux community, Udpcast's primary strength lies in its ability to multicast operating system images from one computer to multiple computers attached to the same network. Functionally, Udpcast is primarily used to multicast data from one computer to multiple computers, but it is capable of unicast data transmissions.

Installation of the Udpcast software on a UNIX/Linux computer is accomplished by (1) using UNIX/Linux package manager software or (2) downloading the software (See Table A1 in the Appendix) and performing a manual install. Once installed, the Udpcast file transfer tool is comprised of two command line programs; udp-sender and udp-receiver. A complete list of Udpcast command line arguments can be obtained from (Knaff 2005). Below, the command line arguments used in this research are discussed.

Udp-sender Command Structure

The structure of the udp-sender command is shown in Figure 1. The `--file` argument specifies which file to multicast to the udp-receivers. If this argument is absent, input is read from standard input, which permits UNIX piping and redirection. The `--interface` argument is used to specify which of the computer's network interfaces will be used to transmit data. If this argument is absent, the default network interface used is eth0. The `-b` argument (note, that the argument is `-b`, not `--b`) specifies the packet size to use when the sender transmits data. If this argument is absent, the default packet size used is 1456.

```
udp-sender --file FILETOSEND --interface INTERFACE –b BLOCKSIZE
--async –-fec IxR/S --max-bitrate BPS –-log LOGFILE
--bw-period SECONDS
```

Figure 1: The udpcast-sender command.

The `--async`, `--fec`, and `--max-bitrate` arguments are used when the sender does not expect request confirmations from the receiver(s). This is referred to as unidirectional mode and is beneficial when a high latency and/or low bandwidth (e.g. amateur packet radio networks) channel is used. Specifically, the `--async` argument defines asynchronous mode and is used without confirmation. The `--fec` argument adds forward error correction (FEC) to the transmitted data to provide reliability when the data communication channel is noisy. The amount of FEC added is based on the factors interleave (I), redundancy (R), and stripsize (S). For detailed information about the `--fec` argument the reader should refer to (Knaff, 2006). The argument `--max-bitrate` is used to limit how fast the sender transmit data to the receiver(s). It essentially prevents the sender from "drowning" one or more receivers with data.

The `--log` and `--bw-period` arguments permit the udp-sender to log instantaneous bandwidth data to a file. The `--log` argument specifies the name of the file in which to write the data. The `--bw-period` argument defines the number of seconds between which instantaneous bandwidth is written to the log file. That is, "every so much seconds, log instantaneous bandwidth seen during that

period" (Knaff, 2005). While Udpcast only generates instantaneous bandwidth, file transmission time can be computed by dividing the size of the file transmitted by the average instantaneous bandwidth written to the log file (Wiedemeier, 2007).

Udp-receiver Command Structure

The structure of the udp-receiver command is shown in Figure 2. Similar to udp-sender, the `--file` argument specifies the name of the file used to store the data received from the udp-sender. If this argument is absent, input is written to standard output. Again, this permits UNIX piping and redirection. The `--nosync` argument must be used when output is written to a file or a pipe. The `--interface` argument is used to specify which of the computer's network interfaces will be used to transmit data. As with the udp-sender command, if this argument is absent, the default network interface used is eth0.

```
udp-receiver --file FILETORECEIVE --nosync --interface INTERFACE
```

Figure 2: The udpcast-receiver command.

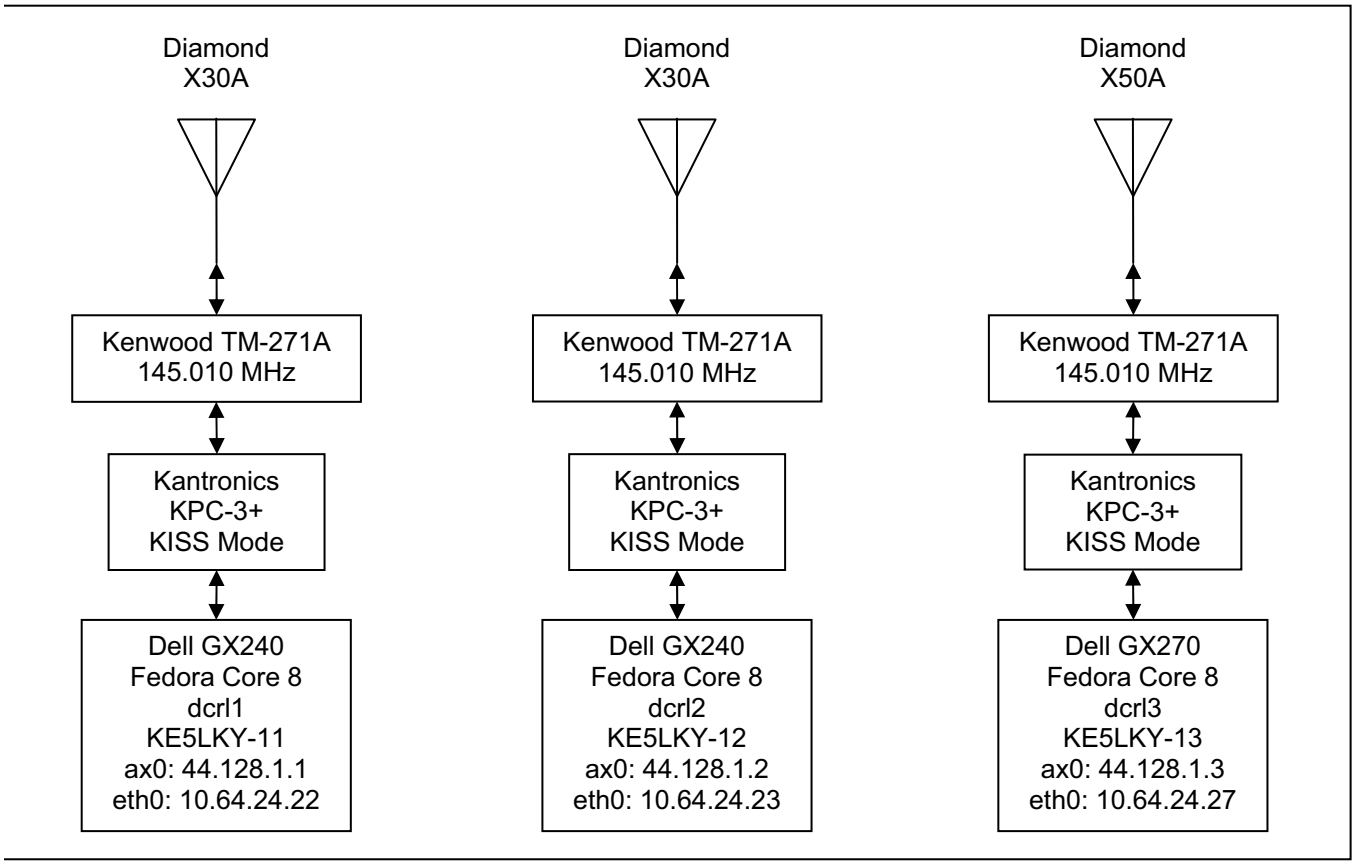

Figure 3: The physical configuration of computers, transceivers, terminal node controllers, and antennas owned and maintained by The University of Louisiana at Monroe Digital Communication Research Laboratory.

Using Udpcast to Multicast Data

The computers, transceivers, terminal node controllers, and antennas used in this research are owned and maintained by The University of Louisiana at Monroe (ULM) Digital Communication Research Laboratory (DCRL) (see Figure 3). As shown, we operate three similarly configured "rigs", each comprised of a Dell GX240 or GX270 computer, a Kantronics KPC-3+ terminal node controller, a Kenwood TM-271A transceiver, and a Diamond X30A or X50A antenna.

Our computers' AX.25 network interfaces (e.g. ax0) (Tranter, 2001) (Jones, 1996) have been assigned amateur packet radio addresses of the form 44.128.C.D because these addresses are designated for testing purposes (Amateur Packet Radio, n.d.). However, we could have assigned any un-routable addresses to our computers (Rekhter, 1996). Likewise, our computers' AX.25 network interfaces have been configured with call signs of the form KE5LKY-##, where KE5LKY is the author's United States of America Federal Communications Commission assigned call sign. Call signs are assigned based on information found within our computers' /etc/ax25/ax25d and /etc/ax25/axports files (see Figures A2 and A3 respectively in the Appendix). Our computers' Ethernet network interfaces (e.g. eth0) have been assigned static addresses associated with ULM's local intranet.

Before we initiate data transmission using Udpcast, all terminal node controllers are placed into KISS mode and the transceivers are tuned to the frequency 145.010 Megahertz (MHz). A Bourne shell script that will place a Kantronics KPC-3+ into KISS mode is shown in Figure A4 in the Appendix. A similar Bourne shell script that will take a Kantronics KPC-3+ out of KISS mode is shown in Figure A5 in the Appendix.

```
$ udp-sender --file file4KB.txt --interface ax0 -b 256 --async --
fec 4x1/1 --max-bitrate 1200 --log logfile.txt --bw-period 5
stripes=4 redund=1 stripesize=1
Udp-sender 2007-12-28
Using mcast address 236.128.1.1
UDP sender for file4KB.txt at 44.128.1.1 on ax0
Broadcasting control to 44.128.1.255
Ready. Press any key to start sending data.
Starting transfer: 00000029
bytes=4 096  re-xmits=0000000  (  0.0%)  slice=0004  4 096 -   0
Transfer complete.
$
```

Figure 4: Transmission of data using the udpcast-sender command on the computer dcrl1.

Example udp-sender and udp-receiver Command Execution

An example of the udp-sender command executed on the computer dcrl1 is shown in Figure 4. The file transmitted was comprised of 4 Kilobytes (KB) of American Standard Code for Information Interchange (ASCII) text and was multicasted over the computer dcrl1's AX.25 network interface ax0 to computers dcrl2 and dcrl3. The packet size was defined to be 256 because this value represents the maximum transfer unit (MTU) for all computers' ax0 network interface (see the Bourne shell script shown in Figure A6 in the Appendix). All Kantronics KPC-3+s' PACLEN values are also set to 256. The `--max-bitrate` was set to 1200 bits per second (bps), which is the Kantronics KPC-3+s maximum

data rate. The instantaneous bandwidth associated with this multicast data transmission was stored in the file `logfile.txt`.

An example of the udp-receiver command executed on the computer dcrl2 is shown in Figure 5. The size of the file received was 4 KB, so we named the file appropriately. We use the `--nosync` argument because we are writing data to a file. We also specify the use of the computer dcrl2's AX.25 network interface ax0 using the `--interface` argument.

```
$ udp-receiver --file file4KB.txt --nosync --interface ax0
Udp-receiver 2007-12-28
UDP receiver for file4KB.txt at 44.128.1.2 on ax0
Connected as #0 to 44.128.1.1
Listening to multicast on 236.128.1.1
bytes=4 096  (  0.00 Mbps)  4 096
Transfer complete.
$
```

Figure 5: Receipt of data using the udpcast-receiver command on the computer dcrl2.

Three tasks must be completed to initiate the transmission of data between the udp-sender and udp-receiver(s). First, execute the udp-sender command using the arguments shown in Figure 4. Once started, the output "`Ready. Press any key to start sending data.`" is generated (see Figure 4). Second, execute the udp-receiver command using the arguments shown in Figure 5 on all computers that will receive the file transmitted by the computer executing the udp-sender. The udp-receiver command generates the output "`Connected as ## to 44.128.1.1 Listening to multicast on 236.128.1.1`", which implies that the udp-receiver is waiting to receive data from the udp-server (see Figure 5). Last, press any key within the udp-server terminal window to start the transmission. Both the udp-sender and udp-receiver(s) show the amount of data that have been transmitted. If the transmission was successful, both the udp-sender and udp-receiver(s) generate the phrase "`Transfer complete`" (see both Figures 4 and 5).

Conclusions

As discussed, the RadioMirror and AltCast software can be used to multicast data over VHF/UHF and HF amateur packet radio network respectively. However, both are written to execute only on the Microsoft Windows operating system, and both require a dedicated computer to continuously transmit data. Additionally, while AltCast is actively supported, it appears RadioMirror is not. These three traits are shown in Table 1.

In this paper, we introduce the Udpcast file transmission tool. In comparison to RadioMirror and AltCast, we find that Udpcast is written for both the Microsoft Windows and UNIX/Linux operating systems, and is actively supported. More importantly, Udpcast does not require a dedicated computer to multicast data. The benefit to using Udpcast, versus RadioMirror or AltCast, is any given computer attached to an amateur packet radio network can execute the udp-sender command and multicast data to multiple udp-receivers. Likewise, the same computer can execute the udp-receiver command and receive multicasted data from another computer executing the udp-sender command. For this reason,

and those shown in Table 1, we advocate using Udpcast to multicast data over amateur packet radio networks during emergencies.

Table 1: A comparison of the RadioMiror, AltCast, and Udpcast software.

Software Traits	RadioMirror	AltCast	Udpcast
Supported Operating Systems	Microsoft Windows 95	Microsoft Windows XP	Microsoft Windows XP UNIX/Linux
Dedicated Multicast Server	Yes	Yes	No
Primary Use	VHF/UHF Packet Radio	HF Packet Radio	Transmission of Operating System Images
Actively Supported	No	Yes	Yes

While our current research uses Udpcast to multicast data over VHF amateur packet radio networks, we believe Udpcast can be used to multicast data over both UHF and HF amateur packet radio networks. As such, in the near future, we plan to purchase Kenwood TM-V71A transceivers and Kantronics KCP-9612+ terminal node controllers to explore multicasting data over UHF amateur packet radio networks using Udpcast.

Acknowledgements

The author graciously thanks E. Benson Scott II, M.D., AE5V, for providing answers to my numerous packet radio questions, and Allison M.D. Wiedemeier, Ph.D., for reading several drafts of this paper. The author's research is supported through funds provided by a ULM College of Business Administration Faculty Summer Research Grant, the Clarke M. Williams, Jr. Endowed Professorship in Computer Science sponsored by the ULM Foundation, and the ULM DCRL.

Appendix

Table A1: Multicast Software URLs.

Multicast Software	URL
RadioMirror	`ftp://ftp.tapr.org/pub/wa0ptv/RadioMirror.exe` `ftp://ftp.tapr.org/pub/wa0ptv/radiomirrorsource.zip`
AltCast	`http://www.comportco.com/hamsoft/AltCast.zip`
Udpcast	`http://udpcast.linux.lu/`

```
ax0     Link encap:AMPR AX.25  HWaddr KE5LKY-11
        inet addr:44.128.1.1  Bcast:44.128.1.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:256  Metric:1
        RX packets:986 errors:0 dropped:0 overruns:0 frame:0
        TX packets:1037 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:10
        RX bytes:201012 (196.3 KiB)  TX bytes:85586 (83.5 KiB)
```

Figure A1: Output returned by the "ifconfig ax0" command executed on the computer dcrl1.

```
[KE5LKY-11 VIA radio]
NOCALL * * * * * * L
default * * * * * * - root /usr/sbin/node node
```

Figure A2: The computer dcrl1's /etc/ax25/ax25d.conf file.

```
radio  KE5LKY-11  9600  256  7  9600 Kantronics KPC-3+ 145.010
```

Figure A3: The computer dcrl1's /etc/ax25/axports file.

```
#!/bin/sh
#####################################################################
# Script Name:   kiss-on.sh
#####################################################################
/bin/stty 9600 < /dev/ttyS0
/bin/echo -e "intface kiss\rreset\r" > /dev/ttyS0
```

Figure A4: A Bourne shell script that places a Kantronics KPC-3+ into KISS mode.

```
#!/bin/sh
#####################################################################
# Script Name:   kiss-off.sh
#####################################################################
/bin/stty "9600" < /dev/ttyS0
/bin/echo -e "\r\xC0\xFF\xC0\r" > /dev/ttyS0
```

Figure A5: A Bourne shell script that takes a Kantronics KPC-3+ out of KISS mode.

```
#!/bin/sh
#####################################################################
# Script Name:   ax0-on.sh
#####################################################################
/usr/sbin/kissattach /dev/ttyS0 radio 44.128.1.1
/sbin/ifconfig ax0 44.128.1.1 broadcast 44.128.1.255 netmask
255.255.255.0 arp mtu 256 up multicast
/sbin/route del -net 44.128.1.0 netmask 255.255.255.0 ax0
/sbin/route add -net 44.128.1.0 netmask 255.255.255.0 irtt 12000
window 1792 ax0
/usr/sbin/ax25d &
/usr/sbin/mheardd -l -f
```

Figure A6: A Bourne shell script that configures the computer dcrl1's network interface ax0, and starts the ax24d and mheardd daemons.

```
#!/bin/sh
#####################################################################
# Script Name:   ax0-off.sh
#####################################################################
/bin/ps -ef | /bin/grep "/usr/sbin/ax25d" | /bin/awk '{print $2}' |
/usr/bin/xargs -i kill -9 {}
/bin/ps -ef | /bin/grep "/usr/sbin/kissattach" | /bin/awk '{print
$2}' | /usr/bin/xargs -i kill -9 {}
/bin/ps -ef | /bin/grep "/usr/sbin/mheardd" | /bin/awk '{print $2}'
| /usr/bin/xargs -i kill -9 {}
```

Figure A7: A Bourne shell script that terminates the ax24d, kissattach, and mheardd daemons.

References

Amateur Packet Radio, net 44, and AMPR.ORG. (n.d.). Retrieved July 16, 2008, from http://www.ampr.org/.

Brabham, Charles (N5PVL). (2005). *Utilizing Amateur Multicast Protocol*. Retrieved July 16, 2008, from http://www.uspacket.org/multicast.htm.

Comer, Douglas E. (2007). *The Internet Book, 4th Edition*. Upper Saddle River, NJ: Pearson Prentice Hall Publisher.

Comer, Douglas E. (2000). *Internetworking with TCP/IP, Volume 1, 4th Edition*. Upper Saddle River, NJ: Pearson Prentice Hall Publisher.

Drom, R.. (1997, March). *Dynamic Host Configuration Protocol*. Request For Comments (RFC) 2131. Retrieved July 16, 2008 from http://www.ietf.org/rfc/rfc2131.txt.

Hansen, John. (1997-A, April 20). *RadioMirror: Rethinking Packet Radio*. Retrieved July 16, 2008 from ftp://ftp.tapr.org/pub/wa0ptv/RadioMirror.exe.

Hansen, John. (1997-B, April 5). *Technical Documentation for the RadioMirror Server*. Retrieved July 16, 2008 from ftp://ftp.tapr.org/pub/wa0ptv/RadioMirror.exe.

Hansen, John. (1997-C, April 20). *RadioMirror Server: Sysop Information*. Retrieved July 16, 2008 from ftp://ftp.tapr.org/pub/wa0ptv/RadioMirror.exe.

Hansen, John. (1997-D, April 20). *RadioMirror Client Documentation*. Retrieved July 16, 2008 from ftp://ftp.tapr.org/pub/wa0ptv/RadioMirror.exe.

Information Sciences Institute (ISI). (1981, September). *Internet Protocol*. Request For Comments (RFC) 791. Retrieved July 16, 2008 from http://www.ietf.org/rfc/rfc791.txt.

Jones, Greg (Ed.) (WD5IVD). (1996). *Packet Radio: What? Why? How? Articles and Information on General Packet Radio Topics*. Tucson, AZ: Tucson Amateur Packet Radio Corporation Publisher.

Knaff, Alain. (2006, March 20). Multicasting over satellite. *Udpcast Forums*. Retrieved July 16, 2008, from http://udpcast.linux.lu/pipermail/udpcast/2006-March/000493.html.

Knaff, Alain. (2005, May 14). *Udpcast Commandline Options*. Retrieved July 16, 2008, from http://udpcast.linux.lu/cmd.html.

Mogul, Jeffery. (1984-A, October). *Broadcasting Internet Datagrams*. Request For Comments (RFC) 919. Retrieved July 16, 2008 from http://www.apps.ietf.org/rfc/rfc919.html.

Mogul, Jeffery. (1984-B, October). *Broadcasting Internet Datagrams in the Presence of Subnets*. Request For Comments (RFC) 922. Retrieved July 16, 2008 from http://www.apps.ietf.org/rfc/rfc922.html.

Plummer, David C.. (1982, November). *An Ethernet Address Resolution Protocol*. Request For Comments (RFC) 826. Retrieved July 16, 2008 from http://www.ietf.org/rfc/rfc826.txt.

Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. J., & Lear, E.. (1996, February). *Address Allocation for Private Networks*. Request For Comments (RFC) 1918. Retrieved July 16, 2008 from http://www.apps.ietf.org/rfc/rfc1918.html.

Tanenbaum, Andrew S.. (2003). *Computer Networks, 4th Edition*. Upper Saddle River, NJ: Pearson Prentice Hall Publisher.

Tranter, Jeff (VE3ICH). (2001, September 19). *Linux Amateur Radio AX.25 HOWTO*. Retrieved July 16, 2008, from http://tldp.org/HOWTO/AX25-HOWTO/.

Wiedemeier, Paul (KE5LKY). (2007, September). Performance Modeling of TCP and UDP over Amateur Packet Radio Networks using the ns-2 Network Simulator. (pp. 141-154). *Proceedings of the 26th ARRL and TAPR Digital Communications Conference*.