Chapter 14 FILE TRANSFER

File transfer concerns the exchange of computer "files" with other packet stations.

14.A TEXT FILE TRANSFER

A text file is one which contains ASCII characters representing text. Only a few specific control characters are included. These are usually such things as carriage returns.

Virtually every computer includes this basic file type. On MS-DOS machines, when you do a *`TYPE filename.ext`* and get out a screen listing as it was typed, it is a text file. C64, Macintosh, and all others of which the author is aware have similar file types. This does not mean that they can be read by exchanging disks. But they can be exchanged by packet. This is because, to the computer, the packet input and output is as if it were typed on that machine.

14.A.1 SENDING TEXT FILES: Almost every terminal program has a "SEND TEXT" option. It may masquerade by different names. Sometimes it appears as "UPLOAD". In general, you simply specify the file name (including extensions for non-Macintosh users).

The computer then sends the text out the serial port to your TNC as if you were typing it. Depending on the program, it may not always end by itself. You may need to specify when the end has occurred.

The only "trick" is that the station at the other end usually must be prepared before the sending starts. See the next section.

14.B.2 RECEIVING TEXT FILES: Almost every terminal program has a "RECEIVE TEXT" option. It may masquerade by different names. Sometimes it appears as "DOWNLOAD" or "CAPTURE". In general, you simply specify the file name (including extensions for non-Macintosh users).

In general, you must initiate CAPTURE prior to the arrival of the first text. Computers which support mouse text selection (such as Macintosh or Windows) may permit you to choose the text on the computer screen after it has been received, then copy it, and paste it into another program (such as a text editor). It can then be edited, if appropriate, then saved.

14.A.3 WHY USE TEXT FILES?: Your BBS messages will look a lot better if you prepare them before-hand with a text editor. Then, when it is all "spiffed up", you can send it. It goes a lot faster, also, than when you type it directly.

14.A.4 CREATING TEXT FILES: Most editors ("word processors, by another name) will produce text files. Simply type the message and save it as a "plain text" file. For example, "BRIEF" (a text editor widely used by programmers) inherently produces plain text. "Word Perfect" allows you to save and input text files using the "TEXT IN/OUT" (control-F5). Microsoft "Word For Windows" does it if you use a file extension of *.TXT*. "PC-Write", one of the popular share-ware editors, does it directly. It appears that many of the Macintosh editors give you the "text" option from the "Save As..." item in the File Menu.

How do you tell if it is a plain text file? The easiest way on a "DOS" machine is to "TYPE" the file (from DOS, *TYPE filename.ext*). If it appears on the screen without strange characters (happy-faces and such), then you have done it.

Please, do not rely on line-wrap by the editor. Many editors in text mode leave out carriage returns at the wrap points. If the person receiving such a file has one of the several editors which cannot easily handle strings of characters longer than 256 without carriage returns, its a mess!

14.A.5 BBS MESSAGES: You can greatly simplify operation with a BBS if you include BBS control messages within your text. Consider the following simple message:

```
S KA7AGH @ KA7AGH.OR.USE.NA
NODES
Hi Jim -
How is progress going on the new node?
73 de Jim
/EX
b
```

A message of this type is sent from the basic BBS prompt. The first line is the "addressed send command" to the BBS. It returns with a request for the "subject" which is satisfied by the second line. Once the subject line is satisfied, it accepts message text; lines 3, 4, & 5 are recognized as message. The "*/EX*" terminates the message and the "*b*" disconnects you from the BBS! All you have to do in this scheme is connect to the BBS!

14.B NON-TEXT FILES

Non-text files are all those other computer files. They may be programs. They may be graphic files ranging from bit-map "paint" to "AutoCAD" or other technical files. In addition, Macintosh files have attached to them a "resource fork" which contains the "icon" for the file as well as information about the kind of program which created the file. In these cases, the file contains many non-text characters. As usual, there are several ways to deal with this situation.

14.B.1 CONVERSION TO TEXT: There are programs which will convert a non-text file into a text-only form. For the Macintosh, "BINHEX" encoding within "Stuffit" is one. A procedure named "UUENCODE" and "UUDECODE" does this for DOS files.

One of the problems with conversion to text is that the file size gets bigger (by about 2X). This is usually dealt with by compressing the original file. For most Macintosh files, "Stuffit" reduces files to about 50% of their original size. For DOS machines, the share-ware program "PKZIP" is a popular compression choice. Both Stuffit and PKZIP permit you to bundle together several files into a single "batch".

Once a text-version of the file has been created, it can be sent as described in section 14.A. The person receiving the file must have access to the proper decoding program. Stuffit both encodes and decodes BINHEX. Likewise the appropriate uncompression is required if this was done in the first place.

It should be pointed out that after conversion to text form, any computer can handle the file. This does not, however, give PCs the ability to use Macintosh files!

14.B.2 TRANSPARENT TRANSFERS: One of the problems with file transfers is the interpretation of file characters by the TNC. For example, if the the file happens to have a character which is the same as "control-C", the TNC immediately switches out of CONVERSE mode into COMMAND mode. At this point, everything grinds to a halt. TRANSPARENT mode was designed specifically for this circumstance. When in this mode, all characters are ignored by the TNC except for a very special set of characters which is not likely to be encountered in a file. You may, in fact, have to use TRANSPARENT mode with some of the protocols describe in the following section.

For TNC2 clones, this mode is entered from the COMMAND mode by typing `cmd: TRANS`. From this point on, the normal "control-C" has no effect! There are no "send packet" characters like the carriage-return in CONVERSE mode. Instead, a packet is sent when there are a certain number of characters accumulated in the TNC or when a certain time has passed. The number of characters is set in TNC2s with PACLEN; the time interval is set in TNC2s with PACTIME. The TNC does not echo these characters to the terminal screen as it normally does in other modes.

No matter how the TNC is set, it sends and receives full 8-bit characters. You should have the terminal program set to operate with 8 bits and NO parity.

TRANSPARENT mode can be exited in TNC2s in several ways. Some terminal programs allow the sending of a "break character". In actuality, this is not a true character and is not in the ASCII table. Sending this condition usually requires a menu selection in the terminal program.

The second exit from TRANSPARENT mode in TNC2s is governed by a parameter called CMDTIME. After the last character is sent, you must wait a time equal to CMDTIME. Then, during a the time CMDTIME to 2*CMDTIME, send the command character 3 times. Then, during the interval from 2*CMDTIME to 3*CMDTIME from the last character, there can be no keyboard entries. If you have been successful, you will see the command prompt! I find it useful to use a value for CMDTIME which is not too short (so that it is missed) or too long (which is frustrating). For me, 5 seconds has worked. Thus, I wait at least 5 seconds after the last file character, then within the NEXT 5 seconds, type 3 control-Cs, then wait another 5 seconds for the prompt.

14.B.3 FILE TRANSFER PROTOCOLS: Most terminal programs which were originally written for use with telephone modems support a variety of "modem" protocols. These include XMODEM, YMODEM, ZMODEM, and KERMIT. We should not forget, here, FTP used by TCP/IP stations. The same protocol must be used by both stations involved.

Among packet users, a number of programs have available a protocol called "YAPP" (Yet Another Packet Protocol). Another is PZC binary exchange protocol supported by LanLink software.

The author has not had much experience with these protocols, yet. It is quite obvious, however, that it may take some experimenting to make them work through TNCs. In part, this is due to the delays between send and acknowledge which is not present when using telephone modems.

14.C FILE TRANSFER CIPHER?

The question has occurred to the author and others about whether compressed, "binhexed", or plain "modem" file transfers fall under the FCC's prohibition against "ciphers".

The point to begin when considering this issue is, of course, Part 97 of the FCC regulations which governs amateur radio. The specific part is:

> 97.113 (d) No station shall transmit: music; radio-communications or messages for any purpose, or in connection with any activity, that is contrary to federal, state, or local law; messages in code or ciphers where the intent is to obscure the meaning (except where specifically excepted elsewhere in this Part); obscene, indecent, or profane words, language, or meaning; and/or false or deceptive messages or signals.

It should be quite clear that "intent" is a major component of the prohibition. When a file transfer is done, the intent is not to obscure meaning; instead, it is to carry out the most efficient transfer.

Furthermore, just because it may appear as jibberish to someone else, that is not evidence of code or cipher. To someone without a TNC, a packet transmission is undecipherable! Just because somebody else

cannot immediately translate it does not make it a prohibited transmission.

The author wants to thank Mike Curtis (wd6ehr@k6ve .#soca.ca.usa.na) for the insight concerning this issue.

14.D FILE TRANSFER SUMMARY

File transfers can be carried out on packet. It can be done as a text file or using one of the file transfer protocols. Be prepared to experiment.