Network Working Group
Request for Comments: 529
NIC: 17165
References: RFCs 454, 513,
NIC # 15372

A. McKenzie
B. Thomas
R. Tomlinson
BBN-TENEX
K. Pogran
MIT-MULTICS

29 June 1973

A Note on Protocol Synch Sequences

This note is motivated by Wayne Hathaway's RFC 513 which comments on the interpretation of the TELNET SYNCH sequence (INS/Data Mark). We agree with Wayne's observation that the phrase "interesting things", as it appears and is explained in the TELNET Protocol Document (NIC# 15372), is much too imprecise to appear in a protocol specification. However, we disagree with his proposal that the interpretation of the TELNET SYNCH sequence should be redefined. Hathaway's comments led us to examine the notion of "interesting things" with respect both to TELNET protocol and to protocols built upon it.

We feel that the definition of the TELNET SYNCH sequence in the TELNET Protocol Document is the proper one [1]. More important, we feel that the (potential) difficulties with respect to the TELNET SYNCH sequence noted in RFC 513 are not the reflection of a TELNET design flaw but rather reflect misuse of the TELNET SYNCH sequence by "higher level" protocols (in particular FTP) that are based on TELNET.

The remainder of this note examines the notion of a synch sequence and suggests an approach to the design of protocols which are to use the TELNET protocol as a basis.

The reason for defining a synch sequence for a protocol is to provide a mechanism by which signals, represented as characters, that for one reason or another are "stuck" in the pipeline between the sender and the protocol interpreter, can promptly be brought to the attention of the interpreter. Flow through the pipeline is, of course, controlled by the receiver; the process operating the interpreter may be doing something else at the moment, and may not be paying attention to the incoming data stream. The sender would like to get the attention of the receiving process, to have it read its incoming data stream and take action as directed by the "interesting" characters in that stream, which will, in general, be protocol commands. To accomplish this, a "SYNCH sequence" is transmitted. A synch sequence consists of:

1. An "out of band" signal which serves to get the attention of the protocol interpreter; and

2. An "in band" marker which serves to mark how much of the data stream is to be processed by the protocol interpreter in response to the "out of band" signal.

For the TELNET protocol the "out of band" signal is the INS of Host-Host Protocol and the "in band" marker is the TELNET Data Mark character (DM). Ignoring for the moment the use of TELNET as a basis for higher level protocols (such as FTP), the class of characters "interesting" to a TELNET interpreter is the set of TELNET commands (including the commands for option negotiation and sub-negotiation [2]).

One might reasonably argue that this class could be enlarged by a server Host to include the set of signals of interest to the terminal support software of that particular Host. For example, in case of TENEX such a set would include the "terminal interrupt" characters enabled by the process reading from the TELNET connection (e.g., ^C, ^T, etc.). Other hosts, such as Multics, might look only for the TELNET commands, such as Interrupt Process (IP), Abort Output (AO), etc. Whether or not one chooses to consider additional signals as interesting during the processing of a TELNET SYNCH sequence should cause the implementer no problem:

> He must treat all TELNET commands as interesting by interpreting them. He may choose either to ignore such additional signals or to pass them on to the process; in either case there is no vagueness since the implementer knows which characters his terminal support software considers interesting.

The difficulty noted in RFC 513 concerning the vagueness of "interesting things" occurs when a higher level protocol makes use of the TELNET SYNCH sequence to force commands of interest to it through to its interpreter. A higher level protocol designed in such a way represents a violation of the protocol layering discipline:

> The TELNET SYNCH mechanism is being misused by attempting to give it meaning at two different levels of protocol.

The problem stems from the fact that, in general, a (increasing) number of different higher level protocols can be designed with

TELNET as a base. A TELNET interpreter has no way of knowing the higher level protocol interpreter (if any) to which it is passing characters, and therefore, can not tell which things are "interesting" to the higher level protocol interpreter. That is, just as an NCP should not have to know whether the data it handles is

for a TELNET connection, an FTP data connection, etc., a TELNET interpreter should not be required to know the kind of process for which it is handling characters. This should, in fact, result in a simplification of the design and implementation of TELNET protocol interpreters.

This difficulty can be resolved by proper design of protocols that make use of TELNET as a base. In particular, if in such a higher level protocol it is important to be able to force commands through to the protocol interpreter, the higher level protocol should include its own synch sequence: i.e., an "out of band" signal used with an "in band" data mark. The TELNET protocol provides the Interrupt Process character (IP) for use as an "out of band" signal. A synch sequence for a protocol built upon TELNET would be:

1. Insert the TELNET IP control character into the data stream;

2. Insert the higher level protocol data mark character (HDM) into the data stream following whatever higher level protocol commands are important at the time.

Receipt of the IP TELNET command causes the higher level protocol interpreter to be interrupted, enabling it to scan the data stream (up to and including the HDM) for commands it considers important.

As an example, consider the case of the File Transfer Protocol (RFC 454) and the problem of aborting a file transfer in progress. To accomplish such an abort the FTP user (process) should:

1. Send the TELNET IP character;

2. Send the TELNET SYNC sequence, that is:

   a. Send the TELNET Data Mark (DM);

   b. Send the Host-Host Protocol INS;

3. Send the FTP ABOR command; and

4. Send the FTP data mark character [3].

The user (or process acting on his behalf) must transmit the TELNET SYNCH sequence of step 2 above to ensure that the TELNET IP gets through to the server's TELNET interpreter.

Endnotes

[1] I.e., any TELNET commands appearing before the Data Mark are to be interpreted; the Data Mark is to be used to terminate the scan initiated by the INS; characters that are not TELNET commands may be discarded or passed to the user process as the implementer sees fit.

[2] We support Hathaway's proposal to fully parenthesize sub-negotiations. Further, we believe that the "closing parenthesis" should be a new command rather than a second SB command; this will aid the receiver in recovering from errors, either in parsing at the receiver or in generation at the transmitter. We disagree with his proposal that sub-negotiations be discarded when encountered during processing of a TELNET SYNCH.

[3] For FTP such a data mark character has not yet been defined and, in fact, may not be necessary under the constraint that the FTP command interpreter should look for exactly one command after being interrupted; this is consistent with the general command-reply orientation of FTP.