Network Working Group
Request for Comments # 595
NIC # 20617
References: NIC # 20812

Wayne Hathaway
AMES-67
12 Dec 1973

Some Thoughts in Defense of the TELNET Go-Ahead

This note is a reply to Edward Taft's "Second Thoughts on TELNET Go-Ahead" (NIC #20812). Specifically, I will attempt to show the following about the three main directions of his objections:

1. It is the idea of line-at-a-time systems which are esthetically unappealing, not the GA mechanism. This may be a valid point, but given the large number of such systems on the net, it would seem a rather academic one.

2. The specified GA mechanism will in fact work very well between (reasonably implemented) line-at-a-time systems, and should provide significant help elsewhere.

3. While the GA mechanism may not be correct in all cases, it can provide significant advantages fro the line-at-a-time systems and users.

My comments will be arranged under the original headings from the subject RFC (NIC #20812).

"TECHNOLOGY"

The definitions of "half-duplex" and "reverse break" are satisfactory. Two points should be made regarding "reverse break", however. First: having reverse break on the terminal is of course not sufficient; the operating system must support it. As "support" is equivalent to "require" in this context, it is not too surprising that some systems do not in fact do this. That is, there are systems which will not type through an unlocked keyboard until the user manually turns the line around, and the operational problems with such systems are much less than might be assumed. Second, at least on IBM 2741's and equivalent, the line turnaround takes a significant amount of time, during which user-typed characters may be missed or garbled. In fact, a fairly standard mode of operation with systems that use reverse break (including TIP's) is to automatically enter a "line delete" character and start over every time the reverse break is used while typing, which can hardly be called esthetic. One solution to this problem would be for the system to not use reverse break once the user has begun typing (as suggested near the end of NIC #20812), but most systems (including TIP's) do not do this.

Some discussion is also warranted at this point about line-at-a-time systems (hereafter abbreviated as LAAT systems). One prime reason for LAAT operation is to avoid the overhead of interrupting the CPU (and possibly the user process) for every character typed. Instead, characters are buffered (in a controller, a front-end computer, etc) until some "end-of-line" signal is received; they are then passed to the system in a group. This means that the system is totally unaware that any typing has occurred until the "end-of-line" signal is sent; a partially completed line will literally never be recognized.

"ESTHETIC OBJECTIONS TO GA"

From the above, I feel that one can see that it is the operating mode of a system rather than the type of features of its terminals which determines whether GA is useful or not. For example, IBM front-ends handle Teletypes in LAAT mode, while the TIP attempts to run 2741's as full-duplex devices (with something less than "a very good job at turning the line around," from my experience).

At any rate, the half-duplex/full-duplex debate can go on forever -- the problem here is to try to smooth the way for users on local LAAT systems connected to foreign systems of varying characteristics.

"WHY GA WON'T WORK"

As mentioned, in LAAT systems no terminal input is recognized until the specified "end-of-line" character is entered, preceding characters having been buffered in a front-end etc. This can of course be carried over into server TELNET: incoming network messages can be buffered at a very low level in the NCP awaiting a TELNET end-of-line signal. User processes wanting input would remain blocked until the end-of-line is received, rather than being handed each character as it is read. In fact, this is the implementation in all of the LAAT systems with which I am familiar. The reason for doing this is obvious: many hosts continue to send single characters even in LAAT systems, resulting in a significant increase in overhead. Equally obvious is the fact that in this mode the GA mechanism will function quite well, in fact as well as turning the line around to unlock the keyboard of a local terminal.

This further brings us what is to me one of the main reasons for the GA mechanisms: the need for a scheme similar to the above for user TELNET's. The problem is as follows: a user TELNET on a LAAT system has no required "end-of-message" signal for incoming server-generated messages, and so is required to read each character as it comes, with attendant overhead. In addition, the user process is forced to write each character as it arrives, since it never knows when the server will stop sending. On systems which support reverse break this results in little more than erratic terminal behavior, but on systems which do not support it, it is left up to the user to manually turn the line around (which he can do reasonably well with "attention"). Of course the overhead of handling character-at-a-time input on a line-at-a-time system is also significant.

This is what I see as the most valuable reason for the GA mechanism, as was noted in NIC#20812: it is not so much a request for input as an assurance (although not an irrevocable one) that the server is through sending output. In fact, that is what the name implies to me: go ahead, it's your turn to type, I'm through for a while. Perhaps some of the objections would be eased if this aspect were given more emphasis? As an aside, the problem of spontaneous system messages that might be generated after a GA is sent is not a major one in practice, as the user will surely see the message as soon as he manually turns the line around (enters his next input line). Note of course that the spontaneous message should also have a GA following, to serve as "end-of-message" to the receiving NCP. Further, if the user system supports reverse break, it can deliver the message as soon as it likes.

"IMPLEMENTATION PROBLEMS"

Perhaps the above discussion will remove some of the objections from this section? The GA should be sent when a system has a "reasonable assurance" that it is not going to generate additional output (eg, after a system prompt). If this assumption turns out to be false there is no problem: the additional output is simply sent, also followed by a GA. The main point here is that known multi-line output (eg, editor printout, message-of-the-day, SYSTAT) would have only the single GA on the end.

Finally about linking. I agree that on a system like TENEX links should probably not use GA's, but have you been involved in a link to a user on a LAAT system? The LAAT user is of course generating complete lines, which are sent over such a link. This can be very disconcerting to a character-at-a-time user, who all of a sudden has dozens of characters printing at full terminal speed

(often against the right margin). And I can hardly imagine linking from a 2741 on a TIP to a TENEX user: one would never get anything typed, with all the line turnarounds.

In fact, in all the linking that I have done from our (LAAT) system to TENEX we have very quickly agreed on a manual GA mechanism (eg, "over"). For straight conversational links I do not feel that it is unreasonable to have a simple way to ask your local process to send a GA (although GA is mostly defined in the server-to-user context, which breaks down somewhat here). One further supportive comment: a spoken conversation is of course line-at-a-time, with "obvious cues" (pauses, questions, etc.) serving as GA's. The situation is of course quite livable, even when spontaneous talk overrides the GA ("Oh, before you answer that, ..."). This occasionally results in the need to repeat a line, in an exact analogy to the problem of lines garbled by a reverse break or printed against the right margin.

The problem of links containing system output intermixed with user input is more difficult. In any implementation it seems the LAAT user will have to be aware of what is happening and manually control his terminal to some extent, but that is reasonable when dealing with an "alien" system. More definition work is called for in this area, to solve the efficiency problem for LAAT hosts.

"A PROPOSAL"

The proposal appears on the surface to be that "suppress GA" should be the NVT default, which would be perfectly acceptable to me (and I would suppose to other LAAT users): two additional messages upon opening a connection is a small enough price.

But in fact that is not the proposal at all -- the proposal is really to remove the requirement that all server systems implement the GA. This I object to very strenuously since, as I feel I have shown, the benefit to the LAAT system and user of GA far outweigh its cost to other types of server systems. And of course the expense of going into "suppress GA" mode when appropriate is truly negligible.

The proposal for having those user TELNET's which do not support reverse break retain permanent control over terminals is also weak, even without GA. In our current implementation the assumption is that for each line entered by the user, the server system will respeed with something. Control of the terminal is thus retained after input until some output is received and printed, when the terminal is again made available for input. The "attention" key is defined as a toggle switch to control the terminal keyboard: if pressed while the keyboard is unlocked (open for input) it will lock it until the next available output message and if pressed while keyboard is locked

it will be unlocked for input. The user may also enter a true unlocked mode, in which the terminal is always returned to him for additional input (after printing all queued output). This is used, for example, for input to a text editor which does not issue prompts for each line, the mode may be changed at any time by the user, and the "attention" key may of course be used to retrieve expected but infrequent output. This combination mode has proven much more effective than the proposed "user must press attention for all input" mode. Of course the addition of GA will allow the user process to wait for a "complete" reply before printing anything, which will eliminate much of the use of "attention", as well as improve system efficiency.

A GRIPE OF MY OWN

I would like to add one complaint of my own at this point. The implementation schedule for the new TELNET called for a date of July 1 when systems should accept new TELNET without causing errors. This date was presumably agreed to by responsible representatives of effectively all active network sites. My system has been using the new TELNET since early September (significantly after the allowable date) but I have been forced to disable all server-generated GA's because (among other problems) TENEX "SNDMSG" does not work when GA's are received over the FTP TELNET control connection. Disabling the GA's was of course required in order for me to receive any deliveries from the Network Information Center. This brings up three points. First, I sincerely hope that service functions like the NIC intend to accept the new TELNET protocol by the January 1 implementation date. Second, in response to RFC#593 by Alex McKenzie and Jon Postel, I do not feel that attempting to use a second TCP socket for "new TELNET" will work, because of the use of TELNET by FTP. In fact, it does not seem too difficult to make a "compatible" TELNET which will accept either mode (which sites have had since July 1 to do) and I feel that this is the most reasonable implementation method, even if it makes the January 1 date impractical. And third, perhaps sites should be more cautious about commitments to implementation schedules in the future.