Network Working Group J. Postel
Request for Comments: 860 J. Reynolds

ISI
Obsoletes: NIC 16238 May 1983

TELNET TIMING MARK OPTION

This RFC specifies a standard for the ARPA community. Hosts on the ARPA
Internet are expected to adopt and implement this standard.

1. Command Name and Code

TIMING-MARK 6

2. Command Meanings

IAC DO TIMING-MARK

The sender of this command REQUESTS that the receiver of this command return a WILL TIMING-MARK in the data stream at the "appropriate place" as defined in section 4 below.

IAC WILL TIMING-MARK

The sender of this command ASSURES the receiver of this command that it is inserted in the data stream at the "appropriate place" to insure synchronization with a DO TIMING-MARK transmitted by the receiver of this command.

IAC WON'T TIMING-MARK

The sender of this command REFUSES to insure that this command is inserted in the data stream at the "appropriate place" to insure synchronization.

IAC DON'T TIMING-MARK

The sender of this command notifies the receiver of this command that a WILL TIMING-MARK (previously transmitted by the receiver of this command) has been IGNORED.

3. Default

WON'T TIMING-MARK, DON'T TIMING-MARK

i.e., No explicit attempt is made to synchronize the activities at the two ends of the TELNET connection.

4. Motivation for the Option

It is sometimes useful for a user or process at one end of a TELNET connection to be sure that previously transmitted data has been completely processed, printed, discarded, or otherwise disposed of. This option provides a mechanism for doing this. In addition, even if the option request (DO TIMING-MARK) is refused (by WON'T TIMING-MARK) the requester is at least assured that the refuser has received (if not processed) all previous data.

As an example of a particular application, imagine a TELNET connection between a physically full duplex terminal and a "full duplex" server system which permits the user to "type ahead" while the server is processing previous user input. Suppose that both sides have agreed to Suppress Go Ahead and that the server has agreed to provide echoes. The server now discovers a command which it cannot parse, perhaps because of a user typing error. It would like to throw away all of the user's "type-ahead" (since failure of the parsing of one command is likely to lead to incorrect results if subsequent commands are executed), send the user an error message, and resume interpretation of commands which the user typed after seeing the error message. If the user were local, the system would be able to discard the buffered input; but input may be buffered in the user's host or elsewhere. Therefore, the server might send a DO TIMING-MARK and hope to receive a WILL TIMING-MARK from the user at the "appropriate place" in the data stream.

The "appropriate place", therefore (in absence of other information) is clearly just before the first character which the user typed after seeing the error message. That is, it should appear that the timing mark was "printed" on the user's terminal and that, in response, the user typed an answering timing mark.

Next, suppose that the user in the example above realized that he had misspelled a command, realized that the server would send a DO TIMING-MARK, and wanted to start "typing ahead" again without waiting for this to occur. He might then instruct his own system to send a WILL TIMING-MARK to the server and then begin "typing ahead" again. (Implementers should remember that the user's own system must remember that it sent the WILL TIMING-MARK so as to discard the DO/DON'T TIMING-MARK when it eventually arrives.) Thus, in this case the "appropriate place" for the insertion of the WILL TIMING-MARK is the place defined by the user.

It should be noted, in both of the examples above, that it is the responsibility of the system which transmits the DO TIMING-MARK to discard any unwanted characters; the WILL TIMING-MARK only provides help in deciding which characters are "unwanted".

5. Description of the Option

Suppose that Process A of Figure 1 wishes to synchronize with B. The DO TIMING-MARK is sent from A to B. B can refuse by replying WON'T TIMING-MARK, or agree by permitting the timing mark to flow through his "outgoing" buffer, BUF2. Then, instead of delivering it to the terminal, B will enter the mark into his "incoming" buffer BUF1, to flow through toward A. When the mark has propagated through B's incoming buffer, B returns the WILL TIMING-MARK over the TELNET connection to A.

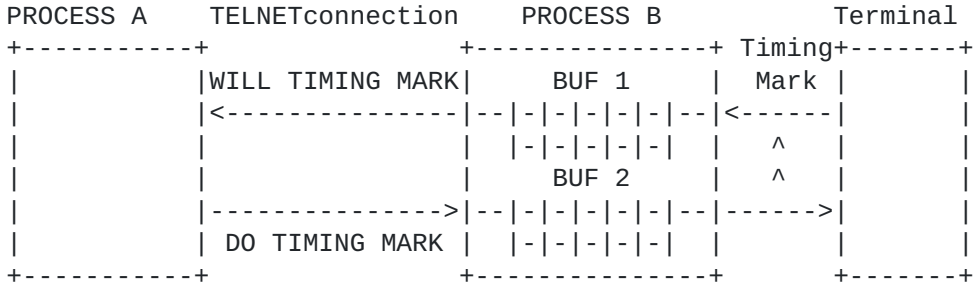

Figure 1

When A receives the WILL TIMING-MARK, he knows that all the information he sent to B before sending the timing mark been delivered, and all the information sent from B to A before turnaround of the timing mark has been delivered.

Three typical applications are:

A. Measure round-trip delay between a process and a terminal or another process.

B. Resynchronizing an interaction as described in section 4 above.

A is a process interpreting commands forwarded from a terminal by B. When A sees an illegal command it:

i. Sends <carriage return>, <line feed>, <question mark>.

ii. Sends DO TIMING-MARK.

iii. Sends an error message.

iv. Starts reading input and throwing it away until it receives a WILL TIMING-MARK.

v. Resumes interpretation of input.

This achieves the effect of flushing all "type ahead" after the erroneous command, up to the point when the user actually saw the question mark.

C. The dual of B above. The terminal user wants to throw away unwanted output from A.

i. B sends DO TIMING-MARK, followed by some new command.

ii. B starts reading output from A and throwing it away until it receives WILL TIMING-MARK.

iii. B resumes forwarding A's output to the terminal.

This achieves the effect of flushing all output from A, up to the point where A saw the timing mark, but not output generated in response to the following command.