Time-out Mechanism in the Host-Host Protocol

On sending a message to a foreign site, the following situations can occur:

1. Destination IMP down - Type 7 message is returned

2. Destination IMP up but destination IMP-HOST interface is down - Type 7 message is returned.

3. Destination IMP and IMP-HOST interface up, but IMP-HOST interface is not taking messages - Type 9 message is returned after IMP time-out (ask BBN for time).

4. Destination IMP and IMP-HOST interface up and IMP-HOST interface is taking messages - Type 5 (rfnm) message is returned.

A suggestion for handling type 7 and type 9 messages has been made in NWG/RFC #117. In this document we would like to discuss in detail the problem: what should happen to the HOST-HOST protocol on receiving a rfnm?

When a NCP sends out a STR or RTS control command on a pair of sockets and gets a rfnm back, this pair of sockets will be in a wait-match state. Everything is fine if a matching RTS or STR, or CLS is returned after a reasonable amount of delay. Trouble will arise when nothing is returned after a long time.

This can happen if the NCP is not running at all but its host is taking in messages (e.g. UCLA's host will receive messages even if the NCP is not running), or if the NCP is running very slowly. The same problem exists on sending out a CLS control command and a matching CLS is never returned. The trouble is that resources are tied up, e.g. sockets, links and table space in the NCP; and one would like to release these resources. In our implementation, when a user does a CLOSE, we can't release the sockets until the matching CLS is returned. This protects us from getting confused if a seconds request is made for the same pair of sockets. This problem can be solved by including a time-out mechanism in the Host-Host protocol. This operates as follows:

a.   On sending out a STR or RTS and if you do not get back a match-
     ing RTS or STR, or a CLS in T time units a CLS will be sent.
     After sending the time-out CLS race condition can be avoided
     by ignoring the matching RTS or STR that arrives before the
     matching CLS.

b.   On sending out a CLS (any kind, including the time-out CLS),
     and if you do not get back a matching CLS in T time units, the
     matching CLS is assumed to have returned. However, if a RTS or
     STR is sent on the same pair of sockets anytime after the time
     out and before a CLS is returned, and then we receive the CLS,
     there is no way to determine whether this returning CLS is for
     matching the previous CLS or for refusing the RTS or STR. (See
     the figure for detail). So far we could not solve this race
     condition except by assigning sequence number to connection
     throughout the Network which we don't think is a good solution
     at all. Hence, we would like to bring the attention of the
     Host-Host Protocol Glitch Cleaning Committe to this problem.
     The time limit T should be a Network Standard and its value should
     be decided also.

            Reason   Our NCP
            ------   -------

     1. User requests connection   1. RTS ->

     2. User gets tired requests CLS
        (or NCP timeout)   2. CLS ->

     3. No matching CLS returned in
        T time units   3. CLS assumed returned
                                                free socket and other
                                                resources

     4. User requests another connection
        over same socket pair   4. RTS ->

     5. CLS received   ?? does it belong to
                                             2 or 4?

                               Figure

         [This RFC was put into machine readable form for entry]
            [into the online RFC archives by Gert Doering 4/97]