Network Working Group
Request for Comments: 37

S. Crocker
UCLA
20 March 1970

Network Meeting Epilogue, etc.

The Meeting

On Tuesday, March 17, 1970, I hosted a Network meeting at UCLA. About 25 people attended, including representatives from MIT, LL, BBN, Harvard, SRI, Utah, UCSB, SDC, RAND and UCLA. I presented a modification of the protocol in NWG/RFC #33; the modifications are sketchily documented in NWG/RFC #36. The main modification is the facility for dynamic reconnection.

The protocol based on sockets and undistinguished simplex connections is quite different from the previous protocol as documented in NWG/RFC #11. The impetus for making such changes came out of the network meeting on December 8, 1969, at Utah, at which time the limitations of a log-in requirement and the inability to connect arbitrary processes was seriously challenged. Accordingly, the primary reason for the recent meeting was to sample opinion on the new protocol.

Recollections may vary, but it is my opinion that the protocol was widely accepted and that the criticism and discussion fell into two categories:

1. Questioning the complexity and usefulness of the full protocol, especially the need for dynamic reconnection.

2. Other topics, particularly character set translation, higher level languages, incompatible equipment, etc.

Notably lacking was any criticism of the basic concepts of sockets and connections. (Some have since surfaced.) The following agreements were made:

1. By April 30, I would be responsible for publishing an implementable specification along lines presented.

2. Any interested party would communicate with me (at least) immediately if he wished to modify the protocol.

3. If major modifications come under consideration, interested parties would meet again. This would happen in two to three weeks.

4. Jim Forgie of Lincoln Labs tentatively agreed to host a meeting on higher level network languages, probably near Spring Joint time.

Mailing List Changes

Paul Rovner of LL is replaced by

James Forgie
Mass. Institute of Technology
Lincoln Laboratory C158
P.O. Box 73
Lexington, Mass. 02173

telephone at (617) 862-5500 ext. 7173

Professor George MEaly is added

George Mealy
Rm. 220
Aitken Computation Lab.
Harvard University
Cambridge, Massachusetts 02138

telephone at (617) 868-1020 ext. 4355

Process

In all of our writing we have used the term process, to mean a program which has an assigned location counter and an address space. A program is merely a pattern of bits stored in some file. A new process is created only by an already existing process. The previous process must execute an atomic operation (forc, attach, etc.) to cause such a creation. Processes may either cause their own demise or be terminated by another (usually superior) process.

The above definition corresponds to the definition given by Vyssotsky, et al on pp. 206, 207 of "Structure of the Multics Supervisor" in the FJCC proceedings, 1965.

Because a process may create another process, and because in general the two processes are indistinguishable when viewed externally, I know of no reasonable way for two processes to request connection directly with each other. The function of sockets is to provide a standard interface between processes.

The Days After

In the time since the meeting I have had conversations with Steve Wolfe (UCLA-CCN), Bill Crowther (BBN), and John Heafner and Erick Harslem (RAND). Wolf's comments will appear as NWG/RFC #38 and fall into a class I will comment on below.

Crowther submitted the following:

"A brief description of two ideas for simplifying the host protocol described at the March meeting. These ideas have not been carefully worked out.

Idea 1. To Reconnect.

"A NCP wanting to Reconnect tells each of his neighbors "I want to reconnect". They wait until there are no messages in transit and respond "OK". He then says "Reconnect as follows" and they do it. In the Rare condition, the NCP gets back an "I want to reconnect instead of an "OK", then one must go and one must stop. So treat a "reconnect" from a higher Host user etc. as an ok and from a lower as a "No-wait until I reconnect you" and do the connection.

Idea 2

"Decouple connections and links. Still establish connections, but use any handy link for the messages. Don't send another message on a connection until a FRNM comes back. Include source and destination socket numbers in the packet.

"To reconnect, say to each of neighbors "please reconnect me as follows...". Hold onto the connect for a short time (seconds) and send both packets and connection messages along toward their destinations. I haven't worked out how to keep the in-transit messages in order, but probably everything works if you don't send out a reconnect when RFNM's are pending."

Bill's first idea does not seem to me to be either decisively better or (after some thought) very different, and I am considering it. I have no strong feelings about it yet, but I am trying to develop some.

Bill's second idea seems contrary to my conception of the role of links. An argument in favor of decoupling connections and links that the number of connections between two hosts might want to exceed 255, and that even if not, it is sounder practice to isolate dependancies in design. On the other hand, the newly provided Cease on Link facility* (page 22 of the soon to be released BBN report #1822 revised Febuary 1970) becomes useless. (Bill, who just put the feature in, doesn't care.) Another objection is that it seems intuitively bad to waste the possibility of using the link field to carry information. (Note the conflict of gut level feelings).

In a conversation with John Haefner and Eric Harslem of RAND, the pointed out that the current protocol makes no provision for error detection and reporting, status testing and reporting, and expansion and experimentation. Error detection and status testing will require some extended discussion to see what is useful, and I expect that such discussion will take place while implementation proceeds. Leaving room for protocol expansion and experimentation, however, is best done now.

I suggest that two areas for expansion be reserved. One is that only a fraction of the 256 links be used, say the first 32. The other area is to use command codes from 255 downward, with permanent codes assigned from the number of links in use to 32, I feel that it is quite unlikely that we would need more than 32 for quite some time, and moreover, the network probably wouldn't handle traffic implied by heavy link assignment. (These two things aren't necessarily strongly coupled: one can have many links assigned but only a few carrying traffic at any given time.)

Some of Heafner's and Harslen's other ideas may appear in NWG/RFC form.

Immediate Interaction

During the next several days, I will still be interested in those editicisms of current protocol which might lead to rejection or serious modification of it. Thereafter, the focus will be a refinement, implementation, extension, and utilization. I may be reached at UCLA through my secretary Mrs. Benita Kristel at (213) 825-2368. Also, everyone is invited to contribuet to the NWG/RFC series. Unique numbers are assigned by Benita.

* The Cease on Link facility is a way a receiving host modifies RFNM's so as to carry a flow-quenching meaning. An alternative proceedure is to use a host-to-host control command.