Two Protocol Suggestions to Reduce Congestion at Swap-Bound Nodes

There is a wide variance in swap rates between core and auxiliary store
among the HOST systems to be nodes in the ARPA IMP network. The slower
of these, of which our 360/50 system with 2303 drump swap store is an
example, might improve the utility of the network not only for
themselves but for all nodes if the two protocol suggestions of this
note were to be adopted.

1. HOST control of ordering of IMP-to-HOST traffic. IMP-HOST protocol
   now calls for delivery of messages from IMP to HOST in the order in
   which the IMP received them. This may lead to wasted swapping if,
   for example, the IMP has messages for its HOST's timeshare users A
   and B, in that order, at a time when user B is in HOST core. B
   would have to be swapped out, A in, and the first message accepted--
   only to discover that now A must be swapped out and B back in again.
   If the HOST could a) read the IMP's queue of waiting messages and b)
   accept them in the order it found most effective, then a new
   mechanism for improvement of network efficiency would be at hand.
   Clearly this change would have an impact on BBN's IMP software.

2. Core-to-core transfers between HOSTS. At another level, perhaps not
   involving HOST-IMP protocol or IMP software changes, is a HOST-HOST
   protocol wherein cooperating HOSTS agree to lock appropriate
   programs in core for the duration of a multi-message file transfer
   on an auxiliary connection. This could greatly reduce the time to
   transfer such a file to and from a swap-bound HOST. Unfortunately,
   the numbers mitigate possible advantages of this approach to some
   extent: if we assume a 50 kilobit/sec line and support further that
   it is dedicated at 100% efficiency to this transfer (which may
   require slightly different handling of RFNMs in this case) this
   comes out to just over 6 8-kilobit messages per second. It may be
   impolitic in a timeshare environment to lock a single program in
   core for more than about 2 seconds. If this is the case, then the
   method would be applicable only for the rather limited range of file
   sizes of 2-16 messages. Nevertheless, the time to move a large file
   could be so greatly enhanced by this approach that I think it
   deserves consideration.

1. Abhi Bhushan, Proj. MAC   10. Jerry Cole, SDC
2. Steve Crocker, UCLA   11. John Kreznar, "
3. Ron Stoughton, UCSB   12. Dick Linde,   "
4. Elmer Shapiro, SRI   13. Bob Long,   "

```
5. Steve Carr, Utah              14.  Reg Martin,   "
6. John Haefner, RAND            15.  Hal Sackman,  "
7. Paul Rovner, LL               16.  C. Weissman,  "
8. Bob Khan, BB & N              17.  Marty Bleier, "
9. Larry Roberts, ARPA


        [ This RFC was put into machine readable form for entry ]
          [ into the online RFC archives by Alex Portnoy 1/97 ]
```