Network Working Group   S. Crocker
Request for Comments: 36   16 March 1970

Protocol Notes

I Overview

The network protocol provides three facilities:

1. Connection establishment

2. Flow control

3. Reconnection

Reconnection is considered separately from connection establishment partly because of the complexity of reconnection and partly because I don't have enough experience with the protocol to present these concepts in an integrated fashion.

Connection Establishment

Connection establishment works essentially the same as in NWG/RFC #33. The major change is that a more general form of switching is provided independently of establishment, so establishment is simplified by not including switching procedures.

A rough scenario for connection establishment follows:

1. Process PA in host A grabs socket SA and requests connection with socket SB. Process PA accomplishes this through a system call.

2. Concurrently with the above, process PB in host B grabs socket SB and requests connection with socket SA.

3. In response to process PA's request, the network control program in host A (referred to as NCPA) sends a Request-for-Connection (RFC) command to host B. NCPB in host B sends a similar command to host A. No ordering is implied: NCPB may send the command to NCPA before or after receiving the command from NCPA.

4. NCPA and NCPB are both aware the connection is established when each has received a RFC command and each has received the RFNM for the one it has sent. They then notify processes PA and PB, respectively, that the connection is established.

One of the rules adhered to is that either SA is a send socket and SB is a receive socket or vice versa. This condition is sometimes stated as "SA and SB must be a send/receive pair."

5. The sending process may now send.

Flow Control

In order to prevent a sending process from flooding a receiving processes it is necessary for the receiving process to be able to stop the flow(*). Flow control is integrated into the network RFNM handling. When a receiving host wishes to inhibit flow on a particular link, the host sends a special message to its IMP which causes the next RFNM on that link to be modified. The sending host interprets this message as a RFNM and as a request to stop sending. A confirming control command is returned.

When the receiving host is ready to receive again, it sends a command (RSM) telling the sending host to resume sending.

Reconnection

For a great many reasons it is desirable to be able to switch one (or both) ends of a connection from one socket to another. Depending upon the restrictions placed upon the switching process, it may be easy or hard to implement. To achieve maximum generality, I present here a scheme for dynamic reconnection, which means that reconnection can take place even after flow has started. It may turn out that for the majority of cases, this scheme is much more expensive than it needs to be; however, the following virtues are claimed:

1. All various forms of switching connections are provided.

2. Reconnection introduces no overhead in the processing of messages sent over a connection i.e., the whole cost is borne in processing the protocol.

---

*BB&N argues that unlimited buffering should be provided. It is possible that this would be a proper strategy: but it is foreign to my way of thinking, and I have based the protocol design on the assumption that only a small buffer is provided on the receive end of each connection.

II Data Structures

1. Connection Table
2. Process Table
3. Input Link Table
4. Output Link Table
5. Link Assignment Table

Connection Table

This holds all information pertaining to local sockets, particularly whether a socket is engaged in a connection, and if so, what state the connection is in. Entries are keyed by local socket, but other tables have pointers into this table also. (See the Process Table, Input Link Table, and Output Link Table.)

Each entry contains the following information:

a) local socket (key)
b) foreign socket
c) link
d) connection state
e) flow state and buffer control
f) pointer to user's process
g) reconnection control state
h) queue of waiting callers

The local socket is a 32 bit number. If no entry exists for a particular socket, it may be created with null values.

The foreign socket is a 40 bit number. This field will be unassigned if no connection is established.

The link is an 8 bit number and is the link over which data is sent from the sender to the receiver. A socket is a receive socket iff its low-order bit is zero.

Connection state refers to whether a connection is open or not, etc. The following possibilities may occur.

a) local process has requested a connection
b) foreign process(es) has/have requested a connection
c) connection established
d) reconnection in progress
e) close waiting
f) reconnection waiting

Flow state and buffer control refer to checking for RFFNM's sending and accepting cease, suspend and resume commands, and keeping track of incoming or outgoing data.

A pointer to the user's process is necessary if the process has requested a connection.

If reconnection is in progress, it is necessary to keep track of the sequence of events. A socket engaged in reconnection is either an end or a middle. If it's a middle, it is necessary to store the eight bit name of the other middle attached to the same process, and to record receipt of END and RDY commands.

Finally, if RFC's are received either when the socket is busy or when no process has engaged it, the RFC's are stacked first-in-first-out on a queue for the named local socket.

Process Table

This table associates a process with a socket. It is used to process system calls.

Input Link Table

This table associates receive links with local sockets. It is used to decide for whom incoming messages are destined.

Output Link Table

This table associates send links with local sockets. It is used to interpret RFNM's and RSM commands.

Link Assignment Table

Links are assigned by receivers. This table shows which links are free.

III   Control Commands

Command Summary

```
   0         <NOP>
   1         <RFC> <me> <you>   or   <RFC> <me> <you> <link>
   2         <CLS> <me> <you>
   3         <RSM> <link>
   4         <SPD> <link>
   5         <FND> <me> <you> <asker>
   6         <END> <link> <end>
   7         <RDY> <link>
   8         <ASG> <me> <you> <link>
```

Commands

No Operation

```
   Form:    NOP
            NOP  is X'00'

   Purpose:  This command is included for completeness and
             convenience.
```

Request for connection

```
   Form:    <RFC> <my socket>  <your socket>
     or     <RFC> <my socket>  <your socket>  <link>
            <RFC> is X'01'
            <my socket> is a 32 bit socket number local to the
            sender
            <your socket> is a 32 bit socket number local to the
            receiver
            <link> is an eight bit link number.
            <my socket> and your socket must be a send/receive pair.
            <link> is included if and only if <my socket> is a
            receive socket

   Purpose:  This command is used to initiate a connection.  When
             two hosts have exchanged  RFC  commands with the same
             arguments (reversed), the connection is established.
             Links are assigned by the receiver.
```

```
Close

   Form:    <CLS> <my socket> <your socket>
            <CLS> is X'02'
            <my socket> and <your socket> are the same as for <RFC>

   Purpose:  This command is used to block a connection.  It may
             also be used to abort the establishment of a connection
             or to refuse a request.  It may happen that no
             connection between the named sockets was established,
             or was in the process of being established.  In this
             event, the <CLS> should be discarded.

Resume

   Form:    <RSM> <link>
            <RSM> is X'03'

   Purpose:  This command is sent by a receiving host to cause the
             sending host to resume transmission on the named link.
             A sending host suspends sending if it receives a
             special RFNM for some message.  (Special RFNM's are
             generated by the receiving IMP upon request by its
             host.)

Suspended

   Form:    <SPD> <link>
            <SPD> is X'04'

   Purpose:  This command is sent by a sending host to acknowledge
             that it has stopped sending over the named link.
             Transmission will resume if a <RSM> command is
             received.

Final End

   Form:    <FND> <my socket> <your socket> <asker>
            <FND> is X'05'
            <my socket> is a 32 bit socket number of a socket local
            to the sender
            <your socket> is a 32 bit socket number of a socket
            local to the receiver
            <my socket> and <your socket> form a send/receive pair.
            A connection should be established between them.
            <asker> is a 40 bit socket number of the same type as
            <my socket>
   Purpose:  If a process decides to short-circuit itself by connecting
             one of its receive sockets to one of its send sockets, the
             NCP sends out two <FND> commands -- one in each direction.
             Each one has <asker> initialized to <my socket>.

             Upon receiving an <FND> command, the NCP checks its <your
             socket>.  If <your socket> is already engaged in a
             reconnection, the command is passed on with a new <my
             socket> and <your socket>.  However, before it is passed
             on, the <asker> is compared with the new <my socket>.  If
             they are equal, a loop has been detected and both sockets
             are closed.

             If <your socket> is not engaged in a reconnection, it is
             marked as the end of a chain of reconnections and an <END>
             is sent back.

             If the connection named is not in progress, a <CLS> is sent
             back and the <FND> is discarded.

End Found

   Form:    <END> <link> <end socket>

            <END> is X'06'
            <link> is an 8 bit link

            <end socket> is a 40 bit socket

   Purpose:  This command indicates which socket is at the end of a
             chain of reconnections.  It is generated at <end
             socket> and passed back to the other terminal socket
             via all the intermediate sockets.  If <end socket> is a
             send socket, <link> refers to a connection with the
             send socket in the sending host and the receive socket
             in the receiving host.  If <end socket> is a receive
             socket, <link> refers to a connection with the send
             socket in the receiving host and the receive socket in
             the sending hose.  ("sending" end "receiving" refer to
             the transmission of this control command.)
```

Ready

```
   Form:    <RDY> <link>

            <RDY> is X'07'
            <link> is an 8 bit link number

   Purpose:  This command is sent from a send socket to a receive
             socket to indicate that all messages have been
             forwarded and that reconnection may occur.
```

Assign New Link

```
   Form:    <ASG> <my socket> <your socket> <link>

            <ASG> is X'08'

   Purpose:  This command completes a reconnection.  It is sent from
             a receive socket to a send socket after the receive
             socket has received a <RDY>.  A new link is assigned
             and transmission commences.
```

[This RFC was put into machine readable form for entry]
[into the online RFC archives by Marc Blanchett 3/00]