Network Working Group
Request for Comments: 438
NIC: 13770
References: 354,385,414,418

B. Thomas
B. Clements
BBN-TENEX

15 January 1973

FTP Server-Server Interaction

The current ARPANET File Transfer Protocol as specified by RFC 354 and updated by RFC's 385, 414 and 418 allows for "third host" participation but does not specify a mechanism by which the process at the third site may be the FTP server at that site. This RFC suggests a simple extension to FTP which would allow an FTP user process at one site to arrange for FTP server processes at other sites to act cooperatively on its behalf.

Such server-server cooperation may appear to be of limited utility. Consider, however, the requirements placed on FTP by a Resource Sharing Executive (RSEXEC) program - a command language interpreter which extends the range of a user's commands beyond the boundaries of the user's local system. Among its services such as RSEXEC could provide its users with a network-wide file system, perhaps allowing, in certain contexts, the use of partially qualified pathnames which omit site specification. Consider, for example the response of the RSEXEC to the user command:

```
APPEND (FILE) PROG1.PL1 (TO FILE) PROG2.PL1
```

for the case in which the two files are at different sites (PROG1.PL1 at SITE1, PROG2.PL1 at SITE2) neither of which is the user's site. A straightforward way for the RSEXEC to "perform" the APPEND would be to establish FTP control connections to the FTP servers at SITE1 and at SITE2, instruct the server at SITE1 to

```
RETR PROG1.PL1
```

using data connection C and instruct the server at SITE2 to

```
APPE PROG2.PL1
```

using the same data connection C.

Unfortunately, at present there is no way within FTP to arrange for such server-server cooperation. This is due primarily to the lack of symmetry in the way that FTP treats the ends of data connections during connection establishment. It specifies one end to be the "server" end, the other to be the "user" end and specifies different means for establishing the connections from the two ends.

FTP could be modified to support server-server interaction by:

1. making the establishment of data connections symmetric, or;

2. providing a mechanism for instructing a server to establish its end of a data connection as if it were a user.

The second alternative probably requires fewer changes to the existing protocol.

The following proposed extension to FTP uses the second method. It involves the addition of a single new command (LSTN) and minor modifications to several existing commands (SOCK, APPE, RETR, STOR):

a. The LSTN (Listen) command requests the FTP server to allocate a socket for use as a data connection. To establish the corresponding data connection the server is to "listen" on the socket allocated when an appropriate transfer command is given.

   syntax: LSTN <direction> CRLF

      where <direction> is either "S" for send or "R" for receive.

   The server responds to LSTN by:

      1. refusing to allocate such a socket, or:

      2. sending the user the number of the socket allocated (the 255 FTP server data socket reply could be used for this purpose).

b. Receipt of an appropriate STOR, RETR or APPE command following a successful LSTN command causes the server to "listen" for an RFC for the socket allocated. Data transfer may proceed after the server receives an RFC for the socket and responds with a matching RFC. Once established, a data connection corresponding to a successful LSTN command has the same duration as one established in the usual way.

c. The user may insure the security of his data transfer by using the SOCK command to instruct the server to accept an RFC for the listening socket only if it is from a specified host and socket.

d. The SOCK command is modified in two ways:

1. On success the reply must be the 255 FTP server data socket reply; that is, the 255 reply can not be deferred until receipt of a data transfer command. (This is to allow the user to transmit the server's response to the program at the third site; see the example below.)

2. After a LSTN command the SOCK command is to be interpreted by the server as specification of the acceptable RFC for subsequent data transfer command that use the allocated socket.

With this extension to FTP, the RSEXEC program could accomplish the APPEND in the example above as follows:

```
     to SITE1:                          to SITE2:

        .                                  .
        .                                  .
        .                                  .

1.                                      LSTN R CRLF
                                        (let X = socket
                                         allocated)

2.   SOCK SITE2,X CRLF
     (let Y = socket in 255
      reply from SITE1)

3.                                      SOCK SITE1,Y CRLF

4.  RETR PROG1.PL1                      APPE PROG2.Pl1 CRLF

        .                                  .
        .                                  .
        .                                  .
```

In closing it is appropriate to note that an experimental RSEXEC program of the sort suggested above has been operational on TENEXs for about 8 months. It currently uses a private, resource sharing protocol (RSP) that includes file transfer operations. RSP supports server-server cooperation; in RSP data connections are established in a symmetric way (alternative 1 above).

[This RFC was put into machine readable form for entry]
[into the online RFC archives by Mirsad Todorovac 5/98]