Network Working Group
Request for Comments: 1579
Category: Informational

S. Bellovin
AT&T Bell Laboratories

February 1994

# Firewall-Friendly FTP

## Status of this Memo

This document provides information for the Internet community. This document does not specify an Internet standard of any kind. Distribution of this document is unlimited.

## Abstract

This memo describes a suggested change to the behavior of FTP client programs. No protocol modifications are required, though we outline some that might be useful.

## Overview and Rational

The FTP protocol [1] uses a secondary TCP connection for actual transmission of files. By default, this connection is set up by an active open from the FTP server to the FTP client. However, this scheme does not work well with packet filter-based firewalls, which in general cannot permit incoming calls to random port numbers.

If, on the other hand, clients use the PASV command, the data channel will be an outgoing call through the firewall. Such calls are more easily handled, and present fewer problems.

## The Gory Details

The FTP specification says that by default, all data transfers should be over a single connection. An active open is done by the server, from its port 20 to the same port on the client machine as was used for the control connection. The client does a passive open.

For better or worse, most current FTP clients do not behave that way. A new connection is used for each transfer; to avoid running afoul of TCP's TIMEWAIT state, the client picks a new port number each time and sends a PORT command announcing that to the server.

Neither scenario is firewall-friendly. If a packet filter is used (as, for example, provided by most modern routers), the data channel requests appear as incoming calls to unknown ports. Most firewalls are constructed to allow incoming calls only to certain believed-to-be-safe ports, such as SMTP. The usual compromise is to block only

the "server" area, i.e., port numbers below 1024. But that strategy is risky; dangerous services such as X Windows live at higher-numbered ports.

Outgoing calls, on the other hand, present fewer problems, either for the firewall administrator or for the packet filter. Any TCP packet with the ACK bit set cannot be the packet used to initiate a TCP connection; filters can be configured to pass such packets in the outbound direction only. We thus want to change the behavior of FTP so that the data channel is implemented as a call from the client to the server.

Fortunately, the necessary mechanisms already exist in the protocol. If the client sends a PASV command, the server will do a passive TCP open on some random port, and inform the client of the port number. The client can then do an active open to establish the connection.

There are a few FTP servers in existence that do not honor the PASV command. While this is unfortunate (and in violation of STD 3, RFC 1123 [2]), it does not pose a problem. Non-conforming implementations will return a "500 Command not understood" message; it is a simple matter to fall back to current behavior. While it may not be possible to talk to such sites through a firewall, that would have been the case had PASV not been adopted.

## Recommendation

We recommend that vendors convert their FTP client programs (including FTP proxy agents such as Gopher [3] daemons) to use PASV instead of PORT. There is no reason not to use it even for non-firewall transfers, and adopting it as standard behavior will make the client more useful in a firewall environment.

STD 3, RFC 1123 notes that the format of the response to a PASV command is not well-defined. We therefore recommend that FTP clients and servers follow the recommendations of that RFC for solving this problem.

## Discussion

Given the behavior of most current FTP clients, the use of PASV does not cause any additional messages to be sent. In all cases, a transfer operation is preceded by an extra exchange between the client and the server; it does not matter if that exchange involves a PORT command or a PASV command.

There is some extra overhead with Gopher-style clients; since they transfer exactly one file per control channel connection, they do not

need to use PORT commands. If this is a serious concern, the Gopher proxy should be located on the outside of the firewall, so that it is not hampered by the packet filter's restrictions.

If we accept that clients should always perform active opens, it might be worthwhile enhancing the FTP protocol to eliminate the extra exchange entirely. At startup time, the client could send a new command APSV ("all passive"); a server that implements this option would always do a passive open. A new reply code 151 would be issued in response to all file transfer requests not preceded by a PORT or PASV command; this message would contain the port number to use for that transfer. A PORT command could still be sent to a server that had previously received APSV; that would override the default behavior for the next transfer operation, thus permitting third-party transfers.

## Implementation Status

At least two independent implementations of the modified clients exist. Source code to one is freely available. To our knowledge, APSV has not been implemented.

## Security Considerations

Some people feel that packet filters are dangerous, since they are very hard to configure properly. We agree. But they are quite popular. Another common complaint is that permitting arbitrary outgoing calls is dangerous, since it allows free export of sensitive data through a firewall. Some firewalls impose artificial bandwidth limits to discourage this. While a discussion of the merits of this approach is beyond the scope of this memo, we note that the sort of application-level gateway necessary to implement a bandwidth limiter could be implemented just as easily using PASV as with PORT.

Using PASV does enhances the security of gateway machines, since they no longer need to create ports that an outsider might connect to before the real FTP client. More importantly, the protocol between the client host and the firewall can be simplified, if there is no need to specify a "create" operation.

Concerns have been expressed that this use of PASV just trades one problem for another. With it, the FTP server must accept calls to random ports, which could pose an equal problem for its firewall. We believe that this is not a serious issue, for several reasons.

First, there are many fewer FTP servers than there are clients. It is possible to secure a small number of special-purpose machines, such as gateways and organizational FTP servers. The firewall's

filters can be configured to allow access to just these machines. Further precautions can be taken by modifying the FTP server so that it only uses very high-numbered ports for the data channel. It is comparatively easy to ensure that no dangerous services live in a given port range. Again, this is feasible because of the small number of servers.

## References

[1] Postel, J., and J. Reynolds, "File Transfer Protocol", STD 1, RFC 959, USC/Information Sciences Institute, October 1985.

[2] Braden, R., Editor, "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, USC/Information Sciences Institute, October 1989.

[3] Anklesaria, F., McCahill, M., Lindner, P., Johnson, D., Torrey, D., and B. Alberti, "The Internet Gopher Protocol (a distributed document search and retrieval protocol)", RFC 1436, University of Minnesota, March 1993.

## Author's Address

Steven M. Bellovin
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974

Phone: (908) 582-5886
EMail: smb@research.att.com