FTP Security Considerations

Abstract

   The specification for the File Transfer Protocol (FTP) contains a
   number of mechanisms that can be used to compromise network security.
   The FTP specification allows a client to instruct a server to
   transfer files to a third machine. This third-party mechanism, known
   as proxy FTP, causes a well known security problem. The FTP
   specification also allows an unlimited number of attempts at entering
   a user's password. This allows brute force "password guessing"
   attacks. This document provides suggestions for system
   administrators and those implementing FTP servers that will decrease
   the security problems associated with FTP.

1   Introduction

   The File Transfer Protocol specification (FTP) [PR85] provides a
   mechanism that allows a client to establish an FTP control connection
   and transfer a file between two FTP servers. This "proxy FTP"
   mechanism can be used to decrease the amount of traffic on the
   network; the client instructs one server to transfer a file to
   another server, rather than transferring the file from the first
   server to the client and then from the client to the second server.
   This is particularly useful when the client connects to the network
   using a slow link (e.g., a modem). While useful, proxy FTP provides
   a security problem known as a "bounce attack" [CERT97:27]. In
   addition to the bounce attack, FTP servers can be used by attackers
   to guess passwords using brute force.

This document does not contain a discussion of FTP when used in
conjunction with strong security protocols, such as IP Security.
These security concerns should be documented, however they are out of
the scope of this document.

This paper provides information for FTP server implementers and
system administrators, as follows. Section 2 describes the FTP
"bounce attack". Section 3 provides suggestions for minimizing the
bounce attack. Section 4 provides suggestions for servers which
limit access based on network address. Section 5 provides
recommendations for limiting brute force "password guessing" by
clients. Next, section 6 provides a brief discussion of mechanisms
to improve privacy. Section 7 provides a mechanism to prevent user
identity guessing. Section 8 discusses the practice of port
stealing. Finally, section 9 provides an overview of other FTP
security issues related to software bugs rather than protocol issues.

2   The Bounce Attack

The version of FTP specified in the standard [PR85] provides a method
for attacking well known network servers, while making the
perpetrators difficult to track down. The attack involves sending an
FTP "PORT" command to an FTP server containing the network address
and the port number of the machine and service being attacked. At
this point, the original client can instruct the FTP server to send a
file to the service being attacked. Such a file would contain
commands relevant to the service being attacked (SMTP, NNTP, etc.).
Instructing a third party to connect to the service, rather than
connecting directly, makes tracking down the perpetrator difficult
and can circumvent network-address-based access restrictions.

As an example, a client uploads a file containing SMTP commands to an
FTP server. Then, using an appropriate PORT command, the client
instructs the server to open a connection to a third machine's SMTP
port. Finally, the client instructs the server to transfer the
uploaded file containing SMTP commands to the third machine. This
may allow the client to forge mail on the third machine without
making a direct connection. This makes it difficult to track
attackers.

3   Protecting Against the Bounce Attack

The original FTP specification [PR85] assumes that data connections
will be made using the Transmission Control Protocol (TCP) [Pos81].
TCP port numbers in the range 0 - 1023 are reserved for well known
services such as mail, network news and FTP control connections
[RP94]. The FTP specification makes no restrictions on the TCP port
number used for the data connection. Therefore, using proxy FTP,

clients have the ability to tell the server to attack a well known
service on any machine.

To avoid such bounce attacks, it is suggested that servers not open
data connections to TCP ports less than 1024. If a server receives a
PORT command containing a TCP port number less than 1024, the
suggested response is 504 (defined as "Command not implemented for
that parameter" by [PR85]). Note that this still leaves non-well
known servers (those running on ports greater than 1023) vulnerable
to bounce attacks.

Several proposals (e.g., [AOM98] and [Pis94]) provide a mechanism
that would allow data connections to be made using a transport
protocol other than TCP. Similar precautions should be taken to
protect well known services when using these protocols.

Also note that the bounce attack generally requires that a
perpetrator be able to upload a file to an FTP server and later
download it to the service being attacked. Using proper file
protections will prevent this behavior. However, attackers can also
attack services by sending random data from a remote FTP server which
may cause problems for some services.

Disabling the PORT command is also an option for protecting against
the bounce attack. Most file transfers can be made using only the
PASV command [Bel94]. The disadvantage of disabling the PORT command
is that one loses the ability to use proxy FTP, but proxy FTP may not
be necessary in a particular environment.

4   Restricted Access

For some FTP servers, it is desirable to restrict access based on
network address. For example, a server might want to restrict access
to certain files from certain places (e.g., a certain file should not
be transferred out of an organization). In such a situation, the
server should confirm that the network address of the remote hosts on
both the control connection and the data connection are within the
organization before sending a restricted file. By checking both
connections, a server is protected against the case when the control
connection is established with a trusted host and the data connection
is not. Likewise, the client should verify the IP address of the
remote host after accepting a connection on a port opened in listen
mode to verify that the connection was made by the expected server.

Note that restricting access based on network address leaves the FTP
server vulnerable to "spoof" attacks. In a spoof attack, for
example, an attacking machine could assume the host address of
another machine inside an organization and download files that are

not accessible from outside the organization. Whenever possible,
secure authentication mechanisms should be used, such as those
outlined in [HL97].

5   Protecting Passwords

To minimize the risk of brute force password guessing through the FTP
server, it is suggested that servers limit the number of attempts
that can be made at sending a correct password. After a small number
of attempts (3-5), the server should close the control connection
with the client. Before closing the control connection the server
must send a return code of 421 ("Service not available, closing
control connection." [PR85]) to the client. In addition, it is
suggested that the server impose a 5 second delay before replying to
an invalid "PASS" command to diminish the efficiency of a brute force
attack. If available, mechanisms already provided by the target
operating system should be used to implement the above suggestions.

An intruder can subvert the above mechanisms by establishing
multiple, parallel control connections to a server. To combat the
use of multiple concurrent connections, the server could either limit
the total number of control connections possible or attempt to detect
suspicious activity across sessions and refuse further connections
from the site. However, both of these mechanisms open the door to
"denial of service" attacks, in which an attacker purposely initiates
the attack to disable access by a valid user.

Standard FTP [PR85] sends passwords in clear text using the "PASS"
command. It is suggested that FTP clients and servers use alternate
authentication mechanisms that are not subject to eavesdropping (such
as the mechanisms being developed by the IETF Common Authentication
Technology Working Group [HL97]).

6   Privacy

All data and control information (including passwords) is sent across
the network in unencrypted form by standard FTP [PR85]. To guarantee
the privacy of the information FTP transmits, a strong encryption
scheme should be used whenever possible. One such mechanism is
defined in [HL97].

7   Protecting Usernames

Standard FTP [PR85] specifies a 530 response to the USER command when
the username is rejected. If the username is valid and a password is
required FTP returns a 331 response instead. In order to prevent a
malicious client from determining valid usernames on a server, it is
suggested that a server always return 331 to the USER command and

then reject the combination of username and password for an invalid
username.

8   Port Stealing

Many operating systems assign dynamic port numbers in increasing
order. By making a legitimate transfer, an attacker can observe the
current port number allocated by the server and "guess" the next one
that will be used. The attacker can make a connection to this port,
thus denying another legitimate client the ability to make a
transfer. Alternatively, the attacker can steal a file meant for a
legitimate user. In addition, an attacker can insert a forged file
into a data stream thought to come from an authenticated client.
This problem can be mitigated by making FTP clients and servers use
random local port numbers for data connections, either by requesting
random ports from the operating system or using system dependent
mechanisms.

9   Software-Base Security Problems

The emphasis in this document is on protocol-related security issues.
There are a number of documented FTP security-related problems that
are due to poor implementation as well. Although the details of
these types of problems are beyond the scope of this document, it
should be pointed out that the following FTP features has been abused
in the past and should be treated with great care by future
implementers:

Anonymous FTP

    Anonymous FTP refers to the ability of a client to connect to an
    FTP server with minimal authentication and gain access to public
    files. Security problems arise when such a user can read all
    files on the system or can create files. [CERT92:09] [CERT93:06]

Remote Command Execution

    An optional FTP extension, "SITE EXEC", allows clients to execute
    arbitrary commands on the server. This feature should obviously
    be implemented with great care. There are several documented
    cases of the FTP "SITE EXEC" command being used to subvert server
    security [CERT94:08] [CERT95:16]

Debug Code

    Several previous security compromises related to FTP can be
    attributed to software that was installed with debugging features
    enabled [CERT88:01].

   This document recommends that implementors of FTP servers with these
   capabilities review all of the CERT advisories for attacks on these
   or similar mechanisms before releasing their software.

10 Conclusion

   Using the above suggestions can decrease the security problems
   associated with FTP servers without eliminating functionality.

11 Security Considerations

   Security issues are discussed throughout this memo.

Acknowledgments

   We would like to thank Alex Belits, Jim Bound, William Curtin, Robert
   Elz, Paul Hethmon, Alun Jones and Stephen Tihor for their helpful
   comments on this paper. Also, we thank the FTPEXT WG members who
   gave many useful suggestions at the Memphis IETF meeting.

References

   [AOM98]   Allman, M., Ostermann, S. and C. Metz, "FTP Extensions
               for IPv6 and NATs", RFC 2428, September 1998.

   [Bel94]   Bellovin. S., "Firewall-Friendly FTP", RFC 1579, February
               1994.

   [CERT88:01] CERT Advisory CA-88:01. ftpd Vulnerability. December,
               1988 ftp://info.cert.org/pub/cert_advisories/

   [CERT92:09] CERT Advisory CA-92:09. AIX Anonymous FTP Vulnerability.
               April 27, 1992. ftp://info.cert.org/pub/cert_advisories/

   [CERT93:06] CERT Advisory CA-93:06. Wuarchive ftpd Vulnerability.
               September 19,1997
               ftp://info.cert.org/pub/cert_advisories/

   [CERT94:08] CERT Advisory CA-94:08. ftpd Vulnerabilities. September
               23, 1997. ftp://info.cert.org/pub/cert_advisories/

   [CERT95:16] CERT Advisory CA-95:16. wu-ftpd Misconfiguration
               Vulnerability. September 23, 1997
               ftp://info.cert.org/pub/cert_advisories/

   [CERT97:27] CERT Advisory CA-97.27. FTP Bounce. January 8, 1998.
               ftp://info.cert.org/pub/cert_advisories/

   [HL97]   Horowitz, M. and S. Lunt, "FTP Security Extensions", RFC
                 2228, October 1997.

   [Pis94]   Piscitello, D., "FTP Operation Over Big Address Records
                 (FOOBAR), RFC 1639, June 1994.

   [Pos81]   Postel, J., "Transmission Control Protocol", STD 7, RFC
                 793, September 1981.

   [PR85]   Postel, J. and J. Reynolds, "File Transfer Protocol
                 (FTP)", STD 9, RFC 959, October 1985.

   [RP94]   Reynolds, J. and J. Postel, "Assigned Numbers", STD 2,
                 RFC 1700, October 1994. See also:
                 http://www.iana.org/numbers.html

Authors' Addresses

   Mark Allman
   NASA Glenn Research Center/Sterling Software
   21000 Brookpark Rd.   MS 54-2
   Cleveland, OH   44135

   EMail: mallman@grc.nasa.gov

   Shawn Ostermann
   School of Electrical Engineering and Computer Science
   Ohio University
   416 Morton Hall
   Athens, OH   45701

   EMail: ostermann@cs.ohiou.edu

Full Copyright Statement

Acknowledgement